Combating Misinformation with BERT

Yash Serai **Mihir Kubavat**

Abstract

This project presents a study on the development and evaluation of different machine learning models for the binary classification of news articles into real and fake categories, using Natural Language Processing (NLP) techniques. With the rapid spread of misinformation, the need for accurate detection of fake news is more critical than ever to uphold the integrity of public discourse. Our project employs a range of models including LSTM, BiLSTM, CNN-BiLSTM, and BERT, each tailored to recognize the nuanced patterns that show whether it is real or fake information. Through the use of data preprocessing, model implementation, and rigorous testing, we aim to achieve high accuracy and robustness against various misinformation topics by using metrics such as accuracy, precision, recall, and F1 score in model evaluation.

1 Introduction

The project focuses on developing a machine learning model adept at accurately determining the authenticity of news articles. This involves the binary classification of news articles into real and fake categories, based on their titles and text content, using Natural Language Processing (NLP) techniques.

1.1 Motivation

- **The Importance of Accurate News Sources:** In today's fast-paced information era, accurately identifying fake news is key to preventing misinformation and protecting public discourse integrity.
- **Adapting to Evolving Misinformation:** The changing nature of fake news requires a detection model capable of adapting. Ensuring robustness against new misinformation tactics is essential to maintain the relevance and effectiveness of fake news detection.
- **Political Elections:** With the elections coming up, we noticed that fake news can really mess with people's opinions. Sometimes, when my friends talk about politics, they say things that aren't true. It makes me wonder if my sources are wrong or if theirs are. Hence the dataset we used is from 2016-2017 and contains a lot of topics related to the elections at that time.

1.2 Goals of the Project

1. **Accuracy:** The main goal is to create a model with high accuracy. Accuracy is a good measure in this case as the dataset is balanced. This is crucial due to the significant impact fake news can have on public opinion and societal dynamics.
2. **Robustness:** The model aims to be effective across various fake news types (different topics), using advanced machine learning strategies like CNNs and BiLSTM layers as well as BERT.

2 Related Work

There is significant research into fake news detection methods like in article by Yuan et al. [2023a]. Which is why recent years have seen the proposal of a variety of approaches to classify and identify fake news mentioned in De Beer and Matthee [2020]. These methods range from analyzing user behavior, leveraging knowledge bases, examining social network structures, to scrutinizing the style and presentation of content [Shahbaznezhad et al., 2021]. Innovations include the introduction of new feature sets aimed at enhancing the prediction performance of detection algorithms, the application of the Bidirectional BERT model, which has shown promise in detecting fake news by examining the relationship between news headlines and body text [Berrondo-Otermin and Cabezuelo, 2023]. Other studies have focused on feature extraction using different levels of n-grams and

proposing ensemble classification models, which have demonstrated improved accuracy in identifying fake news within specific datasets like the Kaggle fake news datasetAhmed et al., 2017. Additionally, neural network-based approaches have been explored for text analysis and fake news detection [Hamed et al., 2023].

3 Data Analysis:

The project will utilize the fake news dataset available on Kaggle, containing titles and text of news articles along with labels indicating their authenticity. The dataset is accessible at `https://www.kaggle.com/code/therealsampat/fake-news-detection/input`. The dataset consists of news articles classified as "true" or "fake." The true news dataset (True.csv) contains columns for title, text, subject, and date, while the fake news dataset (Fake.csv) includes similar columns. The true news examples shown include political and world news topics, whereas the fake news examples cover a variety of subjects such as news, politics, and government, among others. Both datasets have articles with dates ranging from late December 2017 and contains content very related to the elections at the time.

The data visualizations below show the distribution of topics for the true and fake news datasets, These visualizations help as it can show if the model would have any biases towards a specific topic (See figure 1 and 2 for the distributions). The word count helps to decide the shape of the models as well as any pre-processing steps to take as seen in figure 3. The dataset has been further processed for use in modeling by dividing it into training, validation, and test sets. The training set has 28,734 samples, the validation set has 7,184, and the test set has 8,980. The data has also been preprocessed to remove specific stop words, for example: the, a, as in order to focus on the important words. , and then concatenated into one data frame.

