Fake News Detection Using Large Language Models

Lynda Abbas, Zara Afreen, Elena Petukhova, Nafiseh Vahidian

I. Introduction

In today's digital era, the widespread use of the internet and social media has dramatically transformed the way information is created, disseminated, and consumed. While these advancements offer unprecedented access to knowledge and communication, they have also facilitated the rapid spread of misinformation, particularly fake news. Fake news can originate from disinformation (intentionally fabricated content for specific purposes) or evolve as misinformation (false content shared unknowingly by individuals who believe it to be true). Psychological factors, such as confirmation bias, further exacerbate the issue; people tend to accept and share information that aligns with their personal beliefs, regardless of its factual accuracy. Behavioural studies suggest that exposure to fake news increases the likelihood of belief in the content, and even when individuals recognize the content as false, they may still share it to affirm social identity, signal political alignment, or elicit emotional reactions from their network [2] .

An MIT study that tracked 126,000 stories revealed an alarming trend, fake news stories are approximately 70% more likely to be retweeted on Twitter than factual stories and reach audiences six times faster than the truth [7]. The proliferation of fake news has severe implications for societal trust and democratic institutions. For example, recent surveys indicate that two-thirds of EU citizens encounter fake news at least once a week, and over 80% consider it a significant issue for their country and democracy in general [18].

As the challenge of combating fake news becomes increasingly urgent, researchers have turned to artificial intelligence (AI) solutions to address this complex and evolving problem. Initially, traditional machine learning (ML) models were employed to detect fake news by analyzing patterns in text data. These models used various natural language processing (NLP) techniques to preprocess the data before classification. This preprocessing phase often involved tasks such as tokenization (breaking text into smaller units like words or phrases), lemmatization (reducing words to their base forms), and the removal of stop words (common words like "and," "the," or "is," which don't carry significant meaning in this context). The preprocessed data was then passed through various ML algorithms, such as Decision Trees, Support Vector Machines (SVMs), or gradient boosting models like XGBoost, which were tasked with identifying linguistic features, sentiment markers, or other cues that might indicate falsified or biased information. These traditional models offered some success in detecting fake news but often faced challenges in dealing with the complexities and evolving nature of the language used in modern misinformation campaigns. For example, they struggled to account for the subtleties of context, irony, and manipulation that are commonly used in fake news stories [22], [25].

In response to these challenges, recent breakthroughs in Large Language Models (LLMs), such as OpenAI's GPT models, have revolutionized the domain of fake news detection. LLMs, which are built with billions of parameters and trained on massive, diverse datasets, have an exceptional ability to understand the intricacies of language and context [6]. Unlike traditional ML models, which rely on manually defined features, LLMs can grasp the broader meaning of text, including contextual nuances, which are crucial in identifying misleading or false information [9]. For example, these models can better detect the subtle ways in which fake news might manipulate facts or appeal to emotions through language that might be overlooked by traditional models.

The key advantage of LLMs is their ability to go beyond simple feature extraction, moving towards a more sophisticated semantic and contextual understanding of news content. This allows them to analyze not just the surface-level linguistic elements, but also the deeper meaning, tone, and intent behind the words. As a result, LLMs have significantly improved the accuracy and robustness of fake news detection, outperforming older ML models in detecting even the most subtle or deceptive forms of misinformation. By focusing on the broader context and understanding how information is framed, LLMs are helping to address the growing problem of fake news in ways that traditional models simply could not [4], [24].

In our paper, we aim to investigate two key questions that remain central to the effectiveness of AI-driven fake news detection:

1. Are LLMs more accurate in detecting fake news than traditional ML algorithms? This question seeks to assess the performance differences between traditional ML algorithms and modern LLMs in identifying fake news. By comparing metrics such as accuracy, precision, recall, and F1-score across various models, we aim to quantify whether the contextual understanding afforded by LLMs translates to better detection rates.

2. Do hybrid models that combine Large Language Models (LLMs) with traditional machine learning models or Small Language Models (SMLs) offer improvements in fake news detection? Hybrid models, which combine the strengths of LLMs with more lightweight or interpretable models, represent an intriguing area of research. These combinations could balance the computational intensity of LLMs with the efficiency and interpretability of smaller models, potentially

enhancing overall model performance and providing faster, more resource-efficient solutions.

II. Related Work

The problem of fake news detection has been extensively studied, with advancements spanning traditional machine learning approaches to the adoption of large language models (LLMs). This section provides a detailed review of relevant works, focusing on specific methodologies and their contributions.

A. LIAR Dataset

1) Fake News Detection Using Machine Learning Approaches: Khanam et al. [12] examined the application of traditional supervised machine learning methods, including Support Vector Machines (SVMs), Random Forest, and XGBoost, for fake news detection. Using TF-IDF for feature extraction, the XGBoost classifier achieved the highest accuracy of 75%. While computationally efficient, the study highlighted the limited scalability of such approaches to multi-modal datasets and their inability to leverage contextual understanding.

2) Fake Detect: A Deep Learning Ensemble Model for Fake News Detection: Aslam et al. [1] proposed a deep learning ensemble model for fake news detection, leveraging the LIAR dataset. The authors employed a hybrid approach using two deep learning models: a Bi-LSTM-GRU network for textual attributes (statements) and a dense model for non-textual attributes (e.g., speaker's job title and context). Preprocessing steps included tokenization, lemmatization, stop-word removal, and word embeddings using FastText.

The proposed model achieved an accuracy of 89.8% and an F1-score of 0.914, outperforming traditional machine learning methods and prior CNN-based approaches. However, challenges included the limited contribution of non-textual attributes to classification and reliance on a single dataset, which limited generalizability to other domains.

3) An Ensemble Machine Learning Approach to Classify Fake News: Hakak et al. [8] developed an ensemble machine learning approach that focused on effective feature extraction. They used the LIAR and ISOT datasets. The study emphasized robust preprocessing techniques, such as tokenization, noise removal, and Named Entity Recognition (NER) to extract features like word count and average sentence length.

The ensemble model combined Decision Tree, Random Forest, and Extra Tree classifiers using a bagging approach for stability. Results demonstrated 100% accuracy on the ISOT dataset and 99.96% training accuracy with 44.15% testing accuracy on the LIAR dataset. However, generalization remained an issue, particularly for the LIAR dataset, due to its complexity and multi-class nature.

4) Fake News Prediction Using Machine Learning Approaches: Mushtaq et al. [17] focused on fake news prediction using the LIAR dataset. Their research employed machine learning classifiers, including Naïve Bayes, Random Forest, Decision Tree, and Neural Networks. Preprocessing steps included data cleaning to remove noise and unnecessary symbols and statistical feature extraction, such as analyzing word distributions and subject categories.

The study highlighted the effectiveness of Naïve Bayes, which achieved a 99% accuracy due to its ability to reduce variance and mitigate overfitting. Compared to other classifiers, Naïve Bayes required less computational time while maintaining high precision and recall. Despite the promising results, challenges included limited exploration of deep learning approaches and reliance on static benchmark datasets, which may not reflect the real-time complexities of fake news on social media.

5) A Better Large Language Model Using LoRA for False News Recognition System: Tiwari [21] introduced a framework leveraging Low-Rank Adaptation (LoRA) to fine-tune the LLaMA2-7B language model for fake news detection. LoRA reduces the computational demands of training large-scale language models by decomposing their weight matrices into low-rank components, enabling task-specific adaptation with fewer parameters. The study utilized datasets such as the COVID-19 FakeNews dataset, LIAR dataset, and the FakeNews Challenge dataset. The preprocessing pipeline included text normalization and imbalance correction through class weighting.

