APS360 Final Report

Group 8

Fake News Detector

Emily Bao 1004931274

Mymy Tran 1005102078

Ambrose Man 1004937189

Fahim Rahman Talukder 100529533

Word Count (excluding references and title page): 2453

LINK TO THE MASTER GOOGLE COLAB:

https://colab.research.google.com/drive/11E12kykZRhDxfQw0E9ddFBGowVf8hWF1?usp=sharing

1.0 Introduction

A study published by *Science* found that false information circulates “farther, faster, deeper, and more broadly than the truth” [1]. Social media supports the spread of fake news which magnifies the difficulty of verifying the large amounts of information. In a highly digitalized world, using manual effort to classify misinformation from information is insufficient. Machine Learning (ML) can handle this ever-growing problem quite effectively as automated networks such as recurrent neural networks (RNN) can be built to meet the demands of ever increasing fake new sources which have consequential social, ethical, and physical global effects [2]. Using ML, we aim to verify the authenticity of a news article is a reliable source of information in a more efficient manner than we, as humans, are capable of on our own. Our Fake News Detector will take articles as input and use their titles and text bodies to determine if that corpora is real or fake news. Our report outlines the procedure for which we were able to solve our problem to a high degree of effectiveness.

2.0 Illustration / Figure

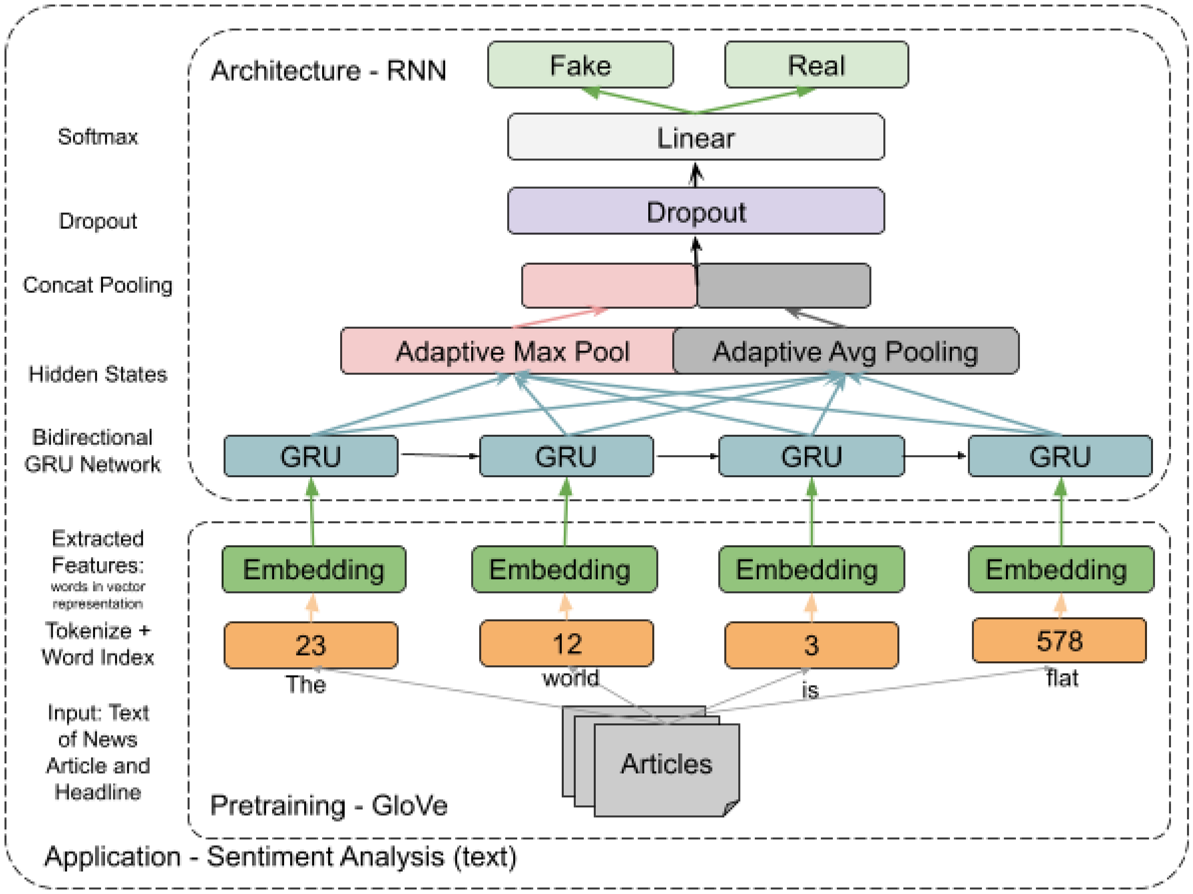


Figure 1: Top-Level Hierarchy of our Project’s Design.