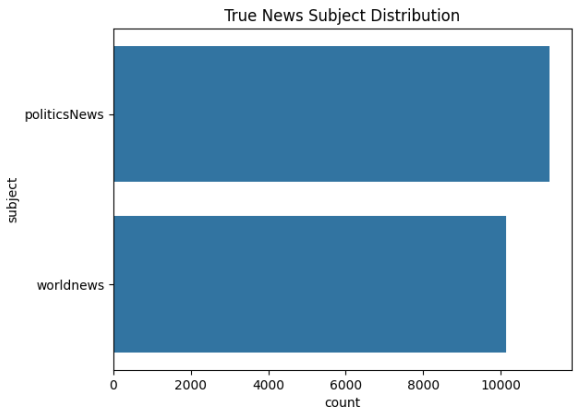


Figure 1: True Distribution

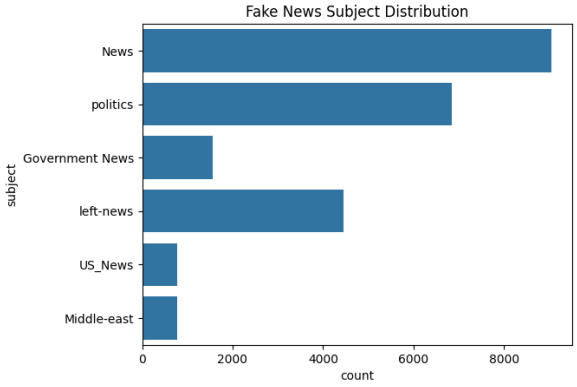


Figure 2: Fake Distribution

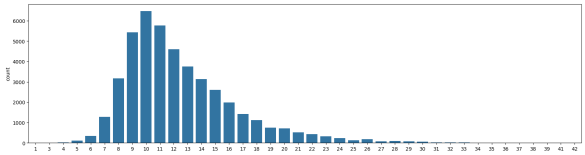


Figure 3: Length Histogram

4 Approach

4.1 Data Preprocessing

This phase involves preparing the input data for the machine learning models. This process includes Characters not between a to z or A to Z are removed, and all characters are converted to lower-case, removing suffixes such as “ed”, “est”, “s”, and “ing” to derive the token stem, and news Titles are adjusted to a maximum length (length not decided yet) to ensure consistency and to mitigate bias from titles of extreme lengths [Lass, 2021].

4.2 Word Embedding

For LSTM, BiLSTM, and CNN-BiLSTM models, a Tokenizer will be used for tokenization and se-

quences are zero-padded. An Embedding layer then learns word embeddings from the training dataset.
For BERT, the BERT tokenizer applies WordPiece Tokenization and introduces special tokens for sentence demarcation. Inputs will be converted into token, segment, and mask tensors [Yuan et al., 2023b].

4.3 Models

Various models are employed for analyzing and classifying the news [Huang et al., 2015]:

- **LSTM (Long Short-Term Memory):** Suitable for sequential data, it remembers long-term information.
- **Bidirectional LSTM (BiLSTM):** Processes data in both directions to enhance context understanding.
- **CNN-BiLSTM:** Combines CNN for feature extraction and BiLSTM for temporal data processing [Shan et al., 2021a].
- **BERT (Bidirectional Encoder Representations from Transformers):** Utilizes a deep transformer model, pretrained on extensive text data and fine-tuned for the classification task.

4.4 Evaluation Metrics

True Positive = TP
True Negative = TN
False Positive = FP
False Negative = FN

- **Precision:** Assesses the model's ability to identify only relevant instances, highlighting its specificity in positive predictions.

$$Precision = \frac{|TP|}{|TP|+|FP|}$$

- **Recall (Sensitivity):** Shows how well the model captures all relevant instances, essential for tasks where missing positives is costly.

$$Recall = \frac{|TP|}{|TP|+|FN|}$$

- **F1 Score:** Combines precision and recall into a single metric, offering a balanced view of the model's precision and sensitivity, ideal for imbalanced datasets.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

- **Accuracy:** Measures the proportion of correct predictions among total predictions, indicating overall model performance across all classes.

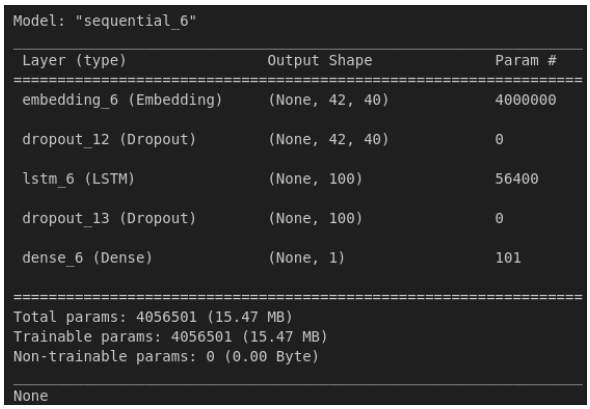
$$Precision = \frac{|TP|+|TN|}{|TP|+|FP|+|TN|+|FN|}$$

5 Results

5.1 What we have achieved

Data Pre-processing: So far we have preprocessed the dataset and prepared it for training. This included cleaning the text, removing stop words, and vectorizing the content for input into neural network models. Also, we have implemented data visualization techniques to better understand the distribution and characteristics of your dataset. We did this using python libraries such as Pandas and NumPy for data manipulation and numerical operations tasks respectively. NLP libraries like NLTK will be utilized for text preprocessing and feature extraction. We used seaborn and matplotlib for the visualization of the data.
Model Implementation: We have implemented Long Short-Term Memory (LSTM) as shown in Figure 4, Bidirectional LSTM (BiLSTM) Figure 5, and Convolutional Neural Network (CNN) model Figure 6. We used sci-kit and tensorflow to help with the implementations for the model, as well as torch and transfromers libraries for BERT .