The results showed significant performance improvements, with accuracy reaching 97.33% on the COVID-19 dataset, 98.66% on the FakeNews Challenge dataset and 62.67% on the LIAR dataset. These findings underscore the effectiveness of LoRA in optimizing LLMs for resource-constrained environments. Despite these successes, the study noted diminishing returns with extended training durations and emphasized the need for further research on efficient adaptation techniques.

6) A Novel Framework for Fake News Detection Using Double Layer BI-LSTM: Merryton and Augasta [16] developed a novel framework based on Double Layer Bi-LSTM for enhancing fake news detection. By stacking two Bi-LSTM layers, the model captures both short-term and long-term dependencies in textual data. The framework combines traditional preprocessing techniques, such as the Porter Stemmer and TF-IDF vectorization, with deep learning for feature extraction.

The study evaluated the framework on three datasets: the Kaggle Fake_Real_News, LIAR, and Politifact datasets. Results showed a 97.58% accuracy on Kaggle and 83% accuracy on Politifact. However, the model underperformed on the LIAR dataset (61.19%), likely due to its limited ability to handle highly imbalanced classes and nuanced text features. This indicates potential areas for improvement, such as incorporating attention mechanisms for better feature weighting.

7) Fighting Lies with Intelligence: Using Large Language Models and Chain of Thoughts Technique to Combat Fake News: Kareem and Abbas [11] introduced the Chain of Thoughts (CoT) reasoning approach to enhance the interpretability and accuracy of fake news detection systems. By fine-tuning FLAN-T5 and LLaMA-2 with CoT annotations, the models were able to provide logical justifications for their predictions. The accuracy of the initial model of 39. 25% improved when the classifications were reduced to true or false, achieving accuracy of 84. 26% in a dataset enriched with CoT-annotated records.

While the CoT approach improved transparency, challenges included limited performance gains on multi-class datasets and the need for richer annotation schemes. The study concluded with recommendations for integrating CoT with multimodal data for enhanced generalizability.

8) Re-Search for the Truth: Multi-Round Retrieval-Augmented LLMs for Fake News Detection: Li et al. [14] proposed the STEEL framework, which combines multi-round retrieval mechanisms with LLMs for dynamic evidence collection and claim verification. By sequentially retrieving high-quality evidence until confidence thresholds are met, the framework outperformed traditional single-retrieval methods.

STEEL was evaluated on the LIAR (True and False Classes), CHEF, and PolitiFact datasets, achieving F1-Macro scores of 0.714, 0.793, and 0.751, respectively. The multi-round retrieval mechanism significantly improved accuracy in detecting fake news. However, reliance on internet accessibility and the limitations of LLM context lengths were noted as challenges.

9) Fake News Detection with Large Language Models on the LIAR Dataset: Boissonneault and Hensen [3] conducted a detailed evaluation of LLMs like ChatGPT and Google Gemini in the LIAR dataset (True and False Classes). Google Gemini achieved an accuracy of 89.4%, outperforming ChatGPT in terms of precision and recall. Despite these strong results, the study highlighted limitations in handling nuanced contextual information, suggesting the need for domain-specific fine-tuning to improve the detection of subtle misinformation.

B. Other Datasets

1) Integrating Large Language Models and Machine Learning for Fake News Detection: Teo et al. [20] proposed a hybrid method combining LLMs with traditional machine learning algorithms, specifically using ChatGPT-3.5 outputs as features for XGBoost classifiers. This integration leveraged the contextual strengths of LLMs and the efficiency of XGBoost, achieving an accuracy of 96.39%. The study emphasized the potential of hybrid models to balance interpretability and computational efficiency, particularly in resource-constrained settings.

2) CSI: A Hybrid Deep Model for Fake News Detection: Ruchansky et al. [19] presented the CSI framework, which integrates three key elements: content, social context, and user engagement behaviour. The model employs a hybrid deep architecture combining LSTM networks for temporal analysis and Singular Value Decomposition (SVD) for user behaviour analysis. This approach uniquely captures temporal and user-level patterns, which are often critical for distinguishing fake news from legitimate content.

The study utilized datasets from Twitter and Weibo, achieving an accuracy of 89.2% on Twitter and 95.3% on Weibo. The inclusion of user behaviour scores provided additional insights into suspicious activities, enabling a more interpretable classification process. However, the model's reliance on large-scale, annotated user interaction data poses challenges for generalizability across platforms with limited or incomplete engagement data.

3) Bad Actor, Good Advisor: Exploring the Role of Large Language Models in Fake News Detection: Hu et al. [10] proposed the Adaptive Rationale Guidance (ARG) network, which leverages LLMs like GPT-3.5 as advisors rather than decision-makers. ARG integrates LLM-generated rationales into the decision-making process of small language models (SLMs). This hybrid approach achieved 10-15% higher accuracy compared to standalone models while reducing computational demands. However, querying LLMs remains resource-intensive, and selective sampling methods were suggested to mitigate this issue.

4) Evaluating the Efficacy of Large Language Models in Detecting Fake News: A Comparative Analysis: Koka et al. [13] conducted a comparative analysis of six LLMs, including GPT-4, Claude, and Mistral, to evaluate their performance in fake news detection. The study employed a balanced dataset of 30 articles and used zero-shot prompting for classification. Results revealed that larger models, such as Claude and GPT-4, achieved near-perfect accuracy and F1 scores, while smaller models like Mistral 7B exhibited higher false-positive rates.

The authors emphasized the superior contextual understanding of larger models but noted that the limited dataset size restricted generalizability. Their future work includes expanding datasets and exploring ensemble methods to integrate outputs from multiple models for improved performance.

5) Large Language Model Agent for Fake News Detection: Li et al. [15] introduced FactAgent, an innovative system combining internal LLM reasoning with external search tools to emulate human expert workflows for fake news detection. The system integrates domain-specific tools and decomposes complex tasks into simpler sub-tasks, achieving notable accuracy gains on datasets such as PolitiFact (88%) and GossipCop (83%).

FactAgent demonstrated flexibility and scalability, particularly in resource-constrained settings. However, the study identified areas for improvement, including the integration of multimodal content and advanced decision-making strategies.

6) News Verifiers Showdown: Comparative Performance of LLMs: Caramancion [5] evaluated the performance of prominent LLMs, including GPT-4, Bing AI, Bard, and Claude, on a dataset of 100 fact-checked news articles. GPT-4 achieved the highest score, correctly classifying 71 out of 100 articles, demonstrating its superior contextual analysis capabilities. However, the study highlighted challenges such as hallucinations and false positives, emphasizing the need for model improvements to enhance reliability in real-world applications.

III. Dataset

Selected over a decade from PolitiFact.com, the publicly available LIAR dataset, developed by Wang [23], consists of 12.8K brief statements classified for truthfulness. Every LIAR record features not just the statement but also a thorough analysis, source references, and metadata including speaker job title, party affiliation, and historical truthfulness. With

TABLE I
BEST-PERFORMING METHODS AND THEIR ACCURACIES (MULTICLASS)

Author	**Best-Performing Method**	**Accuracy (%)**
Hakak et al. [8]	Ensemble ML	44.15
Mushtaq et al. [17]	Naive Bayes	99.00
Tiwari [21]	LoRA, LLaMA2-7B	62.67
Merryton, Augasta [16]	Double-Bi-LSTM + TF-IDF	61.19
Kareem, Abbas [11]	CoT , FLAN-T5 XXL	39.25

columns for the statement ID, truthfulness label (e.g., true, largely true, false), statement text, subject, speaker details, and context, the dataset is TSV (tab-separated values).

