[CM3070]
Final Project

Preliminary Report
Draft

Yau Feng Yuen, Gabriel

Introduction

Template

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

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 create a scalable, accurate & user-friendly fake news detection system that empowers individuals to assess the credibility of information they consume. This will be achieved by leveraging advanced Natural Language Processing (NLP) techniques.

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.

2. 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.

3. 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.

4. 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, individuals need tools to quickly and effectively verify the credibility of content they engage with. A robust and adaptable fake news detection system is critical to empowering users in navigating the complex information landscape of today.

Literature Review

<include these in references section properly later>
[1] - https://data-flair.training/blogs/advanced-python-project-detecting-fake-news/
[2] - https://pmc.ncbi.nlm.nih.gov/articles/PMC7250114/
[3] - https://www.iiis.org/CDs2024/CD2024Summer/papers/SA029NJ.pdf
[4] - https://arxiv.org/html/2401.16441

Other things I may wanna include –
"A systematic literature review & existing challenges toward fake news detection models" - https://link.springer.com/article/10.1007/s13278-022-00995-5 (to discuss techniques/methods - specifically, LSTM/CNN have shown competitive performance in analysing linguistic features of news articles. BERT integration also noted for superior performance in classification tasks)

Previous work on fake news detection

DataFlair: Detecting Fake News with Python and Machine Learning

The project brief included an example project on fake news detection [1]. It 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 simple 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)
3. This project does not address scalability concerns or real-time detection capabilities and thus is not suitable for deployment in real-world settings.

This previous work is relevant to my own as it effectively demonstrates the core idea, but in pursuing a solution that is feasible for real-world application, its limitations underscore the need for:

1. More advanced models – this will allow for context-aware & accurate detection
2. Larger & more diverse datasets – to ensure better generalisation & robustness
3. Design for scalability and user-facing transparency – this example lacks in these areas

Approaches to Identify Fake News: A Systematic Literature Review by de Beer et al.

A systematic literature review by de Beer et al. categorised approaches for fake news detection into five main types: language-based, topic-agnostic, machine learning, knowledge-based, and hybrid approaches.

This paper highlights the evolution of fake news and outlines challenges in detection, including the rapid spread of misinformation, reliance on clickbait and the role of bots, ultimately concluding that hybrid approaches that integrate human expertise and digital tools would be most effective in combating fake news.

This study does several things well:

1. This study provides clarity & structure for researchers – detection approaches are classified systematically into five distinct categories, and examples such as the CSI model for hybrid approaches, and ClaimBuster for knowledge-based detection, give researchers practical insights to the implementation of these methods
2. This study provides historical & technological context – by exploring the evolution of fake news and how it is disseminated via social media & bots, and using real-world scenarios (e.g. combating COVID-19 misinformation) to emphasise relevance & urgency of fake news detection.
3. This study discusses practical implications – the discussion on hybrid models highlights the need to integrate oversight with machine learning which aligns with real-world challenges (i.e. nuanced language, bias detection)

However, it does have several limitations in the context of my project:

1. The study is overly generalised – while the categorisation is robust, the analysis lacks specificity in addressing limitations/trade-offs of each approach (e.g. the computational cost and scalability concerns of hybrid models are not discussed thoroughly). Additionally, some methodologies (e.g. topic-agnostic approach) are described conceptually with little practical implementation details.
2. The paper does not critically evaluate current datasets – while key datasets (e.g. crowdsourced fake news data) are referenced, they are not evaluated for size, diversity or bias – these are critical issues for training robust ML models.
3. This study has minimal technical depth – while this paper serves as a excellent high-level overview, it lacks detailed discussion of advanced techniques (e.g. transformer-based models such as BERT) and their implementation.

As such, this paper offers foundational insights to structure & evaluate current fake news detection methodologies.

1. The five categories of detection approaches provide a roadmap to assess existing methods and will guide how my project is positioned within the broader research landscape.
2. Rapid misinformation spread & the prevalence of bots highlight the importance of scalability and robustness in detection systems.

Comparison of Machine Learning and Deep Learning Algorithms in Detecting Fake News by Li-jing Chang

This paper by Li-jing Chang was presented at the 28th World Multi-Conference on Systemics, Cybernetics & Informatics (WMSCI) 2024 to offer an evaluation of various algorithms applied to fake news detection.

The research methodology involves pre-processing the ISOT dataset, text tokenisation through TF-IDF vectorisation, before the dataset is splits into train/test sets (80:20). Each algorithm being assessed (BERT, Bi-LSTM, LSTM, SVM, LR, SGD, PAC, NN, RF, AB, NB, DT, XGB, GB, KNN) is trained with a grid search via 10-fold cross-validation, then individually evaluated with the test dataset.

