**[CM3070]**

**Final Project**

**Preliminary Report**

**Draft**

Yau Feng Yuen, Gabriel

# [1.0] Introduction

## [1.1] Template

[CM3060] Natural Language Processing – “Fake News Detection”

## [1.2] Project Overview

This project aims to address the critical issue of misinformation in the digital age.

Fake news has far-reaching, destructive consequences, and its continued proliferation on digital platforms poses significant risks to societal trust, economic stability & democratic integrity.

This project seeks to leverage advanced Natural Language Processing (NLP) techniques to create a scalable, accurate & user-friendly fake news detection system that empowers diverse user groups – including journalists and educators – to swiftly and reliably assess the credibility of news they consume.

## [1.3] Motivation

This project is motivated by the urgent need to address the real-world consequences of fake news that span several domains:

1. **Erosion of trust in journalism**

The increasing prevalence of fake news undermines public confidence in legitimate news sources. Trustworthy journalism is vital for an informed society, and the erosion of trust in journalism polarises communities, fosters hostility and impedes constructive discourse.

1. **Economic impact**

Fake news can manipulate markets and harm businesses. For instance, baseless rumours about a company’s fiscal health can lead to a sudden crash in stock prices, unfairly affecting stakeholders and investors – with retail investors being unfairly exposed to excessive risk.

1. **Public health risks**

The COVID-19 pandemic highlighted how misinformation can incite panic – for example, in Singapore, misinformation about supply shortages prompted the local populace to hoard masks and staple foods unnecessarily, straining supply chains and increasing social anxiety unnecessarily.

1. **Political impact**

Disinformation campaigns are often weaponised to distort public opinion, promote political agendas and undermine elections. These campaigns erode the public’s faith in governance.

With the increasing volume of misinformation and the speed of which it is spread, the demand for tools to quickly and effectively verify the credibility of digital news increases. A robust and adaptable fake news detection system is critical to empowering users in navigating the complex information landscape of today. The tool is tailored to address the distinct need of two primary user groups:

1. **Journalists**

Journalists need to verify the credibility of sources and combat the spread of false information to safeguard the integrity of public discourse.

1. **Educators**

Educators need to enhance media literacy and critical thinking ability through introducing tools to identify fake news and facilitate discerning news consumption.

## [1.4] Related Projects

### [1.4.1] DataFlair: Detecting Fake News with Python and Machine Learning

DataFlair uses a passive-aggressive classifier to train a model on a small dataset of 7796 rows. [1]

A screenshot of a computer

Description automatically generated

Figure 1: Screenshot of project tutorial on DataFlair [1]

While this project serves as an excellent technical demonstration for beginners, the pipeline demonstrated in this project is too simple for deployment in real-world applications. The model is also trained on an extremely small dataset, which further limits its ability to generalise to real-world applications.

As such, DataFlair’s project emphasises the need for a solution that leverages state-of-the-art technologies, and is trained on larger datasets that would not inherently inhibit the model’s ability to generalise,

### [1.4.2] ClaimBuster

ClaimBuster applies Natural Language Processing techniques to identify and evaluate factual claims primarily within political discourse.

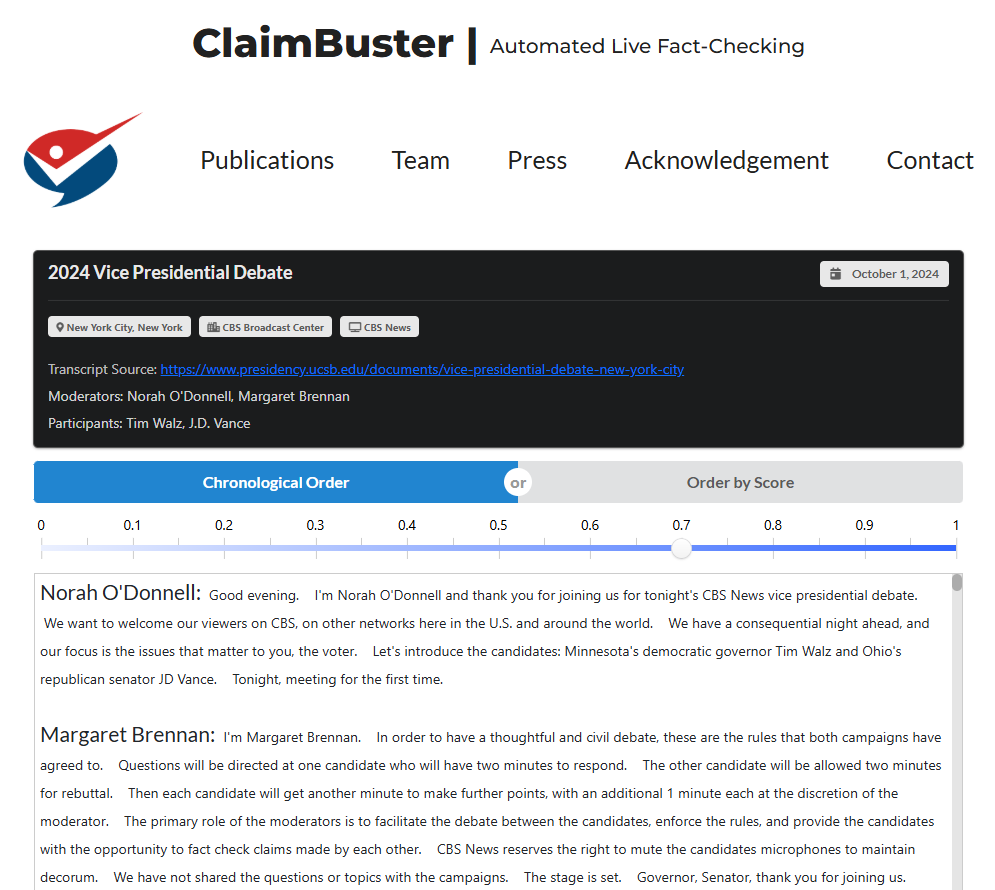


Figure 2: Screenshot of ClaimBuster's user interface [2]

While ClaimBuster is effective in detecting factual claims, its narrow focus on political contexts limit its applicability to broader domains.

