Fake News Detection Using Language Models

Authors: Baris Yazici, Vishal Kumar
ST 311: Artificial Intelligence - Final Project
London School of Economics

Abstract

Fake news detection has become a challenge with the rise of misinformation on social media, politics, and biased news reporting. Various approaches involving machine learning and deep learning have been proposed to detect fake news, but these methods have mostly been limited to specific domains and the patterns in the text on which the models are trained. We find that Large Language Models, with their ability to reason about the context and facts involved in news statements, achieve the best performance in fake news detection, surpassing Small Language Models, as well as traditional machine learning and deep learning methods. We also compare various fine-tuning methods and discuss their trade offs in terms of accuracy and efficiency.

1 Introduction

With the rise of online news sources and use of social media platforms for political communication, sharing and acting upon information without fact-checking has become a widespread phenomenon. This study aims to provide an approach to tackle the problem of fake news and deceptive political statements using transformer based language models. Previous studies on the topic of fake news detection have focused on traditional machine learning and early deep learning models, such as support vector machine (SVM), k-nearest neighbours (KNN), Naive Bayes, convolutional neural network (CNN), and recurrent neural network (RNN), as well as early small language models (SLM) like BERT, RoBERTa, and DistilBERT. However, no comprehensive benchmarking study on fake news detection was done using large language models (LLM). We aim to improve the benchmarks provided for SLMs by previous studies and provide novel benchmarks for LLMs in fake news detection. In doing so, we will also be exploring the practicality of using LLMs and whether daily training and use of LLMs on local hardware is possible.

2 Methodology

As outlined in Figure 1, we propose two main approaches to detect fake news:

1. Utilising transformer-based Small Language Models (SLMs) to classify news as either fake or not.
2. Leveraging Large Language Models (LLMs) to identify potentially misleading and fake tweets/news/statements and generate informed responses.

We evaluate various combinations of model architectures and report which models work better together and are suitable for our task. We also explore various fine tuning techniques, such as Supervised Fine-Tuning (SFT) on SLMs and Parameter-Efficient Fine-Tuning (PEFT) on LLMs, and report their relative performance on fake news detection. The goal is to develop the most efficient and reliable approach in detecting fake news by leveraging the strengths of SLMs and LLMs.

Preprint. Under review.

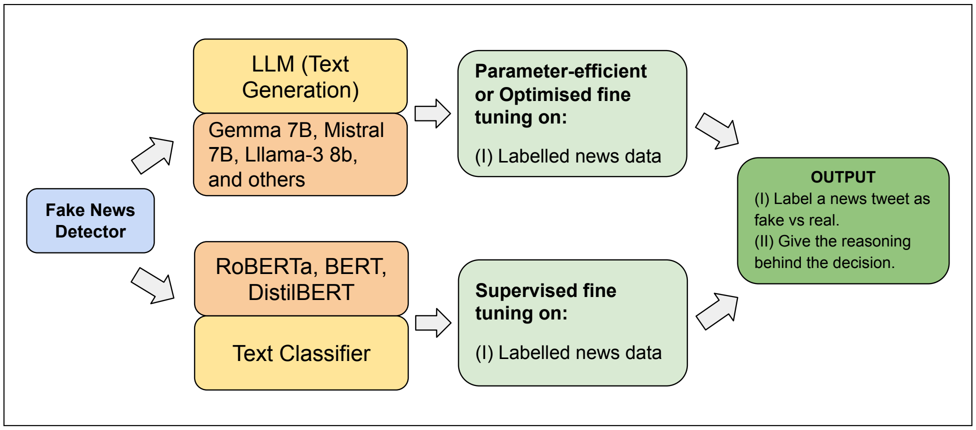
Fake News Detector
LLM (Text Generation)
Gemma 7B, Mistral 7B, Lllama-3 8b, and others
Parameter-efficient or Optimised fine tuning on:
(I) Labelled news data
RoBERTa, BERT, DistilBERT
Text Classifier
Supervised fine tuning on:
(I) Labelled news data
OUTPUT
(I) Label a news tweet as fake vs real.
(II) Give the reasoning behind the decision.