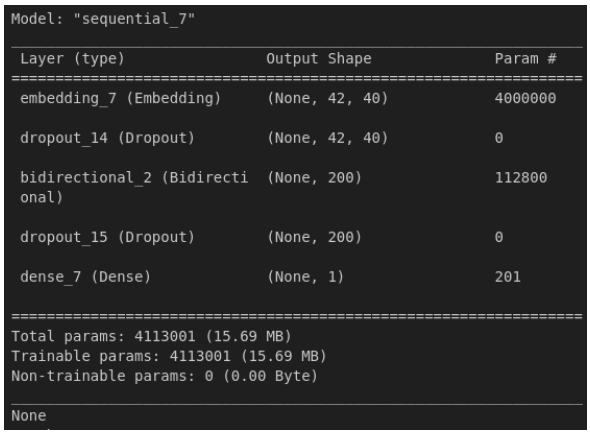
- **The LSTM model:** We utilized TensorFlow's Keras API to construct a sequential architecture aimed at binary classification. Initially, an Embedding layer transforms input tokens into dense vectors of size 40, catering to the model's need for numerical input. Following this, a Dropout layer with a rate of 0.3 reduces overfitting by randomly nullifying a portion of the inputs. The core of the model is an LSTM layer with 100 units, designed to capture long-term dependencies within the text data. Another Dropout layer follows, maintaining the model's generalization capability. The final layer, a Dense layer with a sigmoid activation, makes binary predictions. This setup, optimized for binary crossentropy and using the Adam optimizer [Saxena, 2024].



```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_6 (Embedding)     (None, 42, 40)            4000000

 dropout_12 (Dropout)        (None, 42, 40)            0

 lstm_6 (LSTM)               (None, 100)               56400

 dropout_13 (Dropout)        (None, 100)               0

 dense_6 (Dense)             (None, 1)                 101

=================================================================
Total params: 4056501 (15.47 MB)
Trainable params: 4056501 (15.47 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
```

Figure 4: LSTM Shape

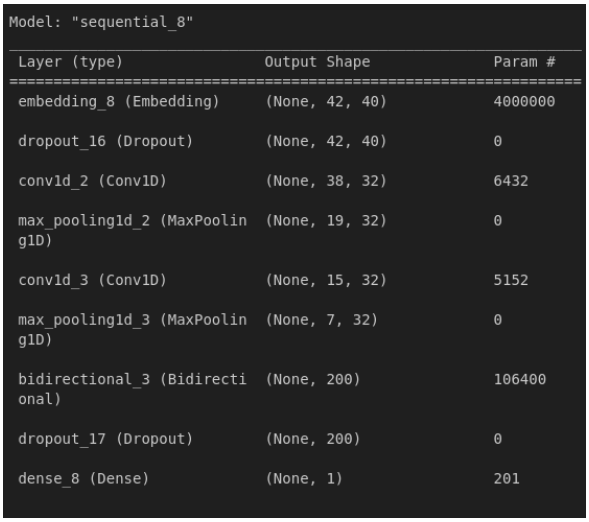
- **The Bidirectional LSTM (BiLSTM) model:** We implemented this by using TensorFlow's Keras API for binary classification, starts with an Embedding layer to convert tokens into 40-dimensional vectors, followed by a Dropout layer at a 0.3 rate to reduce overfitting. The core feature is a Bidirectional wrapper around an LSTM layer with 100 units, enhancing the model's ability to capture context from both directions of the sequence. Another Dropout layer precedes the final Dense layer with a sigmoid activation for binary predictions. Compiled with binary crossentropy and the Adam optimizer [Ghosh et al., 2023].



```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_7 (Embedding)     (None, 42, 40)            4000000

 dropout_14 (Dropout)        (None, 42, 40)            0

 bidirectional_2 (Bidirecti  (None, 200)               112800
 onal)

 dropout_15 (Dropout)        (None, 200)               0

 dense_7 (Dense)             (None, 1)                 201

=================================================================
Total params: 4113001 (15.69 MB)
Trainable params: 4113001 (15.69 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
```

Figure 5: BiLSTM Shape

- **The CNN-BiLSTM model:** We combined CNN layers for local feature extraction and a BiLSTM layer for capturing sequential dependencies in text. It starts with an Embedding layer, converting tokens into dense vectors, followed by a Dropout layer to prevent overfitting. Two Conv1D layers, each paired with a MaxPooling1D layer, extract salient features through convolution and downsample the input to emphasize important patterns. A Bidirectional LSTM layer processes these features in both directions, enhancing the model's understanding of context. A final Dropout layer and a Dense layer with sigmoid activation cater to binary classification. Compiled with binary crossentropy and Adam optimizer [Chiu and Nichols, 2015].



```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_8 (Embedding)     (None, 42, 40)            4000000

 dropout_16 (Dropout)        (None, 42, 40)            0

 conv1d_2 (Conv1D)           (None, 38, 32)            6432

 max_pooling1d_2 (MaxPoolin  (None, 19, 32)            0
 g1D)

 conv1d_3 (Conv1D)           (None, 15, 32)            5152

 max_pooling1d_3 (MaxPoolin  (None, 7, 32)             0
 g1D)

 bidirectional_3 (Bidirecti  (None, 200)               106400
 onal)

 dropout_17 (Dropout)        (None, 200)               0

 dense_8 (Dense)             (None, 1)                 201
```