BERT is shown to be the top performer, with 99.95% accuracy, followed by Bi-LSTM (99%), LSTM (98.81%). SVM was the top-performing machine learning model with 98.65% accuracy.

This study included a wide array of algorithms to provide valuable insights into their relative effectiveness in fake news detection, and applies extremely thorough pre-processing techniques & cross-validation to assure the reliability of the findings.

However, this study has several limitations when evaluated in the context of my project:

1. This study exclusively relies on the ISOT dataset – this dataset may not encompass the full diversity of news sources & topics and may limit the generalisability of the results
2. This study does not address the interpretability of the models – particularly, deep learning models like BERT, which are seen as 'black boxes', still lack explainability

This study provides a solid foundation to understand the efficacy of various machine learning & deep learning algorithms in fake news detection. BERT's performance suggests to me that transformer-based models would make my system more accurate & reliable, however its computational demands may be a limiting factor in implementation. While the study is thorough and comprehensive, addressing challenges relating to dataset diversity, model explainability & computational requirements will be crucial to developing a solution that is effective while remaining practical.

FaKnow: A Unified Library for Fake News Detection

FaKnow is a library designed to standardise the development & evaluation of fake news detection algorithms by integrating various fake news detection algorithms. 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.

FaKnow addresses the issue of reproducibillity 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 may face challenges integrating existing models/workflows to this library.
2. 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.

Design

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. It will leverage advanced Natural Language Processing (NLP) techniques to classify news articles as fake or real, empowering users with tools to verify the credibility of online content.

Domain & Users

Domain: Journalism, Social Media (???)
(Misinformation Detection in Digital Media)

Users:
1. Journalists – to validate sources, ensure reporting accuracy
2. Educators – to teach media literacy using credible news
3. General public – to empower individuals to assess the authenticity & veracity of news articles

Justification for Design Choices

For this project to provide value to our users while solving the targeted problem, the design must prioritise meeting users' demands:

1. Accuracy – users demand precise identification of fake news to make informed decisions.
2. Transparency – to foster trust in the system, users need clear explanation(s) for why content is flagged.
3. Scalability – the system must be able to handle high volume of online news.

As such, I have made the following design choices: (question: do I need to build a UI? Will I be awarded bonus marks for it? Or do I just have the pipeline ready and user operates it via a Terminal?)

1. Natural Language Processing techniques – I utilise BERT as the literature review above shows that BERT offers the highest accuracy of all current Machine Learning techniques available to us.
2. Explainable AI – use of interpretable outputs to explain why a piece of content is flagged. (I'm tempted to say attention heatmaps, but they're also not interpretable by regular non-tech normal people)

Overall Structure

(again, do I need to have a proper front-end? Do I need to fully build/deploy this? If not NEEDED, do I get bonus credit?)

Architecture:

Back-end: FastAPI? To handle API requests & integrate ML model
Database: PostgreSQL or MongoDB
Model (?): BERT model, deployed via AWS Sagemaker
Cloud Hosting (?): AWS

Workflow:

1. User submits news article and/or headline.
2. Back-end pre-processes the input text (tokenisation, stemming).
3. Pre-trained BERT model classifies content as real or fake.
4. Explanatory features (text/scores) are returned.
5. The system logs user interactions for performance evaluation.

Technologies & Methods

(will assemble all this into a table)

Technologies:
Machine Learning: BERT fine-tuning using Hugging Face Transformer models.
Programming Languages: Python (back-end), Javascript (React.js frontend)
Database: PostgreSQL or MongoDB
Cloud Infrastructure: AWS SageMaker (to host the ML model)

Methods:
NLP Pre-processing: Tokenisation, stemming, stop-word removal
Model Training: fine-tuning BERT on fake news datasets (e.g. LIAR, FakeNewsNet, probably find more)
Evaluation Metrics: Accuracy, F1 score, explainability metrics