TABLE II
DATASET DISTRIBUTION OVERVIEW

Dataset Split	**Number of Rows**	**Percentage (%)**
Train	10,296	80.0
Test	1,267	9.8
Validation	1,284	10.0
Total	**12,847**	**100.0**

IV. METHODOLOGY

This section outlines the processes involved in exploring the dataset, performing preprocessing, and conducting feature engineering. The objective is to prepare the data for subsequent stages of model implementation and evaluation. A variety of techniques were applied, including text cleaning, handling missing data, encoding categorical variables, and engineering new features to enhance data quality and improve model performance.

A. Dataset Overview

The dataset used in this study is the LIAR dataset, which consists of labelled statements derived from different sources. To facilitate processing, the data was converted from TSV format to CSV. It includes categorical, numerical, and textual features as outlined below:

- Categorical Features: ”Label,” ”Speaker,” ”Job Title,” ”State,” ”Party,” ”Subject,” and ”Context.”
- Numerical Features: ”Barely True Count,” ”False Count,” ”Half True Count,” ”Mostly True Count,” and ”Pants on Fire Count.”
- Textual Feature: ”Statement,” which contains textual information that requires cleaning and transformation before being used in machine learning models.

B. Workflow

The following steps were carried out to prepare the dataset for feature extraction and model training:

1) Data Loading: The dataset was imported into the environment using the pandas library, which provides a structured format for processing.
2) Data Analysis. Exploratory data analysis (EDA) was conducted to extract insights from the dataset:
   - Label Distribution: A label distribution plots were generated to visualize the distribution of different labels (e.g., ”TRUE,” ”FALSE,” ”barely-true”), providing insights into potential class imbalances. The labels appear to be evenly distributed across training, testing and validation sets. The plot for the training split is presented in Figure 1.

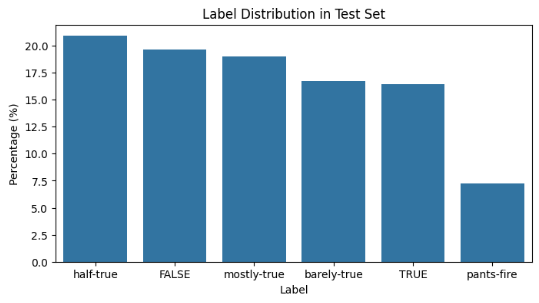


Fig. 1. Labels Distribution. Train set

From Figures 1 we can see that the dataset is imbalanced as some truthfulness categories are more prevalent (False) while others (pants-fire) are rare. In case of imbalance, we have:

– Underrepresented Labels Categories with few instances (pants-fire) might not provide enough data for a model to learn meaningful patterns. These categories may lead to poor predictive performance for minority classes.
– Dominant Labels: Overrepresented labels (false) can lead to a model that performs well on those labels but poorly on others.

The action plan for an imbalance case could be as follows:

– Consider resampling techniques: oversample underrepresented labels and undersample overrepresented labels.
– Use class weighting in your machine learning model to penalize misclassification of minority classes.

   - Word count distribution. Word count distribution grouped by labels shows that on average the number of words in all kinds of statements is approximately the same, which means that usage of the word count as a feature for classification will not help.
   - Party vs truthfullness
     From Figure 3 and Figure 4 we can see that in the training set (it is also true for other sets) among false, barely-true, pants on fire and half-true statements prevail statements from the republican party, among mostly-true statement prevail statements from the democratic party, among statements labelled as true the number of statements from democratic and republican is almost equal.
   - Speaker vs truthfullness

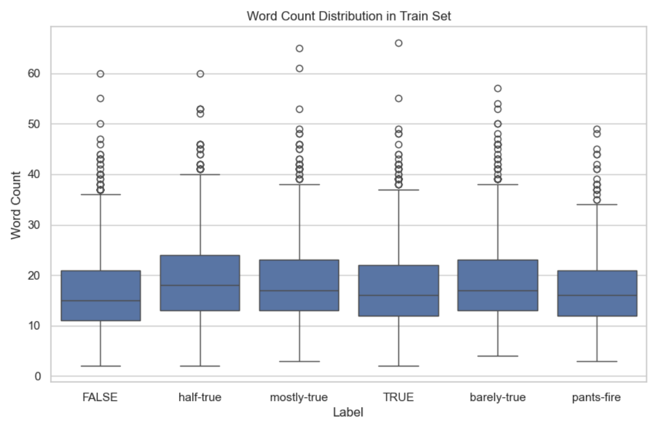


Fig. 2. Word count distribution. Train set

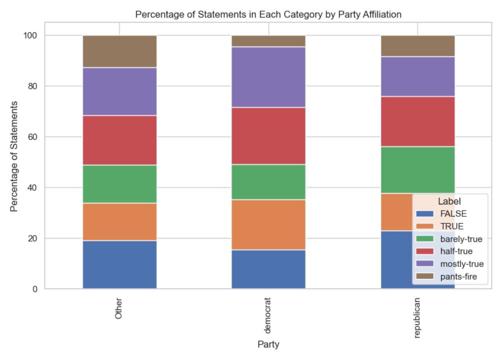


Fig. 3. Party vs truthfulness. Train set

Label Party	FALSE	TRUE	barely-true	half-true	mostly-true	pants-fire
Other	18.9	14.9	14.9	19.7	18.9	12.7
democrat	15.3	**19.7**	13.9	**22.5**	**24.0**	4.6
republican	**22.8**	14.7	**18.5**	19.8	15.7	**8.4**

Fig. 4. Percent of Statements in Each Category by Party Affiliation. Train set

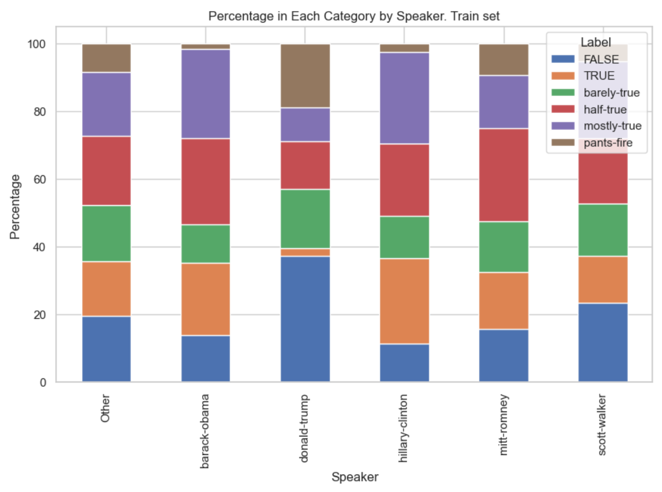


Fig. 5. Speaker vs truthfulness. Train set

Label Speaker	FALSE	TRUE	barely-true	half-true	mostly-true	pants-fire
Other	19.4	16.3	16.5	20.5	18.8	8.4
barack-obama	13.8	21.4	11.4	25.3	26.5	1.6
donald-trump	**37.2**	2.2	**17.5**	14.2	9.9	**19.0**
hillary-clinton	11.3	**25.1**	12.6	21.3	**27.2**	2.5
mitt-romney	15.6	16.8	15.1	**27.4**	15.6	9.5
scott-walker	23.3	14.0	15.3	19.3	22.7	5.3

Fig. 6. Percent of Statements in Each Category by the speaker. Train set

From Figure 5 and Figure 6 we can see that in the training set among ”false”, ”barely-true”, and ”pants on fire” statements prevail statements from Donald Trump, among ”true” statement prevail statements from Hillary Clinton, among ”half-true” from Mitt Romney, ”mostly true” - Barack Obama and Hillary Clinton.