Figure 6: CNN-BiLSTM Shape

- **The BERT model:**The BERT model for binary classification follows an architecture designed for understanding the context and nuances of text data. This model begins with the pre-trained "bert-base-uncased" configuration, that can comprehend the English language in a case-insensitive manner. The model's architecture comprises three critical components: embeddings, encoder, and pooler. The embeddings layer initially processes input tokens, converting them into high-dimensional vectors that capture linguistic features such as syntax and semantics. This layer is instrumental in preparing the input for further processing by encoding positional, segment, and token-type information. Following the embeddings, the encoder - a series of 12 transformer blocks - performs the core computational task. It processes the embedded input through multiple layers of self-attention and feed-forward neural networks. This structure enables the model to capture complex contextual relationships between words in a sentence, across sentences, and even in longer passages of text. The encoder's design, with

12 attention heads and an intermediate size of 3072 for its feed-forward networks, facilitates a deep understanding of the text. The pooler section of the BERT module takes the output of the last encoder layer and applies a pooling operation to derive a fixed-size output vector. This vector serves as a comprehensive representation of the input sequence, encapsulating its contextual meaning. A Dropout layer follows the BERT module, with a dropout rate of 0.1. This layer aims to mitigate overfitting by randomly omitting a subset of features during training, enhancing the model's generalization ability. The final component is a classifier, this is a linear layer that maps the pooled output to two outputs, corresponding to the binary labels. This layer makes the final prediction, determining the class of the input text based on the learned representations. Compiled with binary cross-entropy as the loss function and the Adam optimizer, this BERT model is optimized for binary classification tasks, using the profound contextual insights provided by the BERT architecture to achieve high accuracy in distinguishing between the two categories [Essa et al., 2023].


```
BertConfig {
  "_name_or_path": "bert-base-uncased",
  "architectures": [
    "BertForMaskedLM"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "gradient_checkpointing": false,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "transformers_version": "4.37.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 30522
}
```


Figure 7: BERT Shape

5.2 Issues Fixed since the milestone

Library Version Compatibility: We have been having issues implementing our code to work on the latest libraries this is due to files that the code generates having deprecated functions so the newest version which no longer supports the depreciated function will no longer work. For example, gensim was using a function for scipy but scipy no longer supported that function so gensim was not working at all so we had to roll back scipy . We have fixed this issue by finding the correct versions of the libraries and we have specified the versions required in our requirements.txt file.

Wrong Accuracy being Reported: When using our models on the dataset, we can see that the models are all returning the same accuracies and we think this is do with the library version so we are trying to update the code to the latest libraries as well as Implementing BERT successfully. We realized that this error was caused due to the way we were making our predictions. We were initially using np.argmax(model.predict(padded test), axis=-1): This method calculates the index of the maximum value along the last axis of the predicted probabilities for each instance in padded test. which was incorrect as our model was only returning a binary 1 or 0 so we changed it to (model.predict(padded test) > 0.5).astype("int32"): This method thresholds the predicted probabilities at 0.5, meaning that any probability greater than 0.5 is considered to belong to the positive class, and any probability less than or equal to 0.5 is considered to belong to the negative class.

Operating System Port: Since both me and my classmate work on windows we created the project on windows and had to port it over to ubuntu to match the requirements of the assignment, since some libraries were giving us issues namely gensim we had to figure out a way to make our code work without these libraries of by using alternatives.

BERT Implementation: We did not have BERT working by the last milestone, but we have implemented BERT, and more details can be found in the previous section.

5.3 Model Findings and Training Times

For each of the models we have provided an image with all the metrics and a heatmap which shows the true negative (top-left), false positive (top-right), false negative (bottom-left) and true positives (bottom-right).

5.3.1 LSTM:

The LSTM model achieved results of 96-97% accuracy with the results varying slightly between tests, the figure 8 below contains the detailed results along with all the metrics, and figure 9 shows the heatmap. The training time for this model is 3 minutes and 35 seconds.

	precision	recall	f1-score	support
0	0.98	0.97	0.97	4692
1	0.97	0.98	0.97	4288
accuracy			0.97	8980
macro avg	0.97	0.97	0.97	8980
weighted avg	0.97	0.97	0.97	8980

Figure 8: Results for LSTM

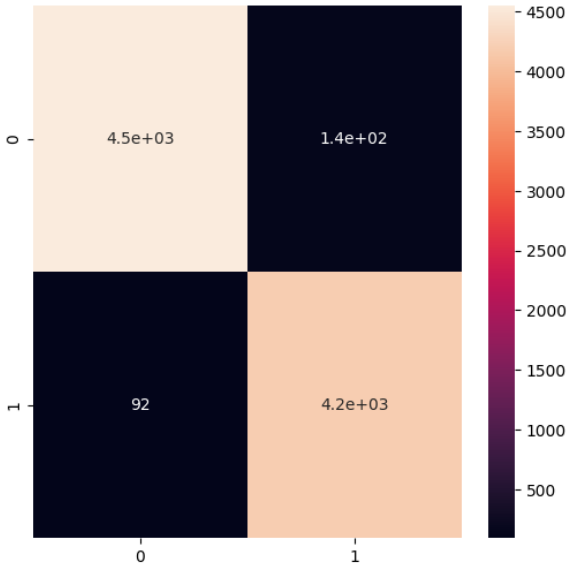


Figure 9: LSTM Heatmap

5.3.2 BiLSTM:

The BiLSTM model achieved results of 97-98% accuracy with it being closer to 97%, the figure 10 below contains the detailed results along with all the metrics, and figure 11 shows the heatmap. The training time for this model is 4 minutes and 9 seconds.