3.0 Background and Related Work

Extensive resources and advancements have been made by companies and research groups to combat fake news using ML. Researchers at the *University of Washington* developed *Grover*, an AI model that learned to generate and identify fake news. They found that the model's accuracy increased from 73% to 92% after learning to generate fake news [3].

A UK-based start-up, *Logically,* developed similar ML algorithms for an AI-based phone app and browser extension to verify news, discussions and images. *Logically* uses Natural Language Processing (NLP) and comparisons with over 100,000 similar sources to rank content reliability: low, medium, or high [4].

Our project focuses on identifying fake news articles by analysing the written style of the text. However, from the aforementioned research, there exists other methods that are being used to monitor the validity of other sources of media, such as images, videos or Tweets.

4.0 Data Processing

4.1 Data Collection and Merging

For the project, we obtained two datasets (.csv files) from Kaggle. Each data sample in the final dataset contains each article's: title, text, and a label indicating authenticity: real (1) or fake (0). The first dataset (DS1) contains 44,898 news articles: 23,481 of which are fake and 21,417 are real [5]. The second dataset (DS2) contains 4,009 real and fake articles [6]. The real news articles were extracted from various reliable news networks such as *CNN, Reuters,* and *The New York Times*, whereas the fake articles were sourced from untrusted news sites. The articles covered ranging topics including politics to sports and entertainment. To merge the datasets for a total of 48,907 data samples, we cleaned and restructured DS1 and DS2 (Figure 2).

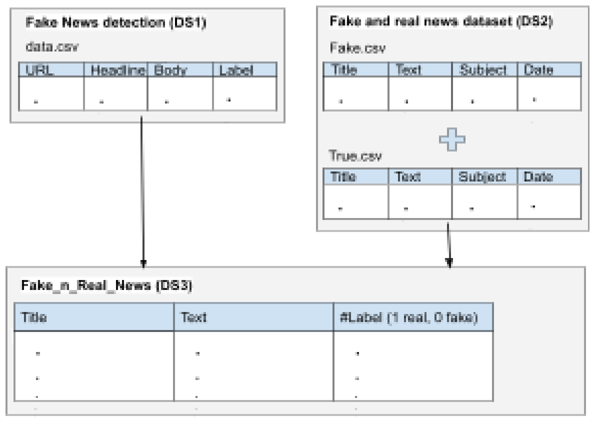


Figure 2: A Visual Representation of Data Merging

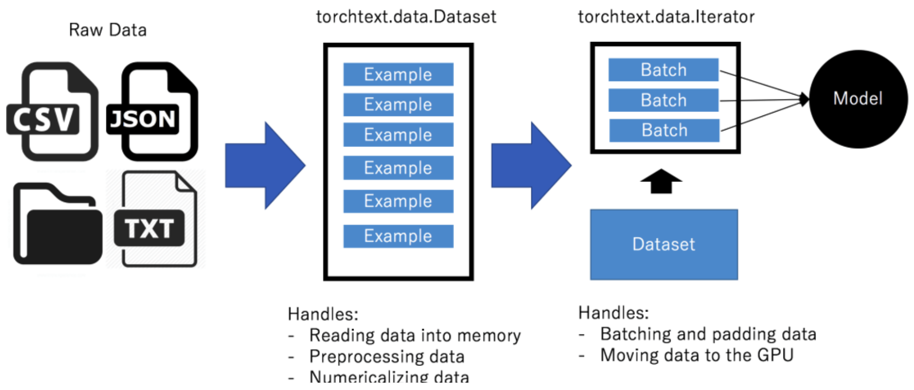


Figure 3: Data Preprocessing with PyTorch's torchtext [7].