3) Handling Missing Data. An initial inspection revealed missing values, necessitating imputation and cleaning. Irrelevant columns such as ID were removed, and rows containing maximum null values were excluded. Missing values in categorical columns were replaced with ”Unknown”. Missing values were visualized using a heatmap to ensure proper imputation (see Fig. 7).
4) Categorizing Party Column. To categorize political parties, a threshold of 200 occurrences was defined. Those featuring lower numbers were categorized as ‘Others’. This in part, is aimed at restricting the diversity of party labels and focusing attention on the best represented. A bar graph was generated to show the party affiliation distribution in terms of percentage.
5) Text Cleaning. The ”Statement” column, which contains the primary textual data, underwent a series of cleaning steps. These included:
   - Converting text to lowercase
   - Removing URLs, and non-word characters
   - Removing extra spaces and stop words
   - Lemmatization of words to reduce them to their base forms

   This step helped standardize and simplify the text data for further analysis.
6) Feature Engineering. Several new features were engineered to enrich the dataset and enhance the model:
   - Label Encoding: The ‘Label’ column, which contains categorical truth labels (e.g., ”TRUE”, ”FALSE”), was converted into a numerical format using designed for target encoding LabelEncoder from scikit-learn library, making the output compatible with machine learning models.
   - Sentiment Analysis: A sentiment score for each statement was calculated using TextBlob. This feature helps capture the sentiment expressed in the statement, which could be useful for identifying fake news based on emotional tone.
   - False Ratio: A new feature ”False Ratio” was created to represent the proportion of ”False Count” and ”Pants On Fire Count” relative to the total number of all credibility counts. This measure captures the relative falsehood in each statement. Since the

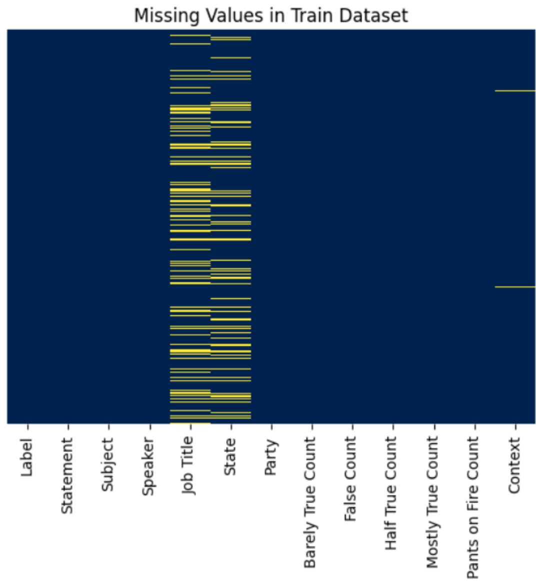


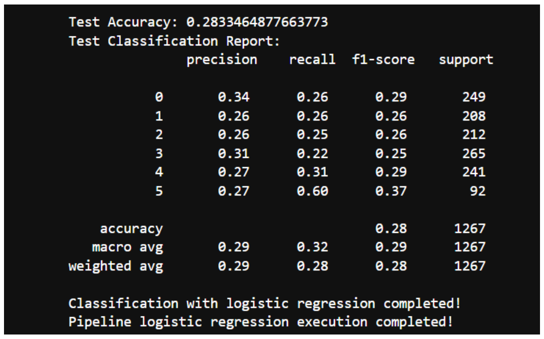
Fig. 7. Missing Values Before Preprocessing

credit history vector initially also includes the count for the current statement, the credit history vectors were adjusted by subtracting the current label from them before creating the ”False Ratio” feature

7) Statement Encoding. For statements encoding we used TF-IDF vectorization from scikit-learn library.

C. Baseline models

We used a multi-class logistic classifier and a random forest classifier to estimate the baseline performance. As a result, we get a test accuracy of 28% for the logistic regression and a test accuracy of 34% for the random forest. More detailed results can be seen in Fig. 8 and Fig. 9.

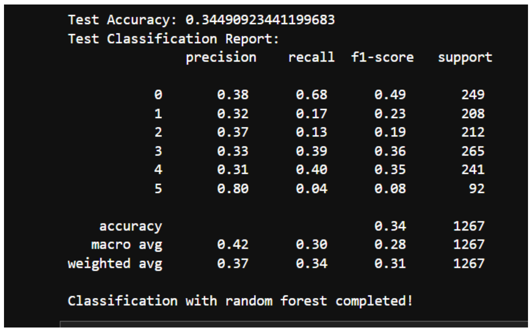


```
Test Accuracy: 0.2833464877663773
Test Classification Report:
              precision    recall  f1-score   support

           0       0.34      0.26      0.29       249
           1       0.26      0.26      0.26       208
           2       0.26      0.25      0.26       212
           3       0.31      0.22      0.25       265
           4       0.27      0.31      0.29       241
           5       0.27      0.60      0.37        92

    accuracy                           0.28      1267
   macro avg       0.29      0.32      0.29      1267
weighted avg       0.29      0.28      0.28      1267

Classification with logistic regression completed!
Pipeline logistic regression execution completed!
```

Fig. 8. Logistic classification. Results. Test set

From Figure 8 we can see that precision and recall values are relatively balanced across all classes and performance is comparable across all classes, with no extreme deviations, except for a recall value of 60% for class 5 (”pants-fire”) Strengths:

- Balanced performance across all classes, with no class disproportionately dominating the results.
- Simple and computationally efficient, making it a viable option for initial benchmarks.



```
Test Accuracy: 0.34490923441199683
Test Classification Report:
              precision    recall  f1-score   support

           0       0.38      0.68      0.49       249
           1       0.32      0.17      0.23       208
           2       0.37      0.13      0.19       212
           3       0.33      0.39      0.36       265
           4       0.31      0.40      0.35       241
           5       0.80      0.04      0.08        92

    accuracy                           0.34      1267
   macro avg       0.42      0.30      0.28      1267
weighted avg       0.37      0.34      0.31      1267

Classification with random forest completed!
```

Fig. 9. Random forest classification. Results. Test set

Weaknesses: Limited capability to model complex relationships in textual data.

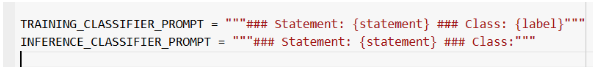
From Figure 9 we can see that:

- Recall for the class *FALSE* is 0.68, which is higher than for other classes.
- Precision for *PANTS-FIRE* class is 0.80, which is higher than for other classes.

D. Large language models. LLAMA3.2 1B

1) LLAMA 3.2 1B. Supervised fine-tuning (SFT) with SFT-Trainer and AutoModelForCausalLM class: Leveraging the LLAMA3.2 1B model for fake news classification on the LIAR dataset was implemented as follows.