	precision	recall	f1-score	support
0	0.98	0.98	0.98	4692
1	0.97	0.97	0.97	4288
accuracy			0.98	8980
macro avg	0.98	0.98	0.98	8980
weighted avg	0.98	0.98	0.98	8980

Figure 10: Results for BiLSTM

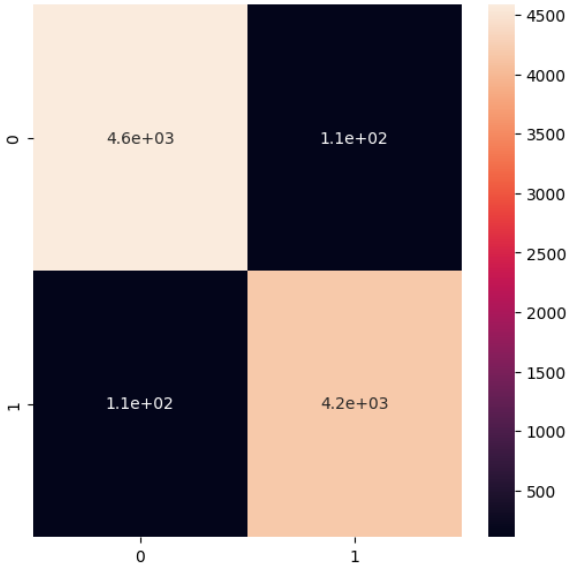


Figure 11: BiLSTM Heatmap

5.3.3 CNN-BiLSTM:

The CNN-BiLSTM model achieved results of 97-98% accuracy with it being closer to 98%, the figure 12 below contains the detailed results along with all the metrics, and figure 13 shows the heatmap. The training time for this model is 2 minutes and 17 seconds.

	precision	recall	f1-score	support
0	0.98	0.98	0.98	4692
1	0.97	0.98	0.97	4288
accuracy			0.98	8980
macro avg	0.98	0.98	0.98	8980
weighted avg	0.98	0.98	0.98	8980

Figure 12: Results for CNN-BiLSTM

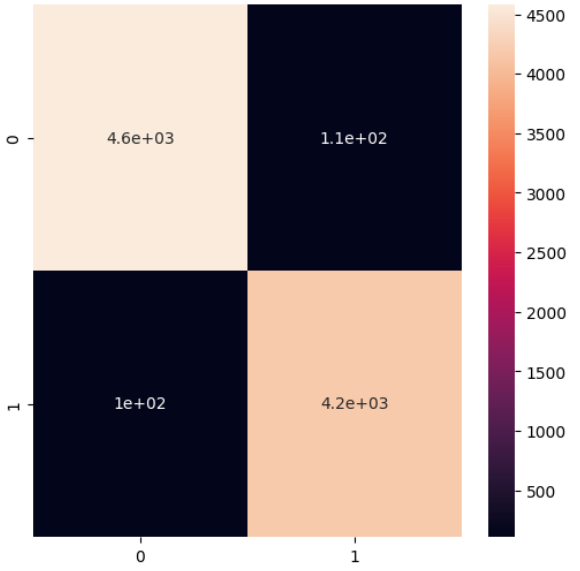


Figure 13: CNN Heatmap

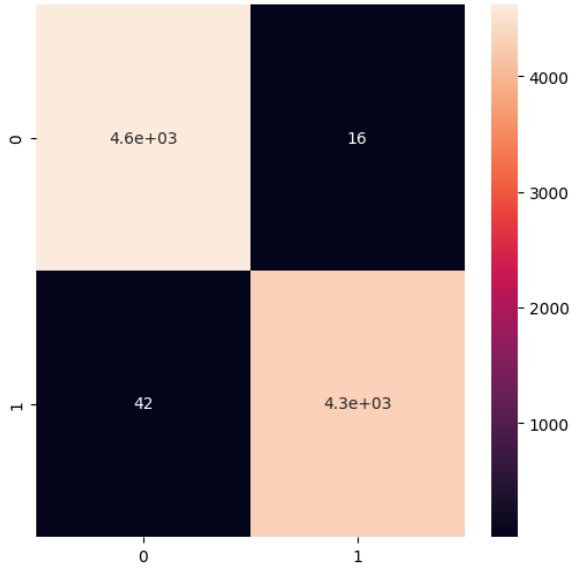


Figure 15: BERT Heatmap

5.3.4 BERT:

The BERT model achieved results of 99% accuracy, the figure 14 below contains the detailed results along with all the metrics, and figure 15 shows the heatmap. The training time for this model is 10 hours+ hence have included the already trained model as a .pth file in the submissions for you to use.