4.2 Data Loading and Pre-Processing

Before training our model, we load our data with torchtext.data.Dataset where we randomize the data by shuffling and split the data into training, validation and testing sets, and assign it fields such as text and label (Figure 3). During preprocessing, we performed tasks such as changing all letters to lowercase and removing non-alphanumeric characters and strings, such as "https". We follow up by creating the vocabulary through mapping the 100,000 most common unique words in the train and valid data to an index. Additionally, we use a glove embedding, which maps the index to the corresponding word embedding right afterwards.

To train the model, we formatted the data using the BucketIterator(), which sorts the data according to the length of the text and batch with similar lengths. This method also reduces the padding required [8].

5.0 Baseline Model

Background of baseline models for similar NLP projects, encouraged our decision for our Random Forest model as our baseline to compare the neural network's performance to [9, 10]. This simple machine learning model trains within a matter of minutes. Since the model requires minimal tuning, the data was split 75:25 into training and testing datasets. The input was turned into a "bag of words" by keeping track of which words occurred in each article. After importing the necessary libraries, such as RandomForestClassifier from sklearn, we created the classifier with 100 estimators and 'entropy' as the criterion. The final accuracy of the Baseline Model was 70.3%.

6.0 Architecture

The Gated Recurrent Unit (GRU) network is the main architecture for our Fake News Detector. This type of RNN is the most ideal for our text-level sentiment analysis problem as this architecture's contains the following features:

1. Accepts dynamically-sized sequential input [11]
2. Recalls past events: has a memory storage mechanism [11]

Our FakeNewsConcatPoolingAdaptiveGRU has an nn.Embedding layer, rather than a direct variable glove as it allows for multiple lookups of word embeddings at once. F. First, our architecture looks up the word embeddings, then it sets a beginning hidden state with GRU. For the GRU to handle the sequences with varying length in a single batch, we call pack_packed_sequence().

The batches fed into this GRU network are sorted in decreasing order of sequence length from the batching step done in the data preprocessing step. We follow with a forward propagation to the GRU and take this output and apply the inverse operation of pack_packed_sequence(), pad_packed_sequence(), on it. Since the input texts contain hundreds of words, we would risk losing information if we only considered the last hidden state of the model.

$$\boxed{\begin{gathered} \mathrm{H} = \{\mathrm{h}_1, \ldots, \mathrm{h}_\mathrm{T}\} \\ \mathrm{h}_\mathrm{c} = [\mathrm{h}_\mathrm{T}, \mathrm{maxpool}(\mathrm{H}), \mathrm{meanpool}(\mathrm{H})] \end{gathered}}$$

Figure 4: Max-Pooling and Mean-Pooling Technique

Accordingly, we introduce the concat pooling technique (Figure 4), where we concatenate the latest hidden state with both the output of max-pooling and the mean-pooling over all the hidden states [12]. We do adaptive pooling instead of regular pooling, because the input text sequence varies in length. If we do not use adaptive pooling, we must reconfigure the stride and kernel-size parameters for every change in our input sequence length, which is not practical. We eliminate this step by using *adaptive pooling* which will automatically update the stride and kernel-size. Subsequently after, we used a dropout layer to decrease the effects of overfitting. Finally, we pass the final layer output of the last time step to the classifier, a fully connected, linear layer.

Table 1: Hyperparameter Tuning Performance

Hyperparameters	Experiment Range	Choice
Final layer Activation	{ReLU, Tanh, Softmax}	Softmax
Batch size	3 - 64	32
Learning Rate	0.00001 - 0.0005	0.0005
Dropout Rate	0 - 1	0.15
Epochs	2-15	15

7.0 Quantitative Results

We used training accuracy, validation accuracy, training loss, and validation loss to measure the performance of our models on training and validation data. The table below summarizes the results after training our model on a 4000 entry dataset.

Table 2:. Model Performance

Dataset Used	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
4000 entries Data from second dataset (DS2)*	98.83%	4.06%	98.24%	6.06%

*[See Data Processing section for details on DS2.]

The high training accuracy shows that the model was successful in learning the training data, and a similarly high validation accuracy confirms the strength of our model. It is likely the model did not simply memorize data from the training set.