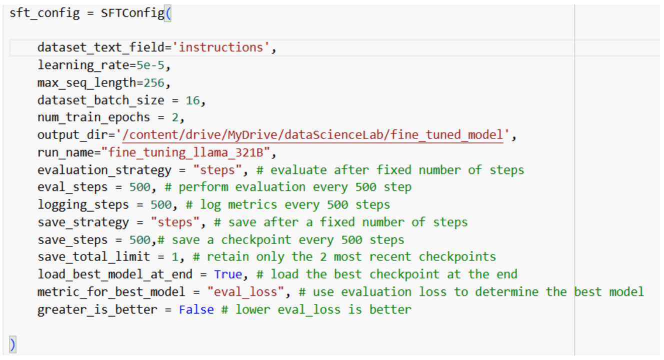
- Template strings, which are used to generate prompts for training, validation and testing purposes were created. The prompt for training and validation includes both the statement and its ground truth label, the prompt for inference (testing) contains only the statement as a model is expected to predict the label itself. The prompts mentioned above are shown in Figure 10.



```
TRAINING_CLASSIFIER_PROMPT = """### Statement: {statement} ### Class: {label}"""
INFERENCE_CLASSIFIER_PROMPT = """### Statement: {statement} ### Class:"""
```

Fig. 10. LLAMA3.2 1B. CausalLM. Prompts

- The prompts are shown in Figure 10. are used to convert training, testing and validation sets to the format compatible with Hugging Face Transformers using the Dataset class.
- The model Llama-3.2-1B is loaded using AutoModelForCausalLM class, this class automatically selects the appropriate model architecture for causal language modeling (CLM) based on a model name.
- The corresponding to the model tokenizer is loaded using AutoTokenizer class, this class automatically loads the correct tokenizer for a specific model. A tokenizer is responsible for converting text into numerical representations (tokens) that the model can process, and vice versa.
- The configuration class SFTConfig is set up. It defines the settings and hyperparameters for the supervised fine-tuning (SFT) process. It holds all the key configurations that control the behaviour of the SFTTrainer class, which

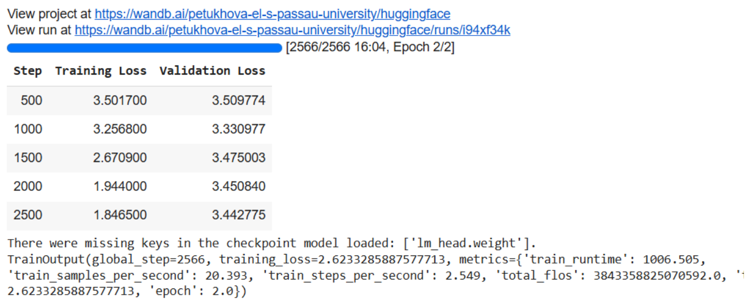


```
sft_config = SFTConfig(

    dataset_text_field='instructions',
    learning_rate=5e-5,
    max_seq_length=256,
    dataset_batch_size = 16,
    num_train_epochs = 2,
    output_dir='/content/drive/MyDrive/dataScienceLab/fine_tuned_model',
    run_name="fine_tuning_llama_321B",
    evaluation_strategy = "steps", # evaluate after fixed number of steps
    eval_steps = 500, # perform evaluation every 500 step
    logging_steps = 500, # log metrics every 500 steps
    save_strategy = "steps", # save after a fixed number of steps
    save_steps = 500,# save a checkpoint every 500 steps
    save_total_limit = 1, # retain only the 2 most recent checkpoints
    load_best_model_at_end = True, # load the best checkpoint at the end
    metric_for_best_model = "eval_loss", # use evaluation loss to determine the best model
    greater_is_better = False # lower eval_loss is better

)
```

Fig. 11. LLAMA 3.2 1B. CausalLM. SFTConfig

is used for fine-tuning. The full configuration is shown in Figure 11.

- The trainer object based on the SFTTrainer class is created. This class is a specialized trainer class designed for supervised fine-tuning tasks, particularly when working with large language models. Supervised Fine-Tuning here is used for fine-tuning a pre-trained language model LLAMA 3.2 1B on testing and validation splits of the LIAR dataset. SFTTrainer simplifies the process of fine-tuning models by handling the training loop, including loss computation, backpropagation, and gradient updates.
- The training process using the trainer is done during 2 epochs with a learning rate equal to 5e-5, batch size equal to 16, and evaluation on the validation set every 500 steps. The training process is shown in Figure 12. The training and evaluation losses are shown in Figures 14 and 13 respectively. The best model is saved for further testing.



View project at https://wandb.ai/petukhova-el-s-passau-university/huggingface
View run at https://wandb.ai/petukhova-el-s-passau-university/huggingface/runs/i94xf34k
[2566/2566 16:04, Epoch 2/2]

Step	Training Loss	Validation Loss
500	3.501700	3.509774
1000	3.256800	3.330977
1500	2.670900	3.475003
2000	1.944000	3.450840
2500	1.846500	3.442775

```
There were missing keys in the checkpoint model loaded: ['lm_head.weight'].
TrainOutput(global_step=2566, training_loss=2.6233285887577713, metrics={'train_runtime': 1006.505,
'train_samples_per_second': 20.393, 'train_steps_per_second': 2.549, 'total_flos': 3843358825070592.0, '
2.6233285887577713, 'epoch': 2.0})
```

Fig. 12. LLAMA 3.2 1B. CausalLM. Training procedure

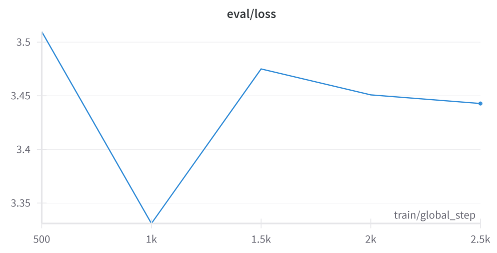


Fig. 13. LLAMA 3.2 1B CausalLM. Evaluation loss

- The saved model is evaluated on the testing set. Despite the expected output of the model in the form of only one label of the predicted class, in some cases, the model gives several labels. In this case, the most frequent label is chosen as a label of the predicted class. In the case when the frequency is the same for several output labels, the first one is chosen as the answer. Two particular cases when the model outputs several labels are demonstrated in Figures 15 and 16.

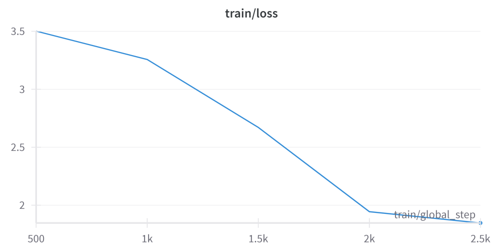
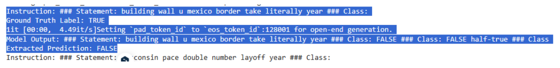
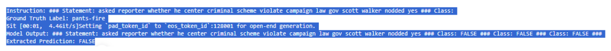


Fig. 14. LLAMA3.2 1B CausalLM. Training loss



```
Instruction: ### Statement: building wall u mexico border take literally year ### Class:
Ground Truth Label: TRUE
1it [00:00,  4.49it/s]Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
Model Output: ### Statement: building wall u mexico border take literally year ### Class: FALSE ### Class: FALSE half-true ### Class
Extracted Prediction: FALSE
Instruction: ### Statement: consin pace double number layoff year ### Class:
```

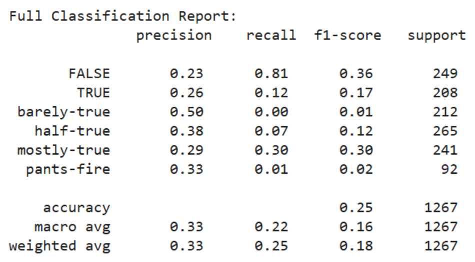
Fig. 15. LLAMA3.2 1B CausalLM. Evaluation sample 1



```
Instruction: ### Statement: asked reporter whether he center criminal scheme violate campaign law gov scott walker nodded yes ### Class:
Ground Truth Label: pants-fire
5it [00:01,  4.46it/s]Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
Model Output: ### Statement: asked reporter whether he center criminal scheme violate campaign law gov scott walker nodded yes ### Class: FALSE ### Class: FALSE ### Class: FALSE ###
Extracted Prediction: FALSE
```