Bert Model Accuracy : 0.9935412026726058

	precision	recall	f1-score	support
0	0.99	1.00	0.99	4634
1	1.00	0.99	0.99	4346
accuracy			0.99	8980
macro avg	0.99	0.99	0.99	8980
weighted avg	0.99	0.99	0.99	8980

Figure 14: Results for BERT

6 Analysis

6.1 Model Comparison and Examples

The comparative analysis of LSTM, BiLSTM, CNN-BiLSTM, and BERT models reveals nuanced differences in their performance, due to their unique architectures and processing capabilities. Each model has strengths in certain contexts, which we illustrate through examples demonstrating instances where one model outperforms another due to its specific features.

6.1.1 LSTM vs. BiLSTM

LSTM (Long Short-Term Memory) networks are powerful for processing sequential data, capable of remembering information for long periods. However, they process data in a single direction, from past to future, which can be a limitation in understanding context fully. For example: Consider a news article with the title "Despite previous allegations, the CEO's latest venture secures unprecedented support". LSTM might focus on "previous allegations" and misclassify the article as fake due to the negative sentiment. In contrast, BiLSTM (Bidirectional LSTM) processes the data in both directions, allowing it to understand that "unprecedented support" changes the sentiment of the article, potentially leading to a correct classification as real news [Zvornicanin and Zvornicanin, 2024].

6.1.2 BiLSTM vs. CNN-BiLSTM

BiLSTM excels in understanding textual context from both directions but may miss out on capturing

local textual features effectively. For example: A fake news article might contain the phrase "revolutionary cure discovered to be a hoax" towards the end. BiLSTM might give undue weight to "revolutionary cure discovered", skewing towards a positive classification. However, CNN-BiLSTM utilizes CNN layers for local feature extraction, identifying key phrases like "to be a hoax" more effectively, leading to a correct classification as fake news [Shan et al., 2021b].

6.1.3 CNN-BiLSTM vs. BERT

While CNN-BiLSTM combines the strengths of CNNs and BiLSTMs for both feature extraction and understanding context, it may still struggle with very nuanced or complex linguistic patterns due to its reliance on the explicit features it has learned during training. Certain types of news articles can slip up CNN-BiLSTM but BERT might be able to catch it, it would have to be a subtly misleading news article might manipulate facts in a way that requires understanding implicit meanings, common knowledge, or intricate language patterns, which could be beyond the CNN-BiLSTM's capability. Here, BERT (Bidirectional Encoder Representations from Transformers), with its deep understanding of language context and nuances, can better discern the misleading nature of the article. BERT's transformer architecture allows it to evaluate the entire context of a sentence, or even multiple sentences, in one go, making it adept at catching sophisticated misinformation attempts [Zhang, 2023].

CNN-BiLSTM vs. BERT Example

- **Context**: Consider a news article with the following statement embedded within a complex narrative: "The groundbreaking study, initially heralded as a major advancement in climate science, was later revealed by experts to contain numerous methodological flaws, casting doubt on its conclusions."

- **CNN-BiLSTM's Analysis**: The CNN-BiLSTM model, with its convolutional layers, excels at extracting noticeable features and patterns from the text, such as keywords and phrases like "groundbreaking study" and "major advancement". Its BiLSTM component allows it to understand the sequence of these events, acknowledging the shift towards a negative sentiment with "methodological flaws". However, the model may not fully grasp the full extent of the implications of these "methodological flaws" on the overall narrative due to the complexity of the language used to describe the shift in the scientific community's perception of the study [Xiao et al., 2024].

- **BERT's Analysis**: BERT, on the other hand, processes the entire context of the sentence in a deeply interconnected manner, thanks to its transformer architecture. This enables BERT to understand not just the local textual features and their sequential arrangement but also the implicit meanings and nuances. BERT can recognize that despite the initial positive presentation of the study, the latter revelation about "methodological flaws" fundamentally undermines its credibility. This understanding is enhanced by BERT's ability to evaluate the context in which these revelations occur, recognizing the impact of expert opinions on the study's reception and the subsequent "casting doubt on its conclusions" [Zhang, 2023].

- **Outcome**: In this scenario, while both models might recognize the shift from positive to negative sentiment, BERT is better equipped to understand the intricate implications of the article's content, potentially leading to a more accurate assessment of its authenticity or reliability. This example highlights BERT's superior capability to handle complex, nuanced language and context, making it particularly valuable in identifying sophisticated misinformation where the devil is truly in the details [Zhang, 2023].

- **Conclusion**: Each model brings a unique set of capabilities to the table. LSTM's sequential processing is foundational, yet BiLSTM's bidirectionality offers a more rounded context comprehension. CNN-BiLSTM further enhances this with local feature sensitivity, but it is BERT, with its deep contextual understanding, that sets a new benchmark in accuracy for complex linguistic patterns. This evolution underscores the importance of choosing the right model based on the specific challenges and nuances of the task at hand [Zhang, 2023].

6.2 What the Results Mean

- **High Accuracy Achievements**: The accuracy levels achieved (96% for LSTM, 97% for

BiLSTM, 98% for CNN-BiLSTM, and 99% for BERT) indicate that the models are highly capable of distinguishing between real and fake news articles. This is a promising development for automated fake news detection systems.