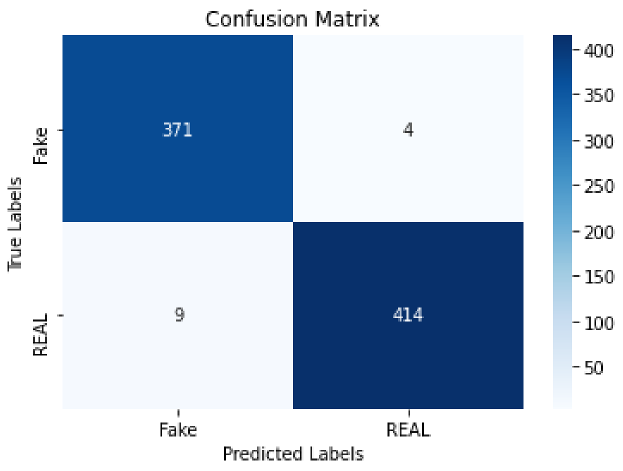


Figure 5: Confusion Matrix over 798 samples.

In addition, we used a confusion matrix (Figure 5), an effective tool for verifying the performance of classification models such as our fake news detector. The matrix shows that over 798 samples from the test set, we were able to identify 414 as true positive and 371 as true negative. In other words, 414 of 423 real news articles were correctly identified, while 371 of 375 fake news articles were correctly identified. Our model was able to classify real and fake news articles at a very high accuracy, and the correct predictions seem evenly spread over the Fake and Real classes.

8.0 Qualitative Results

Although our positive results from Table 2 indicates high performance, we tested our model using different inputs to further verify its accuracy. We noticed that the dataset had representation bias since it was mainly collected political articles from American news outlets. Furthermore, the real news articles were mainly collected from publishers with "left" political bias, whereas fake news largely contained "right" political bias [13]. As a result, our model may discriminate against an article written in a right-wing style and favour an article written in a left-wing style.

We gathered additional news articles of the inverse of typical samples in our dataset, i.e. fake-left and real-right articles, and tested the model's accuracy to see if it would still classify articles with reasonable accuracy. The results we obtained are listed below:

- 75% accuracy for right-leaning real news articles.
- 100% accuracy for left-leaning fake news articles.

These results are encouraging, because while the right-leaning real news accuracy was lower than our final model overall accuracy, it is still higher than our baseline model (70%). Furthermore, the left-leaning fake news articles were correctly predicted with very high accuracy.

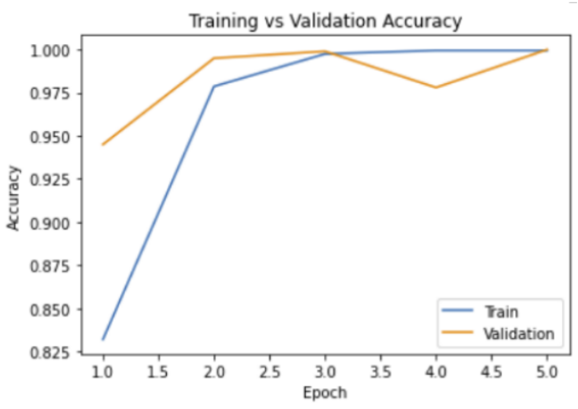
9.0 Evaluate Model on New Data

We took several efforts to ensure that the results are a good representation of the model's performance on new data. Firstly, before we started training, we separated out the dataset into a training, validation, and testing set. The testing set was not to be used to check accuracy until we had finished training and decided on a final model. This was to ensure we did not tune the model via the test set and achieve an artificially high test accuracy. After tuning hyperparameters using the training and validation dataset, we tested our model with our unseen testing dataset. The model achieved an accuracy of 97.99%.

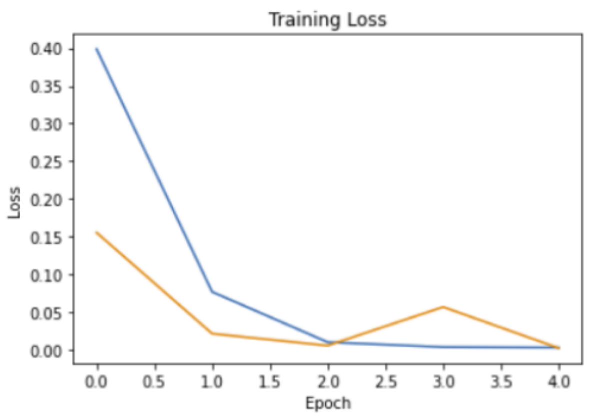
It is important to note that most of the samples in our data were political, which means our model may perform better on political topics rather than others. This is an example of representation bias as our training data did represent all possible news articles. However, users are more likely to encounter fabricated news articles on politics than other topics such as sports and entertainment.