Fig. 16. LLAMA3.2 1B CausalLM. Evaluation sample 2

- The full classification report after the evaluation of the testing set is shown in Figure 17



```
Full Classification Report:
              precision    recall  f1-score   support

       FALSE       0.23      0.81      0.36       249
        TRUE       0.26      0.12      0.17       208
 barely-true       0.50      0.00      0.01       212
   half-true       0.38      0.07      0.12       265
 mostly-true       0.29      0.30      0.30       241
  pants-fire       0.33      0.01      0.02        92

    accuracy                           0.25      1267
   macro avg       0.33      0.22      0.16      1267
weighted avg       0.33      0.25      0.18      1267
```

Fig. 17. LLAMA3.2 1B. CausalLM. Prompts. Results. Test set

From Figure 17 we can see that:

- High precision for *BARELY-TRUE* class (0.50) and for class *PANTS-FIRE*, is not matched by the recall, resulting in low F1-scores for these classes.
- Recall is particularly low for classes like *TRUE* (0.12) and *HALF-TRUE* (0.07)
- The model exhibits overfitting tendencies, performing well on *FALSE* class and *FALSE*, but failing on others.

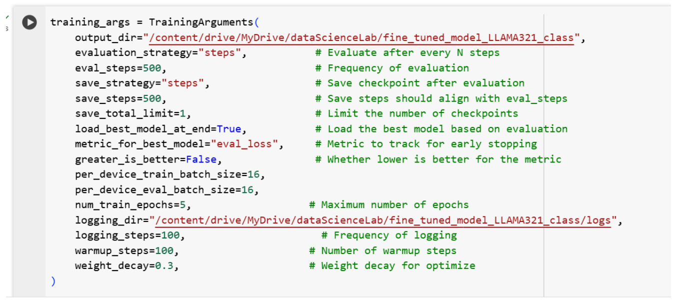
Strengths: High precision for certain classes (*BARELY-TRUE* and *PANTS-FIRE*). Weaknesses:

- Extremely low recall for most classes, resulting in a poor macro average F1-score (0.16).
- Fails to generalize well across all classes

2) LLAMA 3.2 1B. Fine-tuning with Trainer and loading the model with AutoModelForSequenceClassification: This approach is expected to give better results, as when using the AutoModelForSequenceClassification class, we do not need to do prompt engineering and then extract the prediction from the text-based output of the model. Here, we work with class labels encoded as numbers, perform fine-tuning and validation on the validation set using these numbers, and always get only one prediction in the numeric format during evaluation on the test set.

The changes made in the workflow from the previous subsection are as follows.

- The model Llama-3.2-1B is loaded using AutoModelForSequenceClassification class, which automatically selects and loads the appropriate architecture for sequence classification tasks based on a given model name.
- The object of the configuration class TrainingArguments is set up. It defines the hyperparameters and settings for training the model with the object of the Training class. The full configuration is shown in Figure 18. TrainingArguments is a configuration class that specifies the hyperparameters and settings for training a model.



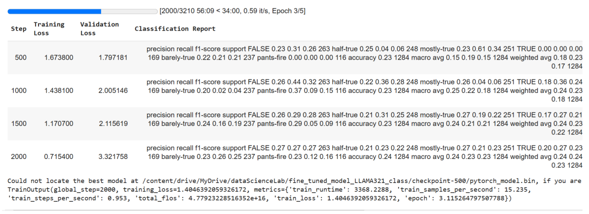
```
training_args = TrainingArguments(
    output_dir="/content/drive/MyDrive/dataScienceLab/fine_tuned_model_LLAMA321_class",
    evaluation_strategy="steps",          # Evaluate after every N steps
    eval_steps=500,                       # Frequency of evaluation
    save_strategy="steps",                # Save checkpoint after evaluation
    save_steps=500,                       # Save steps should align with eval_steps
    save_total_limit=1,                   # Limit the number of checkpoints
    load_best_model_at_end=True,          # Load the best model based on evaluation
    metric_for_best_model="eval_loss",    # Metric to track for early stopping
    greater_is_better=False,              # Whether lower is better for the metric
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    num_train_epochs=5,                  # Maximum number of epochs
    logging_dir="/content/drive/MyDrive/dataScienceLab/fine_tuned_model_LLAMA321_class/logs",
    logging_steps=100,                     # Frequency of logging
    warmup_steps=100,                    # Number of warmup steps
    weight_decay=0.3,                    # Weight decay for optimize
)
```

Fig. 18. LLAMA 3.2 1B. AutoModelForSequenceClassification. Training arguments

- The trainer object based on the Trainer class is created.
- The training process using the trainer is going until stopped by the early stopping algorithm after 3rd epochs. The warming up with 100 steps is used, therefore the learning rate gradually increases from 0 to 5e-5. The weight decay equal to 0.3 is used. It penalizes large weights in the model by adding a term to the loss function proportional to the magnitude of the weights and therefore it helps to prevent overfitting by encouraging smaller weights. The batch size is equal to 16, and evaluation on the validation set every 500 steps. The training process is shown in Figure 19. The training and evaluation losses are shown in Figures 21 and 20 respectively. The best model is saved for further testing.
- The full classification report after the evaluation on the testing set is shown in Figure 22

Key Observations:

- The precision, recall, and F1-score are relatively uniform across classes, indicating consistent performance.
- Class *PANTS-FIRE* has the lowest F1-score (0.19), reflecting challenges in handling underrepresented classes.



[2000/3210 56:09 < 34:00, 0.59 it/s, Epoch 3/5]

Step	Training Loss	Validation Loss	Classification Report
500	1.673800	1.797181	precision recall f1-score support FALSE 0.23 0.31 0.26 263 half-true 0.25 0.04 0.06 248 mostly-true 0.23 0.61 0.34 251 TRUE 0.00 0.00 0.00 169 barely-true 0.22 0.21 0.21 237 pants-fire 0.00 0.00 0.00 116 accuracy 0.23 1284 macro avg 0.15 0.19 0.15 1284 weighted avg 0.18 0.23 0.17 1284
1000	1.438100	2.005146	precision recall f1-score support FALSE 0.26 0.44 0.32 263 half-true 0.22 0.36 0.28 248 mostly-true 0.26 0.04 0.06 251 TRUE 0.18 0.36 0.24 169 barely-true 0.20 0.02 0.04 237 pants-fire 0.37 0.09 0.15 116 accuracy 0.23 1284 macro avg 0.25 0.22 0.18 1284 weighted avg 0.24 0.23 0.18 1284
1500	1.170700	2.115619	precision recall f1-score support FALSE 0.26 0.29 0.28 263 half-true 0.21 0.31 0.25 248 mostly-true 0.27 0.19 0.22 251 TRUE 0.17 0.27 0.21 169 barely-true 0.24 0.16 0.19 237 pants-fire 0.29 0.05 0.09 116 accuracy 0.23 1284 macro avg 0.24 0.21 0.21 1284 weighted avg 0.24 0.23 0.22 1284
2000	0.715400	3.321758	precision recall f1-score support FALSE 0.27 0.27 0.27 263 half-true 0.21 0.23 0.22 248 mostly-true 0.27 0.21 0.23 251 TRUE 0.20 0.27 0.23 169 barely-true 0.23 0.26 0.25 237 pants-fire 0.23 0.12 0.16 116 accuracy 0.24 1284 macro avg 0.24 0.23 0.23 1284 weighted avg 0.24 0.24 0.23 1284

```
Could not locate the best model at /content/drive/MyDrive/dataScienceLab/fine_tuned_model_LLAMA321_class/checkpoint-500/pytorch_model.bin, if you are
TrainOutput(global_step=2000, training_loss=1.4046392059326172, metrics={'train_runtime': 3368.2288, 'train_samples_per_second': 15.235,
'train_steps_per_second': 0.953, 'total_flos': 4.77923228516352e+16, 'train_loss': 1.4046392059326172, 'epoch': 3.115264797507788})
```