Figure 1: Fake News Detection Model Pipeline

2.1 Models and Fine-Tuning Specifics: Small Language Models (SLM)

To tackle the challenge of fake news, our approach incorporates SLMs that are based on the transformer architecture, renowned for its revolutionary impact on natural language processing tasks since its inception in 2017. These models have been at the forefront of AI advancements, particularly excelling in speed and efficiency for text classification. Within our framework, we leverage the following transformer-based SLMs:

1. BERT [1]: A cornerstone model in the transformer family, BERT captures deep bidirectional context, setting a new standard for understanding nuances in language.
2. DistilBERT [6]: This model streamlines the original BERT architecture, preserving a substantial amount of its language understanding capabilities while increasing operational efficiency.
3. RoBERTa [5]: Building upon BERT, RoBERTa fine-tunes the pre-training procedure, leading to improved performance on a range of benchmark tasks.

These transformer-based SLMs are chosen for their swift and effective handling of language tasks, making them exceptionally well-suited for the dynamic and high-volume nature of social media content. Their design facilitates extensive fine-tuning, enabling these models to keep pace with the evolving patterns of misinformation. BERT was trained on a large corpus of text with a masked language modelling objective, which helps it understand context in both directions. For our task, BERT was fine-tuned on the Liar dataset, allowing it to discern nuances in language that indicate misinformation. DistilBERT was developed as a streamlined version of BERT, with fewer parameters but designed to retain most of its predictive power. The fine-tuning of DistilBERT followed the same procedure as BERT, allowing for quicker training and inference times without significant loss of accuracy. RoBERTa iterates upon BERT with changes to key hyperparameters and training with more data, longer, and on bigger batches, resulting in improved performance. Its fine-tuning on the detection task further hones its ability to classify text accurately.

2.2 Models and Fine-Tuning Specifics: Large Language Models (LLM)

The use of LLMs in the form of chatbots through web interfaces, such as ChatGPT and Claude, have become increasingly common, but integrating these models to pipelines for personalised tasks are still not mainstream since use of APIs are not cheap, fine-tuning is computationally expensive, and manual inference using code requires a more complex understanding of the model architecture. Nowadays, the state of the art models have billions of parameters, with the recently released Llama 3 having 8 billion and 70 billion versions (see Introducing Meta Llama 3) and Mixtral 8x22b having a total of 141 billion parameters, of which 39 billion are active (see Mistral Models). Even though both of these models are openly available on Hugging Face for fine-tuning and inference, loading these models require 10s of gigabytes RAM, which is usually not possible with a regular laptop.

Our goal for the LLM section of this study is to explore various approaches for using state of the art text generation models on local hardware to complete personalised and specialised tasks, such as fake news detection. One could imagine an open source LLM being used as a browser extension to the social media platform X to analyse if a tweet by a news site or a politician has a fake or a misleading content or not. This requires careful consideration on what the size of the model is, whether the model needs fine-tuning, and which fine-tuning approach is appropriate given computational limitations.

See the table below for the large language models we used, with their respective parameter sizes and developers. Note that we use the exact names these models are hosted on Hugging Face.

Table 1: Large Language Models

Model Name	**Developed by**	**Parameter Size**
Meta-Llama-3-8B	Meta AI	8 billion
gemma-7b	Google	7 billion
Mistral-7B-v0.1	Mistral AI	7 billion
zephyr-7b-beta	Hugging Face	7 billion
stablelm-2-zephyr-1.6b	Stability AI	1.6 billion
TinyLlama-1.1B-Chat-v1.0	TinyLlama	1.1 billion
bloomz-1b1	BigScience	1.1 billion
bloomz-560m	BigScience	560 million