As such, ClaimBuster emphasises the need for a solution that is generalisable and scalable across various topics and misinformation domains.

### [1.4.3] Snopes

Snopes is a widely-recognised platform for human fact-checking that relies on human subject-matter experts to manually verify claims, evaluate sources, and analyse misinformation across various topics.

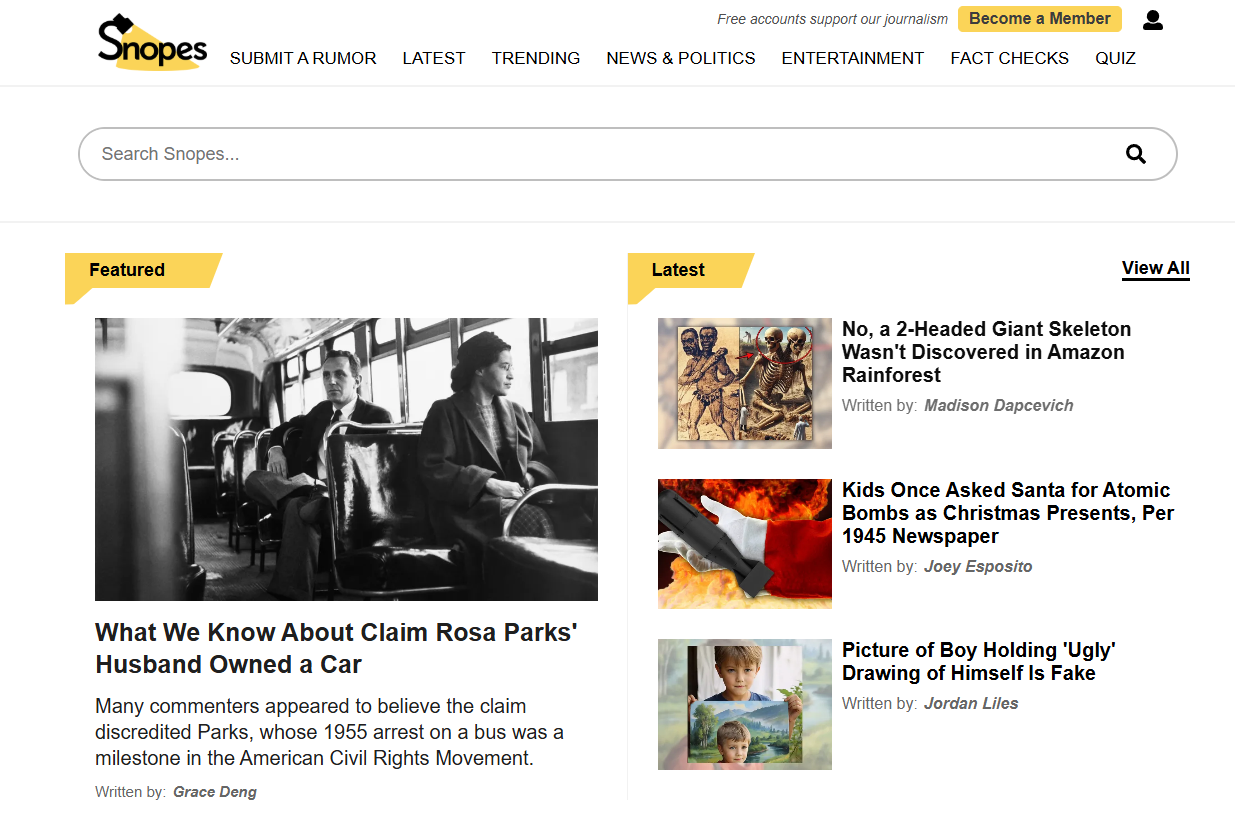


Figure 3: Screenshot of Snopes' homepage [3]

While Snopes is highly credible, its manual nature is inherently slow and labour-intensive, and not scalable for real-time verification tasks.

As such, Snopes’ lack of speed & scalability creates a gap that my solution will seek to fill by accelerating the speed of fact verification and being able to handle large volumes of information to bridge the gap between speed and accuracy.

# [2.0] Literature Review

Detecting and mitigating the spread of fake news has become a critical area of research within computer science, particularly within the discipline of natural language processing (NLP). This literature review examines existing approaches and technologies in fake news detection with a focus on their relevance to their implications on the design and implementation of my solution.

## [2.1] Machine Learning Approaches

Several projects have tackled the challenge of fake news detection with machine learning approaches.

### [2.11] Passive-Aggressive Classifier

The project brief included an example project by DataFlair that uses a small dataset of 7796 rows with news articles labelled as true/fake. TF-IDF vectorisation is first performed to pre-process text data, then a Passive-Aggressive Classifier is applied to train a machine learning model. [1]

This resource serves as an excellent technical demonstration for beginners as it introduces & explains

the fundamental steps of data pre-processing, feature extraction, model training, & pipeline building.

However, this resource has too many limitations:

1. The dataset is not representative of real-world scenarios – it is too small and lacks variety, thus limiting the model’s ability to generalise effectively.
2. The model is excessively simplistic – the Passive-Aggressive Classifier is easy to implement but is inadequate to handle nuanced language patterns & context in fake news, and lacks the sophistication of modern NLP techniques (e.g. transformer-based models), thus only achieving a accuracy of 92.82% [4]. This is further evidenced by Chang’s (2024) comparison of machine learning & deep learning algorithms, where Passive-Aggressive Classifiers were ranked 7th in fake news detection performance compared to other algorithms and outperformed by all deep learning algorithms [4].
3. This project does not address scalability concerns or real-time detection capabilities and thus is not suitable for deployment in real-world settings.