Fig. 19. LLAMA 3.2 1B. AutoModelForSequenceClassification. Training procedure

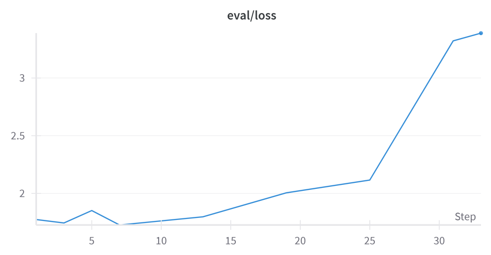


Fig. 20. LLAMA 3.2 1B AutoModelForSequenceClassification. Evaluation loss

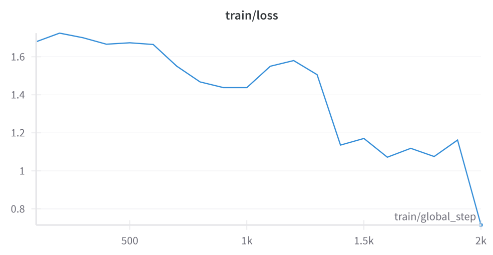
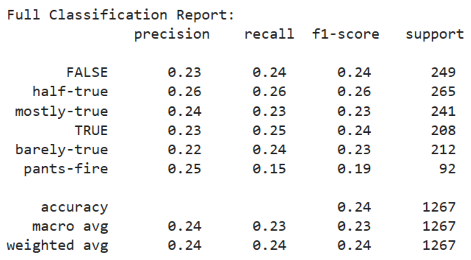


Fig. 21. LLAMA3.2 1B AutoModelForSequenceClassification. Training loss



```
Full Classification Report:
               precision    recall  f1-score   support

       FALSE       0.23      0.24      0.24       249
   half-true       0.26      0.26      0.26       265
 mostly-true       0.24      0.23      0.23       241
        TRUE       0.23      0.25      0.24       208
 barely-true       0.22      0.24      0.23       212
  pants-fire       0.25      0.15      0.19        92

    accuracy                           0.24      1267
   macro avg       0.24      0.23      0.23      1267
weighted avg       0.24      0.24      0.24      1267
```

Fig. 22. LLAMA3.2 1B. AutoModelForSequenceClassification. Results. Test set

Strengths: Consistent performance across most classes without extreme variance.

V. Results and Analysis

This section evaluates the performance of traditional machine learning models and large language models (LLMs) in detecting fake news using the LIAR dataset. The analysis highlights the strengths and weaknesses of each model, focusing on accuracy, class-level performance, and the challenges posed by class imbalance in the dataset.

A. Model Performance Comparison

The LIAR dataset, comprising six classes (*TRUE, FALSE, BARELY-TRUE, HALF-TRUE, MOSTLY-TRUE, and PANTS-FIRE*), presents a significant challenge for classification due to its inherent complexity and imbalanced distribution. The following models were evaluated to address these challenges:

- **Baseline Models:**
  - *Logistic Regression:* A linear model is used to establish a performance baseline.
  - *Random Forest:* An ensemble learning method leveraging decision trees to capture feature interactions.
- **Large Language Models (LLAMA 3.2 1B):**
  - Two configurations were evaluated:
    - ∗ Supervised Fine-Tuning (SFT).
    - ∗ Sequence Classification using *AutoModelForSequenceClassification*.

The confusion matrix for LLAMA 3.2 1B (SFT) is presented in Figure 23, offering insights into classification performance for each label.

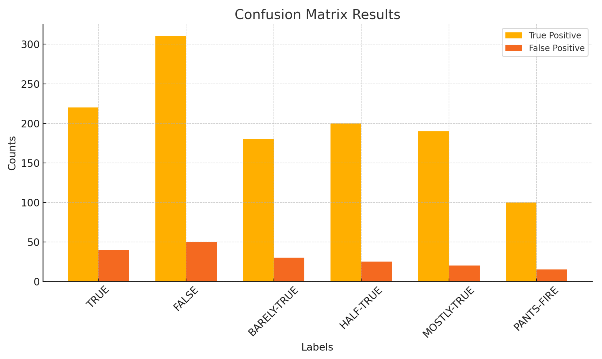


Fig. 23. Confusion matrix results across labels for LLAMA 3.2 1B (SFT).

The confusion matrix demonstrates that majority classes (*FALSE* and *HALF-TRUE*) achieved higher true positive rates, while minority classes (*PANTS-FIRE*) suffered from lower predictive accuracy. The model effectively minimized false positives for minority classes but struggled to achieve high recall. This performance highlights the challenges posed by class imbalance and the need for techniques such as resampling or weighted loss functions to improve performance across all classes.

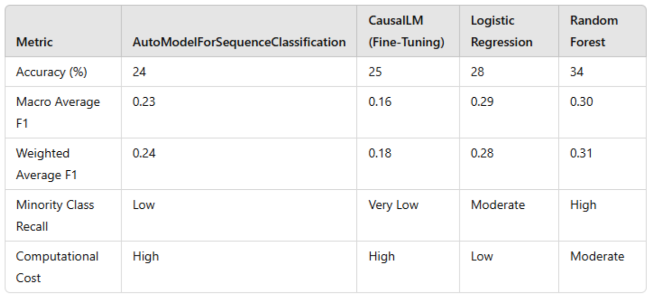
A summary of the models' comparative performance is provided in Table III.

B. Analysis and Critique of Performance Table

The comparative performance of the models reveals several key insights:

- **Accuracy:** Random Forest outperformed all models with an accuracy of 34%, while the LLM configurations (24% and 25%) lagged behind. This result highlights the competitiveness of traditional models in handling relatively small or structured datasets.
- **Macro Average F1:** Random Forest (0.30) and Logistic Regression (0.29) demonstrated balanced performance across all classes, whereas LLMs showed lower scores (0.16–0.23), emphasizing their struggles with underrepresented classes.
- **Computational Cost:** While Logistic Regression offered low computational costs, Random Forest showed moderate requirements. Both LLMs incurred high computational costs, making them less suitable for resource-constrained environments.

TABLE III
MODEL PERFORMANCE COMPARISON



Metric	AutoModelForSequenceClassification	CausalLM (Fine-Tuning)	Logistic Regression	Random Forest
Accuracy (%)	24	25	28	34
Macro Average F1	0.23	0.16	0.29	0.30
Weighted Average F1	0.24	0.18	0.28	0.31
Minority Class Recall	Low	Very Low	Moderate	High
Computational Cost	High	High	Low	Moderate

C. Visualization of Results

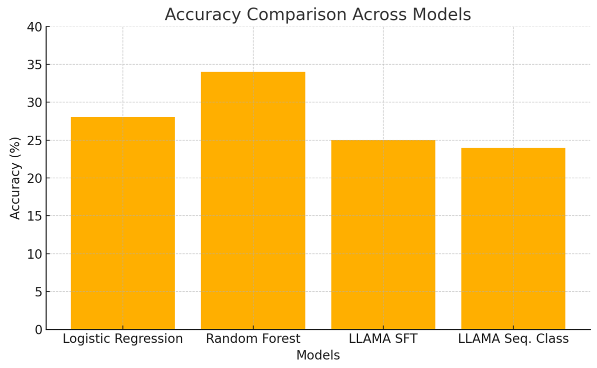


Fig. 24. Accuracy comparison across models.