10.0 Discussion

We are confident that our model performs well after going through several iterations using different hyperparameters and training on different datasets (as seen in The Training Spreadsheet). Before arriving at our final model, we trained our model using a 10,000 entry dataset, which consisted of data from DS1. We were able to achieve unusually high training and validation accuracies and low training and validation losses in a very short period of time.



(a) FakeNewsConcatPoolingAdaptiveGRU Accuracy Curve



(b) FakeNewsConcatPoolingAdaptiveGRU Loss Curve

Figure 6. Learning Curves - 10,000 entry dataset for the final model.

By the end of epoch 1, the training accuracy had reached 95%. The training accuracy quickly plateaus to 100% accuracy. After inspecting at DS1, we noticed some similarities and patterns in the real news data. Many news articles were all taken from Reuters and were based in Washington D.C. Therefore, the topics, views, and writing styles, were very similar. The model may have been able to achieve such a high accuracy by memorizing this style of article and labelling it as “real” and anything else as “false”.

After using our more balanced dataset, DS2, we were still able to achieve high accuracy and low loss after much more tuning and training. The learning curves in Figure 7 show a gradual increase in accuracy and gradual decrease in loss. It is more feasible that the accuracy starts at around 60% and gradually increases to 95% accuracy. In addition, it takes up to 15 epochs, rather than 5, to reach a high degree of accuracy.

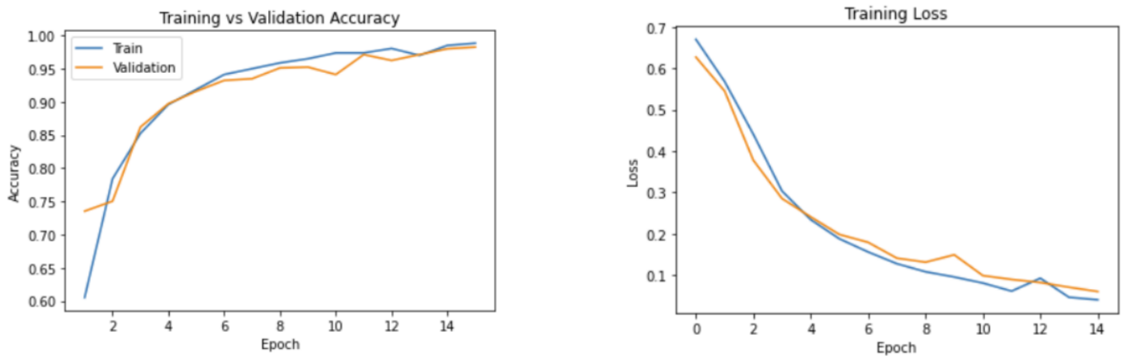
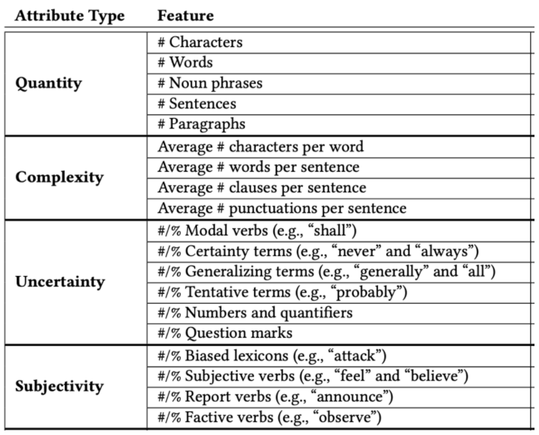


Figure 7. Learning Curves - 4,000 entry dataset.