2.2.1 Parameter-Efficient Fine-Tuning with Prompt Tuning Method

Parameter-efficient fine-tuning (see Hugging Face PEFT library for implementation) is a tuning paradigm that aims to fine tune models by freezing original weights of the model and adding a new layer of trainable parameters, which are very small in size compared to the total number of parameters in the model. This enables low computational cost and memory during training. Where and how this trainable layer is added is what varies in each PEFT method—popular ones being prompt-based, LoRA, and IA3 methods. In our fake news detection task, we use parameter efficient prompt-tuning [4]. Prompt tuning originates from the prompting paradigm, which is done by adding "a series of tokens, P, to the input X, such that the model maximizes the likelihood of the correct Y, $\Pr_\theta(Y|[P;X])$, while keeping the model parameters, θ, fixed" [4]. What makes prompt tuning different from regular prompting is that it doesn't assume that "P [is] parameterized by θ; instead the prompt has its own dedicated parameters, θ_P", that is learned through backpropagation [4]. For example, stablelm-2-zephyr-1.6b, one of the models we work with, has $16,384$ trainable parameters out of $1,644,531,712$ total parameters when prompt tuning is used.

Lester et al. (2021) shows that prompt tuning matches the performance of regular supervised fine tuning, as the parameter number of the original model increases to 10b parameters. However, a downside of PEFT is that it requires loading the whole model, even though it is not updating most of them and thus is faster, which is possible for smaller models (in the range of 500m to 1.7b parameters) but not possible for larger models with 3b+ parameters. Thus, we decided to compare PEFT with another approach: using supervised fine tuning on a quantized and optimised version of a model larger in size (in the range of 1.7b to 8b).

2.2.2 Supervised Fine-Tuning with Quantized and Optimised Models

We use Unsloth to load the 4 bit pre-quantized and custom kernel versions of base models, which requires less memory, while allowing faster training and inference (please check Unsloth AI for an example of a manual autograd written and a short explanation of how they manually optimise differentiation for various layers of models). Unsloth is fully integrated to Hugging Face's SFT repository and can be used to accelerate training and inference for Llama and Mistral architectures. For supervised fine tuning, we use Hugging Face's supervised fine tuning trainer. This approach allows us to update all of the trainable parameters of the model, which yields competitive performance, without bearing the cost and speed issues of loading the unquantized regular model. Specifically, we were able to load and train $7-8$b parameter models on 1 epoch in around 2000 seconds using a single T4 GPU.

3 Study with Real Data

3.1 Dataset

The lack of labelled data is a major challenge for fake-news detection. For our analysis, we used the widely-used benchmark dataset Liar, which contains labelled political statements from PolitiFact [7]. Among the studies conducted on fake news detection, Liar has been the dataset that models consistently perform worse than they do on other datasets (see, for example, Table 5 in Khan et al. 2021 [3]).

Even though it is one of the largest datasets on fake news, Liar has two major limitations:

1. It contains a training set of $10,300$ entries and a test set of $1,280$, which is a relatively small amount of data to train a generalizable deep neural network.
2. The statements included are from U.S. politics, limiting generalisation to other domains or regions.

Although the original dataset has a six-grade truthfulness scale, we analyse the language used in true (labelled as 3) and false statements (labelled as 0). This simplification is aligned with our goal to provide a clear binary response to potentially misleading information and one that is made in other studies as well. To access the dataset through Hugging Face, please use the following link: `https://huggingface.co/datasets/liar`.

3.2 Metrics

We report models' accuracy, precision, recall, and F1 score on the test set, as well as training and inference times. To make it comparable with other studies done on Liar and Fake News Detection, we use the same definitions as in Khan et al. (2021).

Real news are called the 'positive class', whereas fake news are the 'negative class'. Accordingly, True Positive (TP) is when a real news is classified as real, while True Negative (TN) is when a fake news is classified as fake. False Positive (FP) means a fake news is classified as real and False Negative (FN) means a real news is classified as fake. Then, according to Khan et al. (2021),

$$P(R) = \frac{TP}{TP + FP}, \; P(F) = \frac{TN}{TN + FN}, \; P = \frac{P(R) + P(F)}{2} \tag{1}$$

where $P(R)$ is the precision for real news and $P(F)$ is the precision for fake news.