- **Effectiveness of Deep Learning**: The success of these models, particularly the BERT model with an accuracy of 99%, highlights the power of deep learning in processing and understanding natural language, capturing nuances and contextual clues that differentiate fake news from real news [Nikolados et al., 2022].

- **Importance of Contextual Understanding**: The superior performance of the BERT model suggests that incorporating contextual information—something that BERT is specifically designed for—can significantly enhance the model's ability to discern the authenticity of news articles [Gartner, 2023].

6.3 What the Results Don't Mean

- **Not a Silver Bullet**: Despite high accuracy rates, these models are not foolproof. They are trained on specific datasets, and their performance might vary when exposed to news articles from different domains, periods, or geopolitical contexts.

- **Limited by Data**: The performance of these models is tied to the quality and diversity of the training data. Biases present in the dataset could lead to biases in predictions, limiting the generalizability of the models.

6.4 Areas for Improvement

- **Handling Evolving Misinformation**: Misinformation tactics evolve rapidly, necessitating continuous updates to the models' training data to maintain their effectiveness.

- **Reducing Training Time**: The significant training time for the BERT model (over 10 hours) suggests a need for optimization. Techniques such as model distillation, pruning, or the use of more efficient hardware could reduce training times and resource consumption.

- **Expanding the Dataset**: To improve robustness and generalizability, the dataset could be expanded to include a broader array of news sources, topics, and time periods. This would help the model adapt to new forms of misinformation.

- **Exploring Ensemble Methods**: Combining predictions from multiple models through ensemble methods could potentially enhance accuracy and reliability further.

- **Unbalanced Dataset**: In the real world, the ratio of fake to real news is not 50/50; it is usually 90% real and 10% fake news. Therefore, training the model using datasets that better simulate a real-world environment is crucial.

7 Conclusion and Future Plans

7.1 Conclusion

In this paper, we tackled the challenge of distinguishing real news from fake news using various machine learning models, emphasizing the use of BERT, a state-of-the-art model in Natural Language Processing (NLP). We explained the significance of accurate fake news detection in maintaining the integrity of public discourse, especially with the rapid spread of misinformation online. Our approach included a detailed process of data preprocessing, model implementation, and evaluation using a dataset comprising news articles classified as true or fake. We explored several models—LSTM, BiLSTM, CNN-BiLSTM, and BERT—each offering unique strengths in processing and analyzing text data. Through experimentation, we found that BERT, with its deep understanding of language context and structure, outperformed the other models in accuracy, achieving a remarkable 99% accuracy in classifying news as real or fake. Our work underscores the potential of deep learning techniques in the fight against fake news, but we also acknowledge the limitations and challenges, such as the need for continuous model updates and the reliance on high-quality, diverse training data to avoid biases. In conclusion, while our results are promising, we emphasize that no model is foolproof, and ongoing research, along with advancements in machine learning and NLP, is crucial to improving fake news detection systems.

7.2 Future Plans

For future developments, we're considering the integration of user feedback mechanisms to refine

and enhance the accuracy of our fake news detection tool. Initially, we thought that no one would use a feature to vote on the result of the models. However, observing the widespread use of community-driven features like community notes on social platforms like Twitter has shifted our perspective. We thought of implementing a feature that, when our tool flags content as potentially fake, prompts users with a message such as, "Our fake news detection system has identified this content as potentially unreliable. Please proceed with caution. If you believe this is an error, help us improve by providing your feedback." This approach not only involves the community in the verification process but also provides valuable data to further train and refine our models, making them more robust and accurate over time. Engaging users in this way could significantly enhance the tool's effectiveness and reliability, creating a more informed and discerning online community.

7.3 Contribution

This project was a collaboration between Mihir Kubavat and Yash Serai. Mihir developed the core components, including project.py, project.ipynb, and eval.py, which constituted the backbone of our analysis and evaluation framework. His work was instrumental in implementing the project's computational logic and ensuring the integrity of our results. Additionally, Mihir provided valuable assistance in integrating mathematical formulas and results into the project report, enhancing the clarity and precision of our documentation.

Yash took the lead in writing the project report, effectively synthesizing the methodologies, analyses, and findings into a coherent and comprehensive document. His efforts were crucial in articulating the project's objectives and outcomes, making complex concepts accessible to a broad audience. Yash's role extended beyond writing, as he collaborated with Mihir in detailing the project's mathematical and empirical results, ensuring a well-rounded presentation of our work.

References

Hadeer Ahmed, Issa Traoré, and Sherif Saad. *Detection of online fake news using N-Gram analysis and machine learning techniques*, page 127–138. January 2017. doi: 10.1007/978-3-319-69155-8_9. URL https://doi.org/10.1007/978-3-319-69155-8_9.

Maialen Berrondo-Otermin and Antonio Sarasa Cabezuelo. Application of artificial intelligence techniques to detect fake news: a review. *Electronics*, 12(24):5041, December 2023. doi: 10.3390/electronics12245041. URL https://doi.org/10.3390/electronics12245041.

