MSA-8150 MACHINE LEARNING FINAL PROJECT REPORT

GROUP 12

FAKE NEWS DETECTION

GROUP MEMBERS :

Shashank Gampa (002692264)
Sai Srinath Putta (002714662)
Abinesh G (002704182)

ABSTRACT :

Nowadays, the internet is the primary source of news for most people, and social media platforms like Twitter, Facebook, and YouTube make it effortless to express opinions and share information. However, while it is easy to access news online, it is also easy for false information and rumors to spread. Fake news refers to news that is fabricated and disruptive in nature. The internet is rife with fake news, and anyone with malicious intent can create a false story that can harm an individual's reputation. Despite our confidence in our ability to distinguish between fake and genuine news, it is astonishing how even a small tweet can impact a reputable company's stock and business. Nonetheless, with a reliable dataset of news articles and Natural Language Processing techniques, we can develop a model that can identify fake news. Professionals or ordinary users can utilize this model to obtain credible news from an unreliable source.

INTRODUCTION :

The Alliance for Audited Media (AAM) reports a significant decline of 52% in newspaper consumption in 2020 compared to 2000, indicating that a vast majority of the population now relies on the internet and social media as their primary source of news. As the dissemination of false information can have severe consequences, distinguishing truth from falsehood has become a crucial task. Hence, I aim to develop a predictive model using a recurrent convolutional neural network to detect and eliminate fake news.

An example of how fake news can cause public panic is the circulating rumor about COVID-19 vaccines produced by Pfizer altering the recipient's DNA. The experiment on which this claim was based was conducted on cells extracted from liver cancer tissue, not healthy cells. Additionally, while some viruses like HIV can convert RNA to DNA and integrate it into host cells' genome, coronaviruses are not expected to do so. The rumor spread without proper factual basis, causing public anxiety until it was debunked. Fake news can create unnecessary chaos, making it essential to predict and counter it early on.

Upon building a Recurrent CNN, we aim to analyze the features from the given dataset to determine with high accuracy whether the given article holds any veracity or not. Few reasons why fake news is created/circulated are as follows:

1. Not trusting the media.
2. To make a public entity lose power/weaker.
3. Start violence and thereby disrupt communal harmony.
4. Spread of false science, which could lead to bodily harm and death.

DATASET :

The dataset contains 6335 articles and has 4 columns namely number, title, text, and label. The number column is a serial number column. The title column contains the title of the article. The text field has the entire article description, and the label column tells whether it is a fake or a real article. There are 3171 real articles as compared to 3164 fake articles in the dataset. The below figure shows the density distribution of article lengths of different articles.

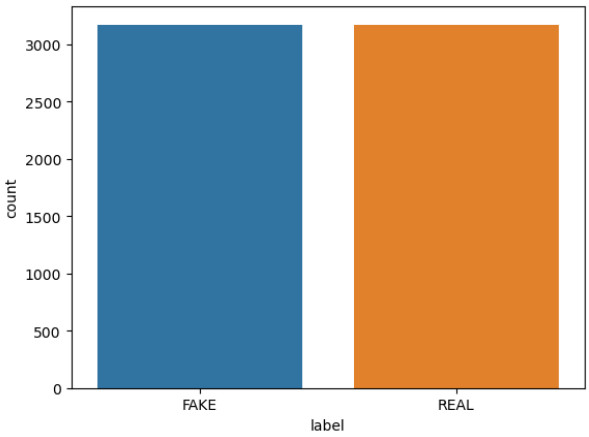

Figure 1: Data distribution in the dataset

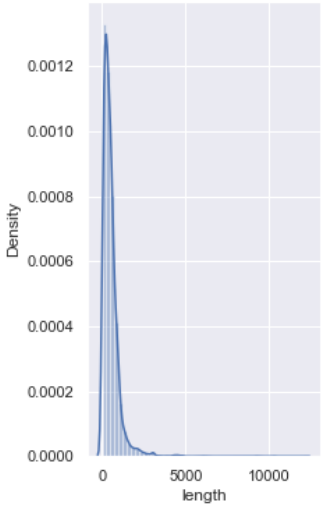

Figure 2: Density distribution of article lengths

DATA PREPROCESSING :

In the data preprocessing steps, we performed four crucial data preprocessing steps to prepare our dataset for training our model. The following steps were :