### [2.12] Random Forest Classifier

Random forest classifiers are ensembles of multiple decision trees and have been widely employed in fake news detection research.

A potential problem with Random Forest Classifiers is that as they comprise numerous decision trees, their performance also similarly declines on imbalanced datasets, as demonstrated by Huh (2021) where standard Random Forest classifiers achieved a relatively high false negative rate, a recall of 0.102 and precision of 0.365. This illustrates that 36.5% of predictions in the minority class are correct, and that standard Random Forest classifiers failed to predict approximately 90% of the minority class when presented with an imbalanced dataset. [5]

While it is possible to improve the performance of Random Forest Classifiers when dealing with imbalanced dataset, using other algorithms may prove to be more effective. In Chang’s 2024 comparison of ML and DL algorithms, random forest classifiers ranked 9th, outperformed by aforementioned Passive-Aggressive Classifiers and all deep learning algorithms. [4]

## [2.2] BERT

BERT (Bidirectional Encoder Representations from Transformers), first introduced by Devlin et al. (2019), is designed to pre-train deep bidirectional representations from unlabelled text by jointly conditioning on both left and right context in all layers. [6]

This means that BERT models are effectively able to model contextual relationships in textual data and thus are able to discern nuanced linguistic patterns. Coupled with the flexibility of transformers allowing them to be fine-tuned for task-specific applications, they have recently been a central focus in the field of NLP-based fake news detection.

BERT consistently outperforms traditional machine learning and other deep learning models. Chang’s (2024) comparison of machine learning & deep learning algorithms has shown that BERT is the highest-performing algorithm amongst those tested, with unprecedented scores of 99.95% accuracy, precision, recall & F1 score. [6]

BERT models have their limitations. Transformer architectures like BERT have high computational demands, as the pre-training corpus included Google’s BookCorpus and English Wikipedia, comprising 800 million and 2500 million words respectively for a total of 3300 million words. However, innovations such as MobileBERT, a thinner model that requires less computational power [7], have surfaced to meet the rising demand to leverage BERT’s capabilities within smaller computational environments.

Additionally, transformer models are inherently not very interpretable. Korolev et. al (2023) asserts that state-of-the-art models such as BERT are highly parameterised black boxes [8]. The opacity of transformer-based models has driven interest & demand in explainable AI techniques such as attention visualisation.

Thus, BERT’s unparalleled performance establishes it as a cornerstone for this project. However, in implementing BERT, it is important to address scalability concerns through the implementation of lighter variants such as MobileBERT that are less computationally intensive. Additionally, it will be pivotal to incorporate explainable AI techniques to enhance the model’s explainability and interpretability, which would then allow users to trust the model.

## [2.3] Hybrid Approaches

Hybrid approaches to fake news detection typically combine content-based and context-based analyses. They often integrate multimodal data – such as text, images, social network metadata – and utilise diverse sources of information to enhance their accuracy and performance.

Hybrid approaches aim to provide a more comprehensive perspective on misinformation propagation through the integration of complimentary data streams.

### [2.31] ClaimBuster

ClaimBuster is a web-based, automated, live fact-checking tool developed by the University of Texas [9].

Hassan et al. (2017) detailed how ClaimBuster works in their published paper “ClaimBuster: The first-ever end-to-end fact-checking system”. It implements a ‘claim monitor’ that continuously monitors and retrieves texts from various sources including broadcast media through a decoding device to extract closed captions, social media, and websites (such as the transcripts of Australian parliament proceedings). Factual claims to be checked are then identified by a ‘claim spotter’ that scores sentences’ likelihood of containing a factual claim using a classification & scoring model. Then, a ‘claim matcher’ takes an important factual claim identified, and searches a fact-check repository and returns fact-checks matching the claim. If a matching fact-check cannot be found, the claim checker queries external knowledge bases and the Web with a question generation tool, before a fact-check reporter finally delivers a report to users through the project website. [10]

ClaimBuster is effective in structured environments such as political debate analysis, where domain-specific knowledge bases facilitate precise fact-checking. Additionally, as pre-verified claims are combined with linguistic patterns, ClaimBuster offers a high degree of accuracy.

However, due to the manual effort required for database curation, scalability of ClaimBuster is constrained, and its applicability in rapidly-evolving contexts may thus be limited. Additionally, as it relies on domain-specific datasets, its ability to adapt to diverse misinformation scenarios may be restricted.

### 

### [2.32] FaKnow

FaKnow is a library designed to standardise the development & evaluation of fake news detection algorithms by integrating various fake news detection algorithms. [11] The library includes a variety of widely-used models, categorised into content-based & social context-based approaches within a single unified framework. Additionally, it also offers functionalities for data processing, model training, evaluation, visualisation and logging to enhance reproducibility and reduce redundancy in fake news detection research.

Graves and Cherubini (2016) emphasise that reproducibility is a persistent challenge in misinformation research due to the lack of standardised datasets and frameworks [12]. FaKnow addresses this critical issue by standardising the implementation of various fake news detection algorithms. Additionally, by encompassing both content-based and context-based models, researchers can use FaKnow to explore & evaluate various different approaches within a single platform.

However, FaKnow has several limitations within the context of my project:

1. PyTorch framework dependency

As FaKnow is built on PyTorch, researchers using other deep learning frameworks (e.g. TensorFlow) may face challenges integrating existing models/workflows to this library. Dependency on a single framework may limit adoption amongst researchers with established preferences or institutional constraints.

1. Continuous maintenance demand

As fake news detection algorithms rapidly advance, continuous updates to the library are required to incorporate the latest models and techniques. This presents maintenance challenges for both users and developers of FaKnow.