Similarly,

$$R(R) = \frac{TP}{TP + FN}, \; R(F) = \frac{TN}{TN + FP}, \; R = \frac{R(R) + R(F)}{2} \tag{2}$$

where $R(R)$ is the recall for real news and $R(F)$ is the recall for fake news.

Finally, F1 score is defined as,

$$F1 = \frac{2 \cdot P \cdot R}{P + R} \tag{3}$$

3.3 Results From Other Studies

As mentioned in 3.1, there has been benchmarking studies done on Liar. Below is the results obtained by Khan et al. (2021), which suggest that RoBERTa performs the best with the Liar dataset [3].

A key difference between our approach and the benchmark study paper lies in the fine-tuning methodology. In the benchmark paper, the authors used pre-trained weights and only fine-tuned the final classification layers of the language models. However, in our project, we performed supervised fine-tuning on all layers, allowing the models to adapt their weights and internal representations to the specific deception detection task on the Liar dataset

Table 2: Performance of Advanced Pre-trained Language Models on the Liar Dataset [3]

Model	Accuracy	Precision	Recall	F1 Score
BERT	.62	.62	.62	.62
RoBERTa	**.62**	**.63**	**.62**	**.62**
DistilBERT	.60	.60	.60	.60
ELECTRA	.61	.61	.61	.61
ELMo	.61	.61	.61	.61

3.4 Our Results

We compared models that have different model types, architectures, and tuning methods on several metrics. We present our results in the table below (bold is the best in each category within model type).

Table 3: Test Performance of Different Models on Liar

Model Type	Model Name	Tuning Method	Metrics				Epochs	Train (s)	Avg Inference (s)
			Accuracy	Precision	Recall	F1 Score			
SLM	bert	base model	0.542	0.271	0.500	0.352	NA	NA	0.0056
		SFT	0.668	0.656	0.648	0.652	5	299	0.0027
	distilbert-base-cased	base model	0.458	0.229	0.500	0.314	NA	NA	0.0040
		SFT	0.642	0.643	0.642	0.642	5	**168**	**0.0016**
	roberta-base	base model	0.458	0.229	0.500	0.314	NA	NA	0.0041
		SFT	**0.677**	**0.657**	**0.658**	**0.658**	5	308	0.0026
LLM	llama-3-8b-bnb-4bit	base model	0.605	0.638	0.577	0.606	NA	NA	3.7896
		SFT	0.699	0.701	0.702	0.701	1	2198	8.6009
	gemma-7b-bnb-4bit	base model	0.586	0.598	0.559	0.578	NA	NA	4.9870
		SFT	**0.720**	**0.738**	**0.730**	**0.734**	1	2483	0.7158
	mistral-7b-bnb-4bit	base model	0.549	0.540	0.516	0.528	NA	NA	7.9848
		SFT (120 steps)	0.683	0.681	0.681	0.681	0.2	**372**	0.6638
		SFT (1 epoch)	0.714	0.713	0.708	0.710	1	1445	0.6681
	zephyr-7b-beta	base model	0.654	0.675	0.623	0.648	NA	NA	31.199
	stablelm-2-zephyr-1.6b	base model	0.553	0.548	0.491	0.518	NA	NA	0.3344
		PEFT (prompt tuning)	0.623	0.583	0.616	0.599	2	1391	0.3346
	tinyllama-bnb-4bit	base model	0.486	0.553	0.519	0.536	NA	NA	4.0390
		SFT	0.566	0.567	0.567	0.567	1	459	0.4165
	bloomz-1b1	base model	unstable	NA	NA	NA	NA	NA	NA
		PEFT (prompt tuning)	0.581	0.592	0.583	0.583	2	1158	0.2690
	bloomz-560m	base model	unstable	NA	NA	NA	NA	NA	NA
		PEFT (prompt tuning)	0.618	0.621	0.699	0.615	2	624	**0.1258**

3.4.1 SLM Results

Base models: The base versions of BERT, DistilBERT, and RoBERTa models exhibited very low precision, recall, and F1 scores on the Liar test dataset. Consequently, the F1 score, which combines precision and recall, was also extremely low. BERT accuracy of around 0.542 was only slightly better than random guessing, as the predicted outputs were all 0s, whereas DistilBERT and RoBERTa had predicted outputs of all 1s giving an accuracy around 0.458. This poor performance is attributed to the fact that these base models were not pre-trained specifically for fake news detection tasks. As pre-trained language models trained on general language data, they may not capture the nuances required for this specialized task without fine-tuning or transfer learning.