1. Removal of Punctuations: Used string. Punctuations to remove all sorts of punctuation marks from the text because they do not add any value to our model.
2. Stop word Removal: Stop words are the common words like articles and prepositions which are present in the text. Since they do not provide any context to the model, I removed them using nltk's stopwords package, this package provided the most occurring stop words in English language. So used this to remove the stop words from the dataset.
3. Stemming: Stemming is the process of reducing a word to its lemma form. Lemma is an abstract version of the word that is selected for utterance in the initial stages of speech production. For example – the words running, run and ran can be stemmed as run.
   Doing this will normalize the dataset for the model to train on.
4. Label Processing: In the dataset the labels are termed as FAKE and REAL, therefore mapped all the articles which were fake as 0, and marked all the articles which were real as 1.

PROPOSED METHOD :

In this step, we have implemented a recurrent convolutional neural network to predict fake news over a preprocessed dataset. In this section we will go through how the model was implemented and the algorithm used.

While building the model we used different sequential layers and methods, mostly including:

1. **A tokenizer:** This disintegrates the text into a sequence of words.
2. **A word embedding layer:** This layer creates a m dimensional vector for each unique word and then applies this embedding for the first n words in each news article. This generates a m x n matrix.
3. **Convolutional layers:** I implemented a 1-Dimensional convolutional layer. This layer comprises a series of kernels, each is a low-dimensional vector that computes dot product on the input vector incrementally and produces an output.
4. **Max-pooling layers:** These layers are used to reduce the size of the input by taking maximum value from local region.
5. **Long short-term memory (LSTM) layers followed by dropout layers:** The key benefit of utilizing an LSTM layer is that it enables us to retain only certain information and maintain the sentence context over longer period. This is advantageous because the article's introduction might contain some information that is pertinent to its later sections

as well. The LSTM also aids in helping us determine which information is essential and should be kept, and which is unnecessary and can be deleted.

6. **A final connected layer:** The final layer computes value between 0 and 1 by taking the weighted sum of the previous layer's output and performing a sigmoid activation on them. The produced output from this layer is then rounded to the nearest integer to get 1(Real) or 0(Fake) as the prediction of the model.

This model was trained over 5 epochs to avoid overfitting on the data.

MODELS & RESULTS :

1. This model was designed for larger context retention and has 3 layers with a pool size of 2 and 5 kernels. Additionally, it has 256 LSTM layers. This model gave an accuracy of 90.2%.

```
Fitting Tokenizer...
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 512, 64)           3002176

 conv1d (Conv1D)             (None, 512, 512)          164352

 max_pooling1d (MaxPooling1D  (None, 256, 512)         0
 )

 conv1d_1 (Conv1D)           (None, 256, 1024)         2622464

 max_pooling1d_1 (MaxPooling  (None, 128, 1024)        0
 1D)

 conv1d_2 (Conv1D)           (None, 128, 1536)         7865856

 max_pooling1d_2 (MaxPooling  (None, 64, 1536)         0
 1D)

 lstm (LSTM)                 (None, 64, 128)           852480

 dropout (Dropout)           (None, 64, 128)           0

 lstm_1 (LSTM)               (None, 128)               131584

 dropout_1 (Dropout)         (None, 128)               0

 dense (Dense)               (None, 1)                 129

=================================================================
Total params: 14,639,041
Trainable params: 14,639,041
Non-trainable params: 0
```

Figure 3: Model 1 structure

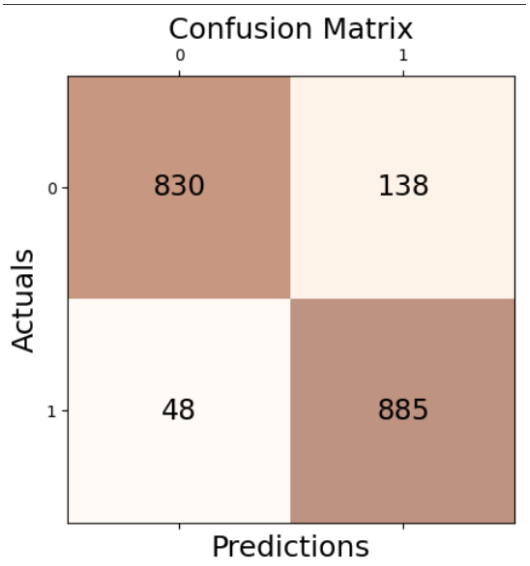

Figure 4: Model 1 Confusion matrix

	precision	recall	f1-score	support
0	0.91	0.88	0.89	968
1	0.88	0.91	0.89	933
accuracy			0.89	1901
macro avg	0.89	0.89	0.89	1901
weighted avg	0.89	0.89	0.89	1901