Ultimately, while FaKnow presents limitations in its PyTorch dependency and continuous maintenance demand to stay abreast of the latest advancements, its strengths in standardisation and usability make it a valuable tool.

## [2.4] Human-operated Fact Checking Websites

Human-operated fact checking websites such as PolitiFact, FactCheck.org, and Snopes, play a critical role in combating misinformation by employing teams of human experts to evaluate the truthfulness of news content and factual claims.

These websites offer several strengths – most notably, they are trusted by the public. Human-operated platforms often maintain higher level of trust & credibility compared to automated systems. Liu et al. (2023) assert that this occurs because users “doubt the ability of machines to adjudicate factual disputes and thus perceive AI-based fact-checkers as less credible than human-based fact-checking services.” [13] Yang et al. assert that this is because automated methods are error-prone [14].

Additionally, the rigorous methodologies that human fact-checkers employ ensure high accuracy in identifying and categorising false information. For example, PolitiFact displays a “Truth-O-Meter” that provides a granular evaluation of claims which serves to aid public understanding of misinformation. Additionally, these websites usually provide context and explanations for their evaluations, which promotes media literacy and critical thinking amongst users, but also transparency which helps to foster user trust.

However, human fact-checking also has inherent problems, the most crucial of which is the lack of scalability of this solution. Lin et al. (2023) assert that traditional manual fact-checking is time-consuming and labor-extensive, which cannot scale up with the unprecedented amount of dis- and mis-information on social media [15].

Additionally, perceived or actual biases in the evaluation of factual claims can undermine user trust, especially amongst ideologically polarised audiences. Human fact-checking can also lag behind the viral spread of misinformation due to the time taken to verify claims, reducing their impact in countering real-time viral narratives.

Ultimately, this is relevant to my solution as insights from the strengths of human-operated systems can help to inform the integration of explainable AI features into my solution.

## [2.5] Evaluation

My review of existing methodologies highlights several critical areas where innovation can provide additional value to state-of-the-art systems.

My solution must address the scalability concern of human fact-checking that drives demand for algorithmic fact-checking in the first place, through incorporating optimising transformer architectures (i.e. BERT).

Additionally, in order to foster user trust & credibility, the development of interpretable AI methods to address the “black box” nature of existing models is crucial. As an additional transparency measure, it would be ideal to explain to users in more easily-understood means, such as through a short paragraph as opposed to through attention heatmaps.

# [3.0] Design

## [3.1] Project Overview

Project Template: [CM3060] Fake News Detection

This project addresses the pervasive issue of misinformation by designing an AI-powered Fake News Detection system.

My proposed solution will be presented as a Command Line Interface (CLI) application that will leverage advanced Natural Language Processing (NLP) techniques to classify user-inputted news articles or headlines as fake or real, empowering users with tools to verify the credibility of online content.

To achieve that goal, the primary objective is to build a data pipeline to process and link data effectively, so it can be coupled with a query generation strategy that will deliver on accuracy.

## [3.2] Domain & Users

### [3.21] Domain

This project falls under the domain of information verification within the field of journalism & digital media. This domain addresses challenges such as identifying misinformation, improving public media literacy, and mitigating the societal impact of fake news.

### [3.22] Users

My proposed solution is built with the view that a diverse group spanning various demographics would find value in this solution. These users would all have varied requirements and use cases, and as such I have elected to focus on groups of primary users while keeping the needs of secondary users in mind:

1. **Journalists (primary user)**

My proposed solution will create value for journalists. Their demographic profile spans ages 25 ro 60, with a media scope that includes traditional newspapers, digital platforms, freelance journalism & multimedia reporting.

Journalists operating in different regions will have different regional challenges. For example, journalists in the Middle East face misinformation challenges addressing religious & ethnic sensitivities, increasing the need for source validation tools, whereas Western journalists struggle to target digital disinformation – specifically, journalists in democracies such as the United States have challenges combating election-related fake news.

Journalists would use my solution to fact-check sources, build credibility in reporting, and to investigate information cascades.

Meanwhile, I expect that there exists a sub-demographic of journalists that will not use my application. One such sub-demographic of journalists exist in highly state-controlled environments (such as North Korea) where independent news verification is not feasible or allowed.

Additionally, journalists part of publications (or other entities) involved in creating or propagating fake news are certainly not going to use nor promote a solution that detects fake news.

With all the above factors considered, I expect journalists to be primary users and adopters of my solution as they have a direct need for tools that aid in fact-checking and verifying sources in their everyday work. Given the immediacy of their nature of work, they require efficient & accurate solutions.

1. **Educators (primary user)**

Educators that are primarily based in academia or non-profit organisations would gain a lot of value from my solution.

While educators span various disciplines, media literacy educators would be foremost adopters amongst this demographic, as they are responsible for equipping students & communities with tools to identify misinformation. They would promote adoption of the solution by recommending (and thus, marketing) it to students and workshop attendees alike.

That said, adoption of my solution would not be exclusive to media literacy educators. Educators spanning various disciplines, teaching various subjects to various demographics of students (ranging from high school to university-level students) have an interest in teaching students to identify fake news when consuming media, and they would also promote adoption of my solution to their students.

Not all educators should be expected to promote or adopt my solution. Educators in some disciplines, such as early childhood education, would face different challenges and their lesson plans would naturally have very different learning outcomes that would not include discerning media consumption.

As educators act as ‘force multipliers’ by teaching communities of students & professionals to critically evaluate news, they play a vital role in adopting and promoting my solution and are thus expected to be primary users & adopters of my solution.

1. **Researchers (secondary users)**

Typically aged 30-60, researchers, often affiliated with universities or think tanks, often operate in data-rich environments with access to large datasets, computational resources and tools for statistical and NLP analysis.