Fine-tuned models: To improve the performance of these models on the Liar dataset, we employed supervised fine-tuning on these pre-trained language models. During training, the pre-trained models were initialised with their pre-trained weights, and then their weights were updated by feeding the Liar dataset through the models in batches of 8 samples. This ensured efficient processing during evaluation. The Adam optimizer was used to update the weights based on the calculated gradients, with a learning rate of $3e-06$ and a linear learning rate scheduler. A smaller learning rate was chosen to ensure a stable and gradual learning process. This process was repeated for 5 epochs, allowing the models to learn the patterns and nuances specific to the task at hand on the Liar dataset.

The fine-tuned BERT, DistilBERT, and RoBERTa models showed significant performance improvements over their base versions on the Liar dataset. BERT achieved a test accuracy of 0.668 and an F1 score of 0.652, while DistilBERT had a test accuracy of 0.642 and F1 score of 0.642 after fine-tuning. RoBERTa performed the best with a test accuracy of 0.677 and F1 score of 0.658 after fine-tuning, demonstrating its effectiveness in classifying true and false statements. In terms of training time, BERT took 299 seconds, RoBERTa took 308 seconds, while DistilBERT was the fastest to train, taking only 168 seconds. The lower training time for DistilBERT can be attributed to its distilled nature, which makes it a smaller and more computationally efficient model compared to BERT and RoBERTa. As for efficiency during inference, DistilBERT had the lowest average inference time per news item (0.0016 seconds), making it computationally efficient due to its distilled nature, followed by RoBERTa (0.0026 seconds), despite being the best performer, and BERT (0.0027 seconds), being least computationally efficient. It's worth noting that while the differences in inference times per news item may seem small, they can become significant when dealing with large volumes of news articles or real-time applications where low latency is crucial.

These results align with our expectation since pre-trained models exhibited higher validation loss with lower accuracy and F1 scores compared to fine-tuned models. Fine-tuning enabled the models to adapt weights, learn task-specific patterns, adjust internal representations, and capture relevant features more effectively, leading to better performance metrics.

3.4.2 LLM Results

Base models: We see that base models exhibited poor performance in terms of accuracy, with some of them not even generating appropriate labels and others having accuracies around 0.50 and 0.55, which is equivalent to randomly guessing a label. The most accurate base model, with a performance better than even some of the fine-tuned models, was zephyr-7b-beta. Fine-tuning zephyr-7b-beta wasn't possible as loading the model requires more than 20 gb of RAM and the memory saving version through Unsloth wasn't available. As the model wasn't fine-tuned with Liar, the responses generated are long and not always optimised to the task we want to perform, which results in a longer inference time. However, the long inference time cannot only be explained by the absence of fine-tuning, as other base models have on average 4 seconds of inference time per a news statement in comparison to 30 seconds for zephyr-7b-beta, which is exceptionally slow. The fastest base model is llama-3-8b-bnb-4bit with 3.8 second inference time per news, while having an accuracy of 0.61, which makes it the best choice for efficient inference when one doesn't want to deal with fine-tuning.

Fine-tuned models: We observe that two of the fine-tuned models (gemma-7b-bnb-4bit and mistral-7b-bnb-4bit) surpass 70% accuracy and perform better than the best SLM, fine-tuned roberta-base. This is a surprising result given that there exists evidence by Hu et al. (2024) that LLMs underperforms SLMs in fake news detection on some datasets [2]. We note that precision, recall, and F1 Scores follow the same pattern as the accuracy.