Figure 5: Model 1 Accuracy Metrics

2. This model was designed for larger context retention and has 3 layers with a pool size of 2 and 5 kernels. Additionally, it has 256 LSTM layers. This model gave an accuracy of 90.2%.

	precision	recall	f1-score	support
0	0.92	0.90	0.91	968
1	0.90	0.92	0.91	933
accuracy			0.91	1901
macro avg	0.91	0.91	0.91	1901
weighted avg	0.91	0.91	0.91	1901

Figure 6: Model 2 Accuracy Metrics

```
Fitting Tokenizer...
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 512, 64)           3002176

 conv1d_3 (Conv1D)           (None, 512, 256)          49408

 max_pooling1d_3 (MaxPooling  (None, 256, 256)         0
 1D)

 conv1d_4 (Conv1D)           (None, 256, 512)          393728

 max_pooling1d_4 (MaxPooling  (None, 128, 512)         0
 1D)

 conv1d_5 (Conv1D)           (None, 128, 768)          1180416

 max_pooling1d_5 (MaxPooling  (None, 64, 768)          0
 1D)

 lstm_2 (LSTM)               (None, 64, 128)           459264

 dropout_2 (Dropout)         (None, 64, 128)           0

 lstm_3 (LSTM)               (None, 128)               131584

 dropout_3 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 1)                 129

=================================================================
Total params: 5,216,705
Trainable params: 5,216,705
Non-trainable params: 0
_________________________________________________________________
```

Figure 7: Model 2 Structure

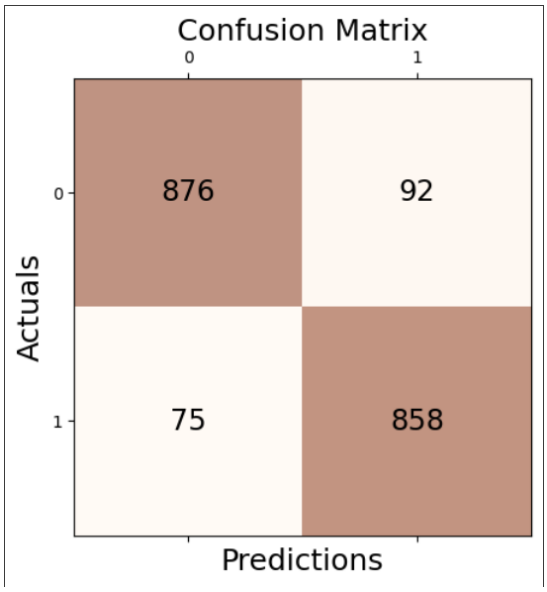

Figure 8: Model 2 Confusion Matrix

3. While creating this model we increased the model depth by adding an additional layer i.e., using 4 layers with kernel size as 5 with 128 LSTM layers and pool size of 2. The accuracy of this model came to 90%.

```
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, 512, 64)           3002176

 conv1d_6 (Conv1D)           (None, 512, 256)          82176

 max_pooling1d_6 (MaxPooling  (None, 256, 256)         0
 1D)

 conv1d_7 (Conv1D)           (None, 256, 512)          655872

 max_pooling1d_7 (MaxPooling  (None, 128, 512)         0
 1D)

 conv1d_8 (Conv1D)           (None, 128, 768)          1966848

 max_pooling1d_8 (MaxPooling  (None, 64, 768)          0
 1D)

 conv1d_9 (Conv1D)           (None, 64, 1024)          3933184

 max_pooling1d_9 (MaxPooling  (None, 32, 1024)         0
 1D)

 lstm_4 (LSTM)               (None, 32, 128)           590336

 dropout_4 (Dropout)         (None, 32, 128)           0

 lstm_5 (LSTM)               (None, 128)               131584

 dropout_5 (Dropout)         (None, 128)               0

 dense_2 (Dense)             (None, 1)                 129

=================================================================
Total params: 10,362,305
Trainable params: 10,362,305
Non-trainable params: 0
_________________________________________________________________
```