While researchers - especially those investigating the spread & impact of misinformation, the role of algorithmic amplification, and its societal implications - will have a strong interest in analysing misinformation patterns, they are expected to be secondary users as their use of a CLI application may be occasional or project-specific rather than daily (as expected of primary users such as journalists).

Additionally, as their use case relates to academic analysis, tailoring tools to focus on academic rigor and reproducibility requires the implementation of advanced customisation ability that, if included, would make the solution excessively complex for primary users.

1. **Content moderators for social media platforms (secondary users)**

Content moderators working for social media platforms are tasked with identifying and removing misinformation from their platforms. They play a critical role in curbing the spread of fake news. Demographically, they tend to be professionals in their 20s to 40s, often with backgrounds in communication or technology.

Content moderators often handle large volumes of flagged content daily under strict organisational guidelines. They are secondary users as they already have primary tools, often GUI-based systems integrated into a larger moderation platform. The proposed solution may be added to their workflow as a supplemental tool for cases where deeper analysis is required.

## [3.3] Justification for Design Choices

The design choices are meticulously aligned with user needs, and the demands of the domain.

### [3.31] Command Line Interface Application

Command-line interface (CLI) applications are lightweight and platform-independent, meaning that the application can operate seamlessly across various operating systems, ensuring broad accessibility.

Additionally, CLI applications offer batch-processing capabilities to support large-scale data analysis.

While CLI applications are not suitable for the general public, who are generally not technically-savvy users, our primary users are technically competent who can understand and most likely already work with CLI tools, or can be easily trainable to operate CLI applications.

### [3.32] Bidirectional Representations from Transformers (BERT)

BERT is chosen as the machine learning model for several reasons – firstly, its ability to capture nuanced word meanings based on context will enhance classification accuracy for complex, ambiguous tests.

Additionally, BERT is expected to have high domain adaptability – fine-tuning BERT on a domain-specific dataset is expected to increase performance in fake news detection.

Benchmarks consistently demonstrate BERT’s superiority over traditional NLP models, thus, it is the optimal choice for this project.

## [3.4] Overall Structure

The project’s architecture comprises the following stages:

1. **Data Layer**

The data layer incorporates datasets from multiple sources such as labelled misinformation repositories and domain-specific datasets. Structured storage mechanisms (e.g. SQLite) will be used to manage raw data and processed features for easy access & reproducibility.

The pre-processing pipeline is also implemented here – with tokenisation, stopword removal, stemming & lemmatisation, and feature engineering (TF-IDF vectorisation & embeddings) to transform raw data to model-ready input.

1. **Modelling layer**

The modelling layer comprises traditional machine learning models (e.g. logistic regression, SVM) for comparative benchmarking, as well as deep learning models (i.e. BERT) fine-tuned for fake news detection. These models are combined to improve overall accuracy and robustness.

Additionally, this layer also incorporates additional NLP techniques such as attention mechanisms to enhance context understanding.

1. **Application layer**

The application layer comprises the Command-Line Interface (CLI) to accept user inputs, process articles, and deliver results in real-time.

Additionally, it will also integrate modules like SHAP (Shapley Additive exPlanations) and/or LIME (Local Interpretable Model-Agnostic Explanations) for transparency. As these modules provide users with insights into model decisions, the added transparency enhances user trust and interpretability.

1. **Evaluation & monitoring layer**

This layer comprises the evaluation framework that will track performance metrics (i.e. precision, recall, F! score, ROC-AUC) during training and deployment phases.

This layer will also include error analysis tools to identify common misclassifications and potential biases to guide iterative improvements, and tools to log system performance metrics to ensure reliability and scalability.

1. **Deployment layer**

The solution will be packaged into a deployable Python package, with dependencies managed through Conda.

## [3.5] Technologies & Methods

|  |  |
| --- | --- |
| **Programming Language** | Python |
| **Key Libraries** | **Scikit-learn** – tools for traditional machine learning  **Transformers (Hugging Face)** – fine-tuning BERT for contextual NLP tasks  **Pandas** – for data cleaning & manipulation  **NumPy** – for numerical computation  **Matplotlib** – for data visualisation  **Click** – to simplify CLI creation & improve user interaction |
| **Evaluation Techniques** | Cross-validation  ROC curves  SHAP/LIME for explainability |

## [3.6] Work Plan

A screenshot of a data

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

## [3.7] Testing & Evaluation

### [3.71] Test Plan

1. **Functional testing**

* Verify end-to-end functionality of the system, from input to output.
* Test BERT model on separate test dataset to evaluate metrics such as accuracy, precision, recall, F1 score.
* Additionally, use confusion matrices to analyse false positives/negatives.

1. **User testing**

* Collect feedback from target users on usability & effectiveness
* Conduct surveys with target users assess ease of use, clarity of results & overall satisfaction.
* Distribute a SUS (System Usability Scale) questionnaire to quantitatively evaluate usability
* Include task-based performance metrics (e.g. time on task, error rates) to measure efficiency.

1. **Security testing**

* Perform input validation tests to test again SQL injection and other vulnerabilities
* Ensure the CLI handles malformed or adversarial inputs gracefully

### [3.72] Evaluation Metrics

#### Model Evaluation Metrics

1. Accuracy
2. Precision
3. Recall (Sensitivity)
4. F1 score
5. ROC-AUC Curve (if necessary)
6. Calibration analysis to ensure model confidence aligns with prediction accuracy

#### User Evaluation Metrics

1. System Usability Score (SUS) score – to assess overall usability
2. Interpretability Score – users rate the clarity of flagged explanations on a 1-10 scale
3. Time on task – to measure how efficiently users complete fake news identification tasks
4. Error rate – to track user misunderstandings of system outputs
5. Net Promoter Score (NPS) – to assess overall user satisfaction, and likelihood of recommending the system

# [4.0] Feature Prototype

## [4.1] Overview

The feature prototype demonstrates the implementation of fine-tuning a MobileBERT [16] model for the purpose of fake news detection.