We see that larger models in terms of parameter size perform better in terms of accuracy, which is expected given that they are able to capture more complex patterns and are able to reason better. On the other hand, there is no causal relationship between model size and fake news detection performance. Llama-3-8b was trained on 15T + tokens, whereas Gemma-7b had a training data of 6T tokens. We don't have official information about the token size of the training dataset used in Mistral-7b. This information suggests that it is possible that a model smaller in parameter and training data size can outperform a larger LLM in a specialised task such as fake news detection.

We also note that there is not a huge difference between training times of large and small LLMs, as we are using Unsloth to accelerate training. Overall, a single epoch of training took 1167 seconds on average, whereas if we only look at 7b+ models, the training time is 2042 seconds on average, which is equivalent to 34 minutes. This suggests that using a single T4 GPU on Google Collab one can train and use an LLM for fake news detection.

4 Conclusion

4.1 Takeaways

Overall, our results suggest that using LLMs to detect fake news is possible using various acceleration and efficiency frameworks and has better performance than SLMs. We also suggest that one can get better performance using supervised fine-tuning with quantized and optimised models in comparison to parameter-efficient fine-tuning on relatively smaller models. However, it is important to note that fine-tuning an SLM is less complicated, less memory consuming, and faster than fine-tuning an LLM, so it might be preferable for some use cases.

4.2 Limitations and Future Directions

Claiming that we developed a fake news detector deployable in all domains and regions would be an exaggeration, as we fine-tuned our models only on the Liar dataset. There exist other datasets, such as BuzzfeedPolitical, ISOT FakeNews, and CredBank; we could integrate them to our training pipeline in the future for better generalizability.

Lester et al. (2021) claim that prompt tuning is resilient to domain shift unlike SFT, as it freezes the original model parameters [4]. It would be a good idea to compare the performance of our SFT and PEFT models on a completely different test set to see if PEFT will have a better performance than SFT.

Finally, we think that a near-to-perfect and generalizable performance in fake news detection by just looking at a politician statement or a news headline requires an up to date knowledge base and/or ability to reason about word choice, topic, and context of the statement. Fine-tuning an LLM for these objectives is not an impossible task but one that requires labelled “reasoning” data in addition to a labelled news data. We have extracted and preprocessed the Community Notes data of X (Twitter), which could serve that purpose, but it doesn’t contain the text of original tweets about which these notes are written. Using Kahneman-Tversky Optimization (KTO) is possible on Community Notes data, but it wasn’t feasible given our computational resources. So, for now, we just released the cleaned dataset on Hugging Face for public use. Moreover, LLMs themselves can be utilised to create synthetic reasoning data for fake news detection, which could be an interesting topic in line with the rising popularity of synthetic data research.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: pre-training of deep bidirectional transformers for language understanding. (2019). arXiv: 1810.04805 [cs.CL].

[2] Beizhe Hu, Qiang Sheng, Juan Cao, Yuhui Shi, Yang Li, Danding Wang, and Peng Qi. 2024. Bad actor, good advisor: exploring the role of large language models in fake news detection. (2024). arXiv: 2309.12247 [cs.CL].

[3] Junaed Younus Khan, Md. Tawkat Islam Khondaker, Sadia Afroz, Gias Uddin, and Anindya Iqbal. 2021. A benchmark study of machine learning models for online fake news detection. *Machine Learning with Applications*, 4, (June 2021), 100032. DOI: 10.1016/j.mlwa.2021.100032.

[4] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. (2021). arXiv: 2104.08691 [cs.CL].

[5] Yinhan Liu et al. 2019. Roberta: a robustly optimized bert pretraining approach. (2019). arXiv: 1907.11692 [cs.CL].

[6] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. (2020). arXiv: 1910.01108 [cs.CL].

[7] William Yang Wang. 2017. "liar, liar pants on fire": a new benchmark dataset for fake news detection. (2017). arXiv: 1705.00648 [cs.CL].