Jen-Fu Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *arXiv (Cornell University)*, November 2015. doi: 10.48550/arxiv.1511.08308. URL https://arxiv.org/abs/1511.08308.

Dylan De Beer and Machdel Matthee. *Approaches to Identify Fake News: A Systematic Literature review*, page 13–22. May 2020. doi: 10.1007/978-3-030-49264-9_2. URL https://doi.org/10.1007/978-3-030-49264-9_2.

Ehab Essa, Khairuddin Omar, and Ali Alqahtani. Fake news detection based on a hybrid bert and lightgbm models. *Complex Intelligent Systems*, 9(6):6581–6592, May 2023. doi: 10.1007/s40747-023-01098-0. URL https://doi.org/10.1007/s40747-023-01098-0.

Rob Gartner. The power of contextual knowledge - rob gartner - medium. *Medium*, August 2023. URL https://medium.com/@RobGartner/the-power-of-contextual-knowledge-151bdd073c78.

Soumitra Ghosh, Asif Ekbal, and Pushpak Bhattacharyya. *Natural language processing and sentiment analysis: perspectives from computational intelligence*, page 17–47. January 2023. doi: 10.1016/b978-0-32-390535-0.00007-0. URL https://doi.org/10.1016/b978-0-32-390535-0.00007-0.

Suhaib Kh. Hamed, Mohd Juzaiddin Ab Aziz, and Mohd Ridzwan Yaakub. A review of fake news detection approaches: A critical analysis of relevant studies and highlighting key challenges associated with the dataset, feature representation, and data fusion. *Heliyon*, 9(10):e20382, October 2023. doi: 10.1016/j.heliyon.2023.e20382. URL https://doi.org/10.1016/j.heliyon.2023.e20382.

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv (Cornell University)*, August 2015. URL https://arxiv.org/pdf/1508.01991.

Data Lass. Nlp punctuation, lower-case and stopwords pre-processing. *Medium*, December 2021. URL https://medium.com/@LauraHKahn.

Evangelos-Marios Nikolados, Arin Wongprommoon, Oisin Mac Aodha, Guillaume Cambray, and Diego A. Oyarzún. Accuracy and data efficiency in deep learning models of protein expression. *Nature Communications*, 13(1), December 2022. doi: 10.1038/s41467-022-34902-5. URL https://doi.org/10.1038/s41467-022-34902-5.

Sawan Saxena. Understanding embedding layer in keras - analytics vidhya - medium. *Medium*, February 2024. URL https://medium.com/analytics-vidhya/understanding-embedding-layer-in-keras-bbe3ff1327ce.

Hamidreza Shahbaznezhad, Rebecca Dolan, and Mona Rashidirad. The role of social media content format and platform in users' engagement behavior. *Journal of Interactive Marketing*, 53:47–65, February 2021. doi: 10.1016/j.intmar.2020.05.001. URL https://doi.org/10.1016/j.intmar.2020.05.001.

Liqun Shan, Yanchang Liu, Min Tang, Ming Yang, and Xueyuan Bai. Cnn-bilstm hybrid neural networks with attention mechanism for well log prediction. *Journal of Petroleum Science and Engineering*, 205:108838, October 2021a. doi: 10.1016/j.petrol.2021.108838. URL https://doi.org/10.1016/j.petrol.2021.108838.

Liqun Shan, Yanchang Liu, Min Tang, Ming Yang, and Xueyuan Bai. Cnn-bilstm hybrid neural networks with attention mechanism for well log prediction. *Journal of Petroleum Science Engineering*, 205:108838, October 2021b. doi: 10.1016/j.petrol.2021.108838. URL https://doi.org/10.1016/j.petrol.2021.108838.

Li Xiao, Shijian Zhou, Fengwei Wang, and Lunkai Fu. An improved sparrow search algorithm and cnn-bilstm neural network for predicting sea level height. *Scientific Reports*, 14(1), February 2024. doi: 10.1038/s41598-024-55266-4. URL https://doi.org/10.1038/s41598-024-55266-4.

Lei Yuan, Honglu Jiang, Hao Shen, Lei Shi, and Nanchang Cheng. Sustainable development of information dissemination: A review of current fake news detection research and practice. *Systems*, 11(9):458, September 2023a. doi: 10.3390/systems11090458. URL https://doi.org/10.3390/systems11090458.

Ye Yuan, Wang Wang, Guangze Wen, Zikun Zheng, and Zhemin Zhuang. Sentiment analysis of chinese product reviews based on fusion of dual-channel bilstm and self-attention. *Future Internet*, 15(11):364, November 2023b. doi: 10.3390/fi15110364. URL https://doi.org/10.3390/fi15110364.

Bowen Zhang. A bert-cnn based approach on movie review sentiment analysis. *SHS Web of Conferences*, 163:04007, January 2023. doi: 10.1051/shsconf/202316304007. URL https://doi.org/10.1051/shsconf/202316304007.

Enes Zvornicanin and Enes Zvornicanin. Differences between bidirectional and unidirectional lstm | baeldung on computer science, March 2024. URL https://www.baeldung.com/cs/bidirectional-vs-unidirectional-lstm.