MobileBERT is a variant of BERT chosen for its computational efficiency while retaining its performance in natural language processing tasks. The objective of this prototype is to train MobileBERT to classify political statements as either ‘fake’ or ‘real’ using a simplified, binarized version of the LIAR dataset’s labels.

This prototype’s implementation serves to assess the viability of employing MobileBERT in a fake news detection application with real-world use cases.

## [4.2] Implementation

### [4.21] Dataset Preparation

The LIAR dataset comprises political statements categorised into six classes – “pants-fire”, “false”, “barely-true”, “half-true”, “mostly-true” and “true” [17] – that encapsulate a spectrum of truthfulness. In order to simplify the classification task for this prototype, these classes were merged into simplified binary labels:

1. Fake – “pants-fire”, “false”, “barely-true”
2. Real – “half-true”, “mostly-true”, “true”

The dataset was then prepared over multiple steps:

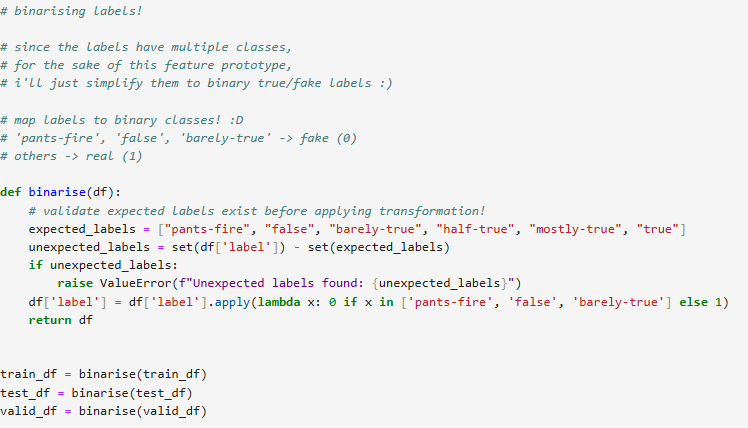
1. **Data Loading**

A screenshot of a computer code

Description automatically generated

Training, validation, and test splits were imported using pandas. This way, data would be cleanly separated, and evaluation of the model would be unbiased.

1. **Label binarisation**



To transform multi-class labels to binary labels ‘fake’ (0) and ‘real’ (1), a mapping function was implemented to map “pants-fire”, “false” and “barely-true” to ‘fake’ (0), and “half-true”, “mostly-true” and “true” to ‘real’ (1).

A screen shot of a computer code

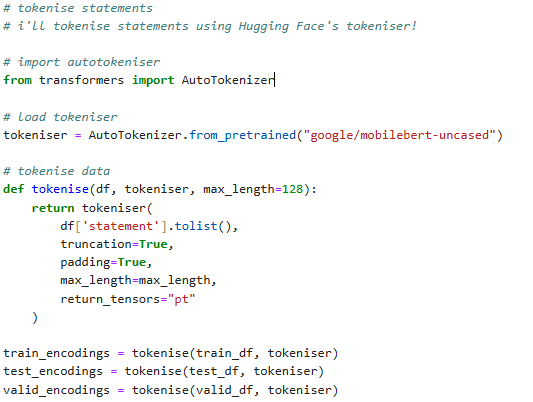
Description automatically generated

Some basic validation work was also done to verify that all labels conformed to the expected binary format.

1. **Tokenisation**

Hugging Face’s AutoTokenizer was employed to tokenise the text statements. This would ensure uniformity in length through truncation and padding to a maximum sequence length of 128 tokens.

This process preserves critical semantic features without compromising computational efficiency.



### [4.22] Model Training

MobileBERT is then fine-tuned to perform binary classification in the task-specific setting on detecting fake news.

The fine-tuning pipeline included the following key components:

1. **Dataset creation**

**A screen shot of a computer code

Description automatically generated**

The tokenised text and corresponding labels were encapsulated into PyTorch-compatible datasets to enable it to be seamlessly integrated with PyTorch’s DataLoader, thus streamlining the training and evaluation workflows.

1. **Model configuration**

**A computer screen shot of a computer code

Description automatically generated**

A custom classification head with two output labels was initialised to adapt MobileBERT’s architecture for binary classification. This modification would leverage MobileBERT’s pre-trained language understanding capabilities while ensuring alignment with the binary classification task.

1. **Training setup**

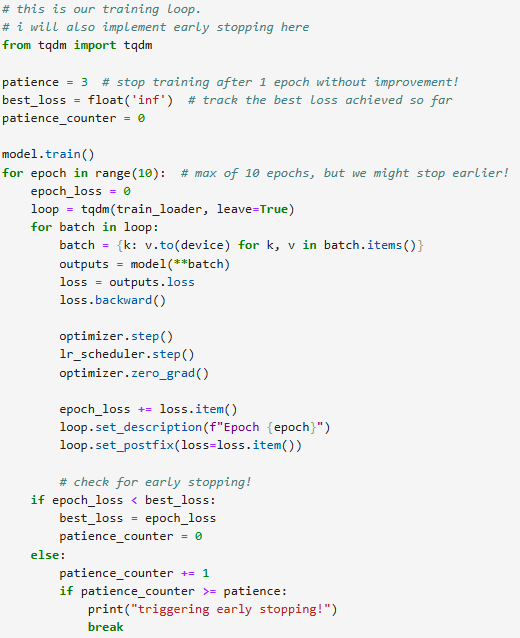


The AdamW optimiser is employed with a learning rate of 5e-5 to ensure stable gradient updates.

A linear learning rate scheduler with no warm-up steps adapts the learning rate dynamically throughout training.