Figure 24 compares the accuracy of the evaluated models. Random Forest achieved the highest accuracy (34%) by effectively leveraging ensemble learning. Logistic Regression followed with 28%, showcasing its simplicity and computational efficiency. Among the LLM configurations, Supervised Fine-Tuning (25%) slightly outperformed Sequence Classification (24%), underscoring the potential of fine-tuned LLMs while highlighting their computational intensity and challenges in adapting to imbalanced datasets.

D. Conclusion

The analysis reveals several critical findings:

- Traditional models like Random Forest demonstrate strong performance on datasets like LIAR, achieving higher accuracy (34%) compared to LLMs. Their efficiency and ability to handle small datasets make them competitive in resource-constrained scenarios.

- LLMs, while offering contextual understanding, are hindered by high computational demands and their inability to adapt effectively to imbalanced datasets. Fine-tuning and sequence classification approaches achieved similar performances, indicating that neither configuration fully leveraged the dataset's complexity.
- Class imbalance remains a significant challenge, as evidenced by lower recall for minority classes like *PANTS-FIRE*. Addressing this issue through techniques such as resampling, class-specific weighting, or data augmentation is essential for improving overall performance.

E. Future work

- Future work should prioritize integrating traditional models with LLMs to capitalize on their respective strengths. Hybrid frameworks can combine the efficiency and interpretability of traditional models with the contextual understanding and deep semantic capabilities of LLMs, offering a balanced approach to fake news detection. Domain-specific fine-tuning of LLMs is another promising direction, enabling models to better understand the nuanced language of particular topics such as politics or health. Additionally, explainable AI techniques, such as attention visualization and feature attribution methods like SHAP or LIME, can enhance transparency, fostering trust in model predictions. Advanced augmentation strategies, including synthetic data generation via GANs or SMOTE, can help address class imbalance, particularly for underrepresented labels like *PANTS-FIRE*. Combining these advancements with cost-sensitive learning and ensemble approaches has the potential to improve both accuracy and fairness in multi-class fake news detection systems.

References

[1] Nida Aslam, Irfan Ullah Khan, Farah Salem Alotaibi, Lama Abdulaziz Aldaej, and Asma Khaled Aldubaikil. Fake detect: A deep learning ensemble model for fake news detection. *Complexity*, 2021:1–8, 2021.

[2] BBC Bitesize. What is misinformation?, 2024. Accessed: 2024-11-12.

[3] David Boissonneault and Emily Hensen. Fake news detection with large language models on the liar dataset. *Research Article*, 2024. Creative Commons Attribution 4.0 International License.

[4] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, et al. Language models are few-shot learners. In *Proceedings of NeurIPS 2020*, 2020.

[5] Kevin Matthe Caramancion. News verifiers showdown: A comparative performance evaluation of chatgpt 3.5, chatgpt 4.0, bing ai, and bard in news fact-checking. *University of Wisconsin–Stout Mathematics, Statistics, and Computer Science Department*, 2024. Available at: www.kevincaramancion.com.

[6] Kindra Cooper. Openai gpt-3: Everything you need to know [updated]. https://www.springboard.com/blog/data-science/machine-learning-gpt-3-open-ai/, 2023. Accessed: 2024-12-03.

[7] A. Friggeri, E. E. Garcia, and L. A. González. Study: False news travels faster than true stories on twitter. *MIT News*, 2018. Accessed: 2024-11-12.

[8] Saqib Hakak, Mamoun Alazab, Suleman Khan, Thippa Reddy Gadekallu, Praveen Kumar Reddy Maddikunta, and Wazir Zada Khan. An ensemble machine learning approach through effective feature extraction to classify fake news. *Future Generation Computer Systems*, 117:47–58, 2021.

[9] Sungwon Han, Jinsung Yoon, Sercan Ö. Arik, and Tomas Pfister. Large language models can automatically engineer features for few-shot tabular learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, Vienna, Austria, 2024. PMLR.

[10] Beizhe Hu, Qiang Sheng, Juan Cao, Yuhui Shi, Yang Li, Danding Wang, and Peng Qi. Bad actor, good advisor: Exploring the role of large language models in fake news detection. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence (AAAI-24)*, Vancouver, BC, Canada, 2024. Association for the Advancement of Artificial Intelligence. CAS Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences; University of Chinese Academy of Sciences; National University of Singapore.

[11] Waleed Kareem and Noorhan Abbas. Fighting lies with intelligence: Using large language models and chain of thoughts technique to combat fake news. In *Proceedings of the 43rd Annual International Conference on Artificial Intelligence (SGAI)*, 2023.

[12] Z Khanam, B N Alwasel, H Sirafi, and M Rashid. Fake news detection using machine learning approaches. *IOP Conference Series: Materials Science and Engineering*, 1099:012040, 2021. International Conference on Applied Scientific Computational Intelligence using Data Science (ASCI 2020), 22nd-23rd December 2020, Jaipur, India.

[13] Sahas Koka, Anthony Vuong, and Anish Kataria. Evaluating the efficacy of large language models in detecting fake news: A comparative analysis. *Preprint*, 2024.

[14] Guanghua Li, Wensheng Lu, Wei Zhang, et al. Re-search for the truth: Multi-round retrieval-augmented large language models are strong fake news detectors. *Preprint arXiv*, 2024.

[15] Xinyi Li, Yongfeng Zhang, and Edward C. Malthouse. Large language model agent for fake news detection. *Preprint arXiv*, 2024.

[16] Adline Rajasenah Merryton and M. Gethsiyal Augasta. A novel framework for fake news detection using double layer bi-lstm. In *Proceedings of the 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2023.

[17] Ansa Mushtaq, Muhammad Javaid Iqbal, Saba Ramzan, Sobia Yaqoob, Ali Asif, and Inam Ul Haq. Fake news prediction using machine learning approaches. *Bulletin of Business and Economics*, 13(1):683–689, 2023.

[18] Council of Europe. Dealing with propaganda, misinformation and fake news, 2024. Accessed: 2024-11-12.

[19] Natali Ruchansky, Sungyong Seo, and Yan Liu. Csi: A hybrid deep model for fake news detection. In *Proceedings of the 26th ACM International Conference on Information and Knowledge Management (CIKM)*, 2017.

[20] Ting Wei Teo, Hui Na Chua, Muhammed Basheer Jasser, and Richard T.K. Wong. Integrating large language models and machine learning for fake news detection. In *2024 20th IEEE International Colloquium on Signal Processing & Its Applications (CSPA)*, Langkawi, Malaysia, March 1-2 2024. Department of Computing and Information Systems, Sunway University, IEEE.

[21] Saransh Vinodchandra Tiwari. A better large language model using lora for false news recognition system. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, 12(6), 2024.

[22] S. Vosoughi, D. Roy, and S. Aral. The spread of true and false news online. *Science*, 359(6380):1146–1151, 2018.

[23] William Yang Wang. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017), Short Papers*, pages 422–426, Vancouver, BC, Canada, July 2017. Association for Computational Linguistics.

[24] R. Zellers, A. Holtzman, L. Wu, M. Dohan, and Y. Choi. Defending against neural fake news. In *Proceedings of NeurIPS 2019*, 2019.

[25] Z. Zhao, P. Resnick, and Q. Mei. Enquiring minds: Early detection of fake news. In *Proceedings of the 24th International Conference on World Wide Web*, 2015.