11.0 Ethical Considerations

What people choose to trust build their core beliefs and shape the way they look at the world. There are major ethical concerns in allowing a largely opaque model to vet the authenticity of media people consume. Although our model may be more efficient, it does not have the absolute confirmation that comes from rigorous fact checking. Most publishers write with some bias, and unfortunately most of our "Real" are left leaning news, which write differently than sources with a right-skewed political bias. Furthermore, the word embeddings themselves from GloVe may hold biases, such as gender or racial bias, that unexpectedly skew the prediction. There is ethical concern that if the model is used as an absolute metric for reliability, people may be misled, as articles can be written deliberately in a certain style to fool our model.

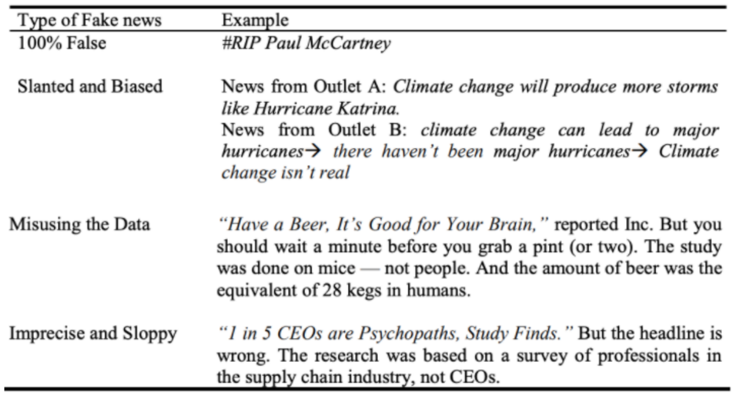
12.0 Difficulty



Attribute Type	Feature
Quantity	# Characters
	# Words
	# Noun phrases
	# Sentences
	# Paragraphs
Complexity	Average # characters per word
	Average # words per sentence
	Average # clauses per sentence
	Average # punctuations per sentence
Uncertainty	#/% Modal verbs (e.g., “shall”)
	#/% Certainty terms (e.g., “never” and “always”)
	#/% Generalizing terms (e.g., “generally” and “all”)
	#/% Tentative terms (e.g., “probably”)
	#/% Numbers and quantifiers
	#/% Question marks
Subjectivity	#/% Biased lexicons (e.g., “attack”)
	#/% Subjective verbs (e.g., “feel” and “believe”)
	#/% Report verbs (e.g., “announce”)
	#/% Factive verbs (e.g., “observe”)

Figure 8: Attributes of Text Inputs and their associated Features [14].

The difficulty of the problem at hand stems from the countless characteristics associated with NLP. News articles consist of numerous attributes and features that could help the model distinguish fake from real news (Figure 8). Considering all these subtleties involved in conveying meaning through text, this classification task proves difficult. Referring to Figure 9, fake news can be categorized into different types, leading into a multi-classification problem. However, to limit the complexity of the problem, we chose to make it a binary classification task by tagging news articles as real or fake. We implemented 2 alternate FakeNews models, which are discussed below.



Type of Fake news	Example
100% False	*#RIP Paul McCartney*
Slanted and Biased	News from Outlet A: *Climate change will produce more storms like Hurricane Katrina.* News from Outlet B: *climate change can lead to major hurricanes→ there haven't been major hurricanes→ Climate change isn't real*
Misusing the Data	*“Have a Beer, It's Good for Your Brain,”* reported Inc. But you should wait a minute before you grab a pint (or two). The study was done on mice — not people. And the amount of beer was the equivalent of 28 kegs in humans.
Imprecise and Sloppy	*“1 in 5 CEOs are Psychopaths, Study Finds.”* But the headline is wrong. The research was based on a survey of professionals in the supply chain industry, not CEOs.

Figure 9:. Types of Fake News [14].

12.1 Primary Model: Simple Text RNN

For the first implementation, similar to our final model (Figure 1) there is an nn.Embedding layer to allow for multiple lookups of word embeddings at once. This model first looks up the word embedding, then it sets a beginning hidden state. Afterwards, it follows up with forward propagation to the RNN, which consists of an input size of 50 and hidden size of 50. Finally, the final layer output of the last time step is passed to the classifier, a fully connected, linear layer.