The training process uses a batch size of 16. Shuffling was applied to the training set to prevent learning biases and to improve generalisation.

1. **Training loop**



The training process was capped to a maximum of 10 epochs. Early stopping was implemented based on validation loss to prevent overfitting.

A close-up of a grid

Description automatically generated

Loss values were monitored per epoch, and gradients were computed and optimised after each batch. Early stopping criteria is based on validation loss, and training would stop if no improvement was observed over three consecutive epochs.

### [4.23] Evaluation

A screenshot of a computer program

Description automatically generated

The test set was used to assess the model’s performance using standard classification metrics – accuracy, precision, recall, and F1 score.

A screen shot of a computer code

Description automatically generated

Additionally, a confusion matrix was generated to represent the distribution of true positives, true negatives, false positives and false negatives. The visualisation is meant to facilitate error analysis and highlight potential areas for model improvement.

### [4.24] Explainability

To enhance interpretability and build user trust, SHAP (Shapley Additive exPlanations) was integrated into the prototype to identify and highlight the most influential input features contributing to the model’s predictions.

This approach aims to introduce transparency to the “black box” transformer models tend to be, which would allow stakeholders to understand the rationale behind its classifications.

However, SHAP integration faced challenges with tokenised inputs. Further iterative refinements are necessary to improve its effectiveness in future iterations.

## [4.3] Results of Initial Implementation

The prototype demonstrated the feasibility of fine-tuning MobileBERT for the task of fake news detection.

The key performance metrics achieved during evaluation are as follows:

A number of numbers on a white background

Description automatically generated

Accuracy: 58.56%

Precision: 60.22%

Recall: 78.01%

F1 Score: 67.97%

These values indicate that while the model is more accurate than randomly guessing, there is large room for improvement, especially in optimising prediction and recall.

A blue squares with numbers and a white line

Description automatically generated with medium confidence

The confusion matrix reveals that the model exhibits a tendency towards higher false positives, thus significantly affecting precision. The high recall score indicates the model is capable of capturing real statements, but improving precision will be a main goal moving forward in future iterations.

## [4.4] Further Iterations

As the original implementation did not perform exceptionally well, in subsequent iterations of the prototypes, more techniques were attempted to improve the performance of the model.

### [4.41] Mixed Precision Training

The original training loop was worked on to implement mixed precision training using PyTorch’s torch.cuda.amp module to enable efficient training by performing certain operations in lower 16-bit floating point precision, while maintaining stability in others with a 32-bit floating point precision.

The potential benefit of implementing mixed precision training is to reduce computation time by utilising tensor cores on GPUs, and lower memory usage so larger batch sizes or models would be able to fit in the same memory.

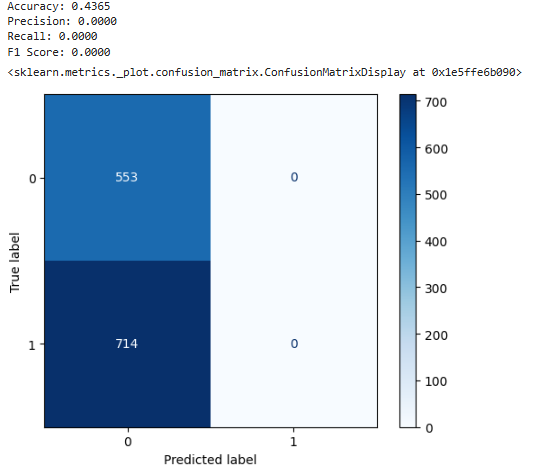
A screenshot of a computer program

Description automatically generated

To implement mixed precision training,

1. A gradient scaler ‘GradScaler()’ is initialised.
2. ‘autocast’ is used to wrap the forward pass in the training loop.
3. The scaler is used to scale the loss before calling ‘backward’, and to handle the optimizer step and update.





The results of the model trained with mixed precision training indicated a serious issue in the training process, as evidenced by the NaN loss during training across all epochs. This suggests that the model encountered numerical instability during the training loop, preventing the model from learning any meaningful patterns.

Looking at the evaluation metrics, the precision, recall and F1 score are all zero, indicating that the model trained with mixed precision training did not predict any class correctly. The confusion matrix indicated that all predictions were consistently ‘0’, regardless of the true label.

Attempts were made to reduce the learning rate, monitor gradients and adjust gradient clipping. However, unfortunately, the same issues persisted and time constraints restricted any further troubleshooting and debugging to make mixed precision training work at this time.

### [4.42] Gradient Accumulation and Clipping

Gradient accumulation enables training with larger effective batch sizes without increasing memory requirements by accumulating gradients over multiple batches before performing an update, instead of updating the model parameters after each batch.

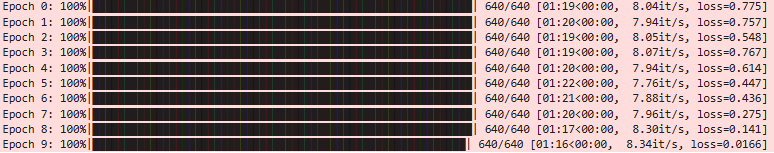
The potential improvement of implementing this is improved model stability, convergence, and numerical stability due to the prevention of exploding gradients. This also improves resource efficiency, as GPU memory usage is reduced as compared to directly increasing the batch size.

A screenshot of a computer program

Description automatically generated

The original training loop was adapted to implement gradient accumulation and clipping, through the following key changes:

1. The loss is divided by accumulation steps to ensure that gradients are appropriately scaled during accumulation,
2. Gradients are clipped before the optimiser step to prevent exploding gradients. ‘max\_norm’ sets the maximum gradient norm.
3. The model parameters are updated only after ‘accumulation\_steps’ batches.
4. ‘optimizer.zero\_grad()’ is called after the optimiser step to reset gradients.