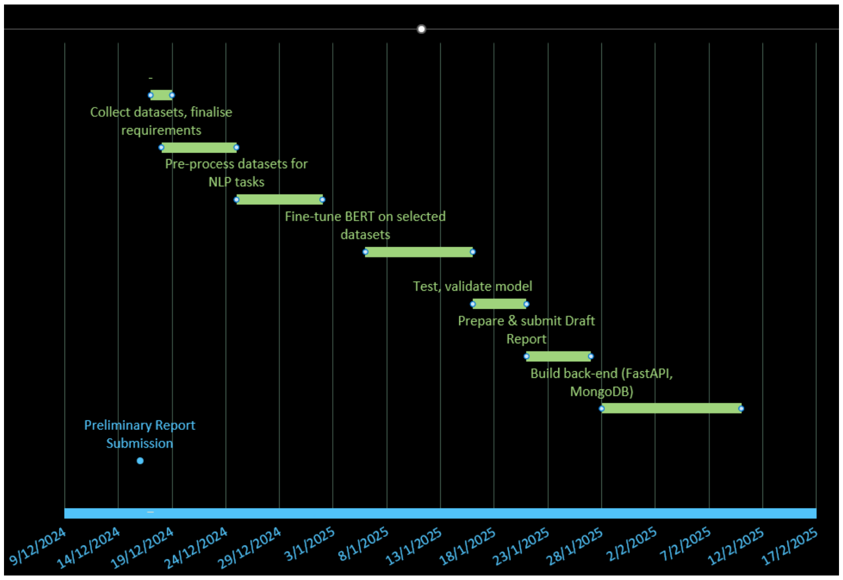
Work Plan

Milestones

This column should be ordered sequentially.	*The position column, charts milestones within the task chart.*	*Enter the date for a milestone in this column.*	*Enter a milestone description in this column. These descriptions will appear in the chart.*
No.	**Position**	**Date**	**Milestone**
1	1	16/12/2024	Preliminary Report Submission
2	2	27/1/2025	Draft Report Submission
3	3	10/3/2025	Written Examination
4	4	24/3/2025	Final Report Submission

Tasks

This column should be ordered sequentially.	*Enter the start date for each task below. For best results sort this column in ascending order.*	*Enter the end date for each task or activity below, in this column.*	*Enter tasks and/or activities in this column.*
No.	**Start Date**	**End Date**	**Task**
1	17/12/2024	18/12/2024	-
2	18/12/2024	24/12/2024	Collect datasets, finalise requirements
3	25/12/2024	5/1/2025	Pre-process datasets for NLP tasks
4	6/1/2025	15/1/2025	Fine-tune BERT on selected datasets
5	16/1/2025	20/1/2025	Test, validate model
6	21/1/2025	26/1/2025	Prepare & submit Draft Report
7	28/1/2025	9/2/2025	Build back-end (FastAPI, MongoDB)
8	10/2/2025	19/2/2025	Develop front-end
9	20/2/2025	26/2/2025	Integrate front-end/back-end/model
10	26/2/2025	2/3/2025	Conduct system testing, debugging
11	3/3/2025	7/3/2025	Scalability & performance testing
12	8/3/2025	15/3/2025	User testing, feedback collection
13	15/3/2025	20/3/2025	Refinements based on feedback
14	20/3/2025	24/3/2025	Refine final report for submission



Testing & Evaluation

Test Plan

1. **Functional testing** – verify end-to-end functionality of the system.
   Test BERT model on separate test dataset to evaluate accuracy, precision, recall, F1 score.
   Additionally, use confusion matrices to analyse false positives/negatives.

2. **User testing** – collect feedback from target users on usability & effectiveness
   Conduct surveys with target users and ask about ease of use, clarity of results & overall satisfaction.
   Also give users a SUS (System Usability Scale) questionnaire to assess performance.

3. **Scalability testing** – test system under high traffic loads to check performance (if this is required for first)

4. **Security testing**
   Input validation tests to test again SQL injection, XSS / CSRF (if we actually need to develop a front-end)

Evaluation Metrics

Model Evaluation Metrics

1. Accuracy
2. Precision
3. Recall (Sensitivity)
4. F1 score
5. ROC-AUC Curve if necessary

User Evaluation Metrics

1. Usability Score (SUS)
2. Interpretability Score – ask users to rate clarity of flagged explanations. Maybe 1-10 or Likert scale

Benchmarking – compare system’s performance against existing fake news detection tools.

Feature Prototype

My plan is to:

Implement a BERT-based model to classify input text as fake/real
Use attention heatmaps to visualise which part of text influenced classification – this may not be useful for end-user, but as a proof of concept it should be ok

Technical:

1. Prepare Dataset with a small labelled dataset (e.g. LIAR) to train/evaluate model
   Pre-process dataset by tokenising text, removing special charas, and lowercasing the text.

2. Load a pre-trained BERT model (e.g. bert-base-uncased) from the HuggingFace transformers library.
   I'll train the model on labelled data (80 training/20 test set).

3. Evaluate model on accuracy, F1 score, precision, recall.
   Additionally, analyse a confusion matrix to identify patterns in misclassification.

4. If possible – add explainability (extract attention weights from BERT to highlight significant words in text)

Tools & Libraries:
NLP/Modelling: Hugging Face Transformers, PyTorch or TensorFlow (most likely latter), scikit-learn
Data Processing: Pandas, NumPy, NLTK
Evaluation: Matplotlib or Plotly (most likely latter) for confusion matrices/attention heatmap visualisation