12.2 Secondary Model: BERT

We also attempted to implement a model using Bidirectional Encoder Representations from Transformers (BERT) as transfer learning to improve the accuracy of our model. The Hugging Face transformer library was used, with the "bert-base-uncased" pretrained model in specific [15], with 12 layers, which utilized lowercase similar to our other model iterations. After going through the BERT portion of the model, which had 1,538 parameters that would be trained during our code, the outputs were to be connected to a fully connected linear layer. The plan was to add more layers and complexity between BERT and the linear layer after getting it working initially, but unfortunately, due to the difficulties faced and time constraints, this model was abandoned, and we focused instead on training and improving the final model.

References

[1] B. Marr, “Fake News Is Rampant, Here Is How Artificial Intelligence Can Help,” *Forbes,* 25-Jan-2021. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2021/01/25/fake-news-is-rampant-here-is-how-artificial-intelligence-can-help/?sh=efb14eb48e4e. [Accessed: 12-Feb-2021].

[2] K. Mok, "Deep Learning AI Tool Identifies ‘Fake News’ with Automated Fact Checking," The New Stack, [Online]. Available: https://thenewstack.io/deep-learning-ai-tool-identifies-fake-news-with-automated-fact-checking/.

[3] R. Zellers, Y. Bisk, H. Rashkin, and A. Holtzman, “Grover: A State-of-the-Art Defense against Neural Fake News,” *University of Washington.* [Online]. Available: https://rowanzellers.com/grover/. [Accessed: 12-Feb-2021].

[4] B. Marr, “Fake News Is Rampant, Here Is How Artificial Intelligence Can Help,” Forbes, 25-Jan-2021. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2021/01/25/fake-news-is-rampant-here-is-how-artificial-intelligence-can-help/?sh=efb14eb48e4e. [Accessed: 12-Feb-2021].

[5] “Fake News detection,” *Kaggle*, 07-Dec-2017. [Online]. Available: https://www.kaggle.com/jruvika/fake-news-detection. [Accessed: 12-Feb-2021].

[6] C. Bisaillon, “Fake and real news dataset,” *Kaggle*, 26-Mar-2020. [Online]. Available: https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset?select=True.csv. [Accessed: 12-Feb-2021].

[7] B. Trevett, pytorch - Text Classification. Programmer Sought, 2019.

[8] Himanshu, “Sentiment Analysis - TorchText,” Medium, 28-Apr-2018. [Online]. Available: https://medium.com/@sonicboom8/sentiment-analysis-torchtext-55fb57b1fab8. [Accessed: 03-Apr-2021].

[9] R. Gupta, “Build and Compare 3 Models - NLP Sentiment Prediction,” Medium, 12-Nov-2019. [Online]. Available: https://towardsdatascience.com/build-and-compare-3-models-nlp-sentiment-prediction-67320979de61. [Accessed: 10-Mar-2021].

[10] A. Patil, “Spam detector using NLP and Random Forest,” Kaggle, 01-Jul-2018. [Online]. Available: https://www.kaggle.com/adityapatil673/spam-detector-using-nlp-and-random-forest. [Accessed: 15-Mar-2021].

[11] L. Zhang, "Lecture 12; July 4, 2019" Presented to APS360: Fundamentals of AI, University of Toronto, Toronto, Ontario, Canada, July 4th, 2019 [Powerpoint slides]. Available: https://www.cs.toronto.edu/~lczhang/360/lec/w08/lec12.pdf. [Accessed on Feb 10, 2021].

[12] J. Howard and S. Ruder, “Universal Language Model Fine-tuning for Text Classification,” 2018. [Online]. Available: https://arxiv.org/pdf/1801.06146.pdf.

[13] “Media Bias Chart 7.0,” Ad Fontes Media, 01-Apr-2021. [Online]. Available: https://www.adfontesmedia.com/. [Accessed: 05-Apr-2021].

[14] A. Thota, N. Lohia, P. Tilak, and S. Ahluwalia, “Fake News Detection: A Deep Learning Approach.” SMU Data Science Review, Southern Methodist University, Dallas, TX , 2018. SMU Data Science Review

[15] “Pretrained models,” Pretrained models - transformers 4.5.0.dev0 documentation. [Online]. Available: https://huggingface.co/transformers/pretrained_models.html. [Accessed: 07-Apr-2021].