Figure 9: Model 3 Structure

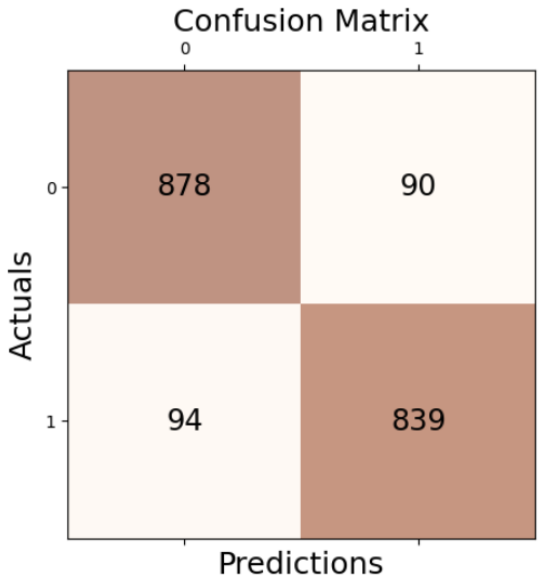

Figure 10: Model 3 Confusion Matrix

	precision	recall	f1-score	support
0	0.90	0.91	0.91	968
1	0.90	0.90	0.90	933
accuracy			0.90	1901
macro avg	0.90	0.90	0.90	1901
weighted avg	0.90	0.90	0.90	1901

Figure 11: Model 3 Accuracy

4. Hyperparameter tuned model configuration – 3 layers, kernel size of 5 with 128 LSTM layers and a pool size of 2. An accuracy of 94% was achieved with the above parameters.

```
Fitting Tokenizer...
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_6 (Embedding)     (None, 512, 64)           3002176

 conv1d_19 (Conv1D)          (None, 512, 512)          229888

 max_pooling1d_19 (MaxPoolin  (None, 256, 512)         0
 g1D)

 conv1d_20 (Conv1D)          (None, 256, 1024)         3671040

 max_pooling1d_20 (MaxPoolin  (None, 128, 1024)        0
 g1D)

 conv1d_21 (Conv1D)          (None, 128, 1536)         11011584

 max_pooling1d_21 (MaxPoolin  (None, 64, 1536)         0
 g1D)

 lstm_12 (LSTM)              (None, 64, 128)           852480

 dropout_12 (Dropout)        (None, 64, 128)           0

 lstm_13 (LSTM)              (None, 128)               131584

 dropout_13 (Dropout)        (None, 128)               0

 dense_6 (Dense)             (None, 1)                 129

=================================================================
Total params: 18,898,881
Trainable params: 18,898,881
Non-trainable params: 0
```

Figure 12: Model 4 Structure

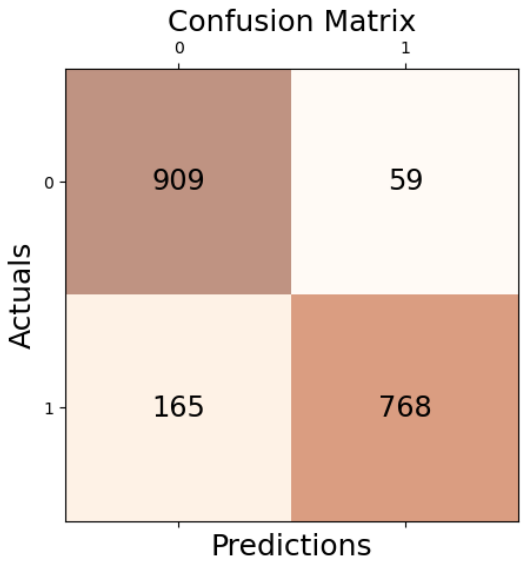

Figure 13: Model 4 Confusion Matrix

	precision	recall	f1-score	support
0	0.93	0.84	0.89	968
1	0.84	0.82	0.87	933
accuracy			0.94	1901
macro avg	0.93	0.93	0.93	1901
weighted avg	0.95	0.93	0.93	1901

Figure 14: Model 4 Accuracy

CONCLUSION :

MODEL	ACCURACY
LSTM MODEL 1	89%
LSTM MODEL 2	91%
LSTM MODEL 3	90%
LSTM MODEL 4	94%

From these results we can conclude that the model with hyperparameter-tuned configuration gives the best accuracy.