During training, the loss steadily decreases across epochs, starting from 0.775 in epoch 0 and finally converging to 0.0166, indicating that the model is learning the training data effectively.

A number of numbers and symbols

Description automatically generated with medium confidence

A blue squares with numbers and labels

Description automatically generated

The accuracy value of 60.93% has improved compared to the initial implementation (58.56%), as has the precision score (63.88%) compared to the initial implementation (60.22%).

However, the recall has decreased, with a 70.59% value compared to the initial implementation’s 78.01%. The F1 score is also slightly lower at 67.07% (initial implementation: 67.97%). The lower recall value and the confusion matrix indicates the model fails to correctly classify more true instances, and suggests that while gradient accumulation and clipping have improved training stability, the balance between precision and recall is not fully optimised yet.

The significant decrease in training loss has not translated to better evaluation metrics. Overfitting may be occurring as the loss has converged too aggressively to near-zero, suggesting that the model may have potentially memorised the training data while not generalising well to unseen data.

Attempts to tune the learning rate, increase the patience for early stopping, and using weighted loss functions were made to improve on the results, however the evaluation metrics did not see tangible improvements, with time constraints prohibiting further troubleshooting.

## [4.5] Future Improvements

While time constraints limited the amount of iterative work done on the prototype, several possible areas for improvement were identified for future work.

1. **Addressing class imbalance**

The dataset had an uneven distribution of labels. This potentially skewed the model’s performance, and can be addressed by techniques such as oversampling or applying class weights during training to mitigate bias and improve the model’s robustness.

1. **Precision improvements**

The high rate of false positives indicates a need to refine decision boundaries, possibly by adding more nuanced examples of fake statements to the dataset, or applying advanced loss functions to penalise false positives.

1. **Expanding evaluation metrics**

Incorporating additional metrics such as ROC-AUC and calibration curves could provide more comprehensive insights and a better understanding of the model’s reliability. This would be critical when the model is deployed for real-world applications, where confidence in predictions is crucial.

# [5.0] References

[1] “Advanced Python Project - Detecting Fake News with Python,” DataFlair, Sep. 16, 2019. https://data-flair.training/blogs/advanced-python-project-detecting-fake-news/

[2] “ClaimBuster,” Uta.edu, 2024. https://idir.uta.edu/claimbuster/event/70/ (accessed Dec. 15, 2024).

[3] Snopes, “Snopes.com,” Snopes.com, 2018. https://www.snopes.com/

[4] L.-J. Chang, “Comparison of Machine Learning and Deep Learning Algorithms in Detecting Fake News,” Proceedings of the 28th World Multi-Conference on Systemics, Cybernetics and Informatics, pp. 203–209, Sep. 2024, doi: https://doi.org/10.54808/wmsci2024.01.203.

[5] K. Huh, “Surviving In a Random Forest with Imbalanced Datasets,” SFU Professional Computer Science, Feb. 13, 2021. https://medium.com/sfu-cspmp/surviving-in-a-random-forest-with-imbalanced-datasets-b98b963d52eb

[6] J. Devlin, M.-W. Chang, K. Lee, K. Google, and A. Language, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” 2019. Available: https://aclanthology.org/N19-1423.pdf

[7] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, “MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices,” arXiv:2004.02984 [cs], Apr. 2020, Available: https://arxiv.org/abs/2004.02984

[8] V. Korolev and P. Protsenko, “Accurate, interpretable predictions of materials properties within transformer language models,” Oct. 13, 2023. Accessed: Dec. 15, 2024. [Online]. Available: https://www.cell.com/patterns/pdfExtended/S2666-3899(23)00158-7

[9] “ClaimBuster,” RAND. https://www.rand.org/research/projects/truth-decay/fighting-disinformation/search/items/claimbuster.html

[10] N. Hassan et al., “ClaimBuster,” Proceedings of the VLDB Endowment, vol. 10, no. 12, pp. 1945–1948, Aug. 2017, doi: https://doi.org/10.14778/3137765.3137815.

[11] Y. Zhu, Y. Li, J. Wang, M. Gao, and J. Wei, “FaKnow: A Unified Library for Fake News Detection,” arXiv (Cornell University), 2024. https://arxiv.org/abs/2401.16441 (accessed Dec. 15, 2024).

[12] L. Graves and F. Cherubini, “The Rise of Fact-Checking Sites in Europe,” Reuters Institute, 2016. Available: https://reutersinstitute.politics.ox.ac.uk/sites/default/files/research/files/The%2520Rise%2520of%2520Fact-Checking%2520Sites%2520in%2520Europe.pdf

[13] X.-J. Liu, Q. Li, L. Wang, and M. J. Metzger, “Checking the Fact-Checkers: The Role of Source Type, Perceived Credibility, and Individual Differences in Fact-Checking Effectiveness,” Communication Research, Oct. 2023, doi: https://doi.org/10.1177/00936502231206419.

[14] J. Yang, D. A. Vega-Oliveros, T. Seibt, and A. Freitas, “Scalable Fact-checking with Human-in-the-Loop,” arXiv (Cornell University), Dec. 2021, doi: https://doi.org/10.1109/wifs53200.2021.9648388.

[15] Zi Hen Lin, Z. Wang, M. Zhao, Y. Song, and L. Lan, “An AI-based System to Assist Human Fact-Checkers for Labeling Cantonese Fake News on Social Media,” 2022 IEEE International Conference on Big Data (Big Data), Dec. 2022, doi: https://doi.org/10.1109/bigdata55660.2022.10020949.

[16] “google/mobilebert-uncased · Hugging Face,” Hugging Face. https://huggingface.co/google/mobilebert-uncased

[17] tfs4, “GitHub - tfs4/liar\_dataset: dataset liar,” GitHub, 2019. https://github.com/tfs4/liar\_dataset (accessed Dec. 15, 2024).