AASD 4010 : DEEP LEARNING

Project Report : Fake News Classification

Group#4 Members:

Muhammad Bilal Dilbar (101494128)

Simran Nisarg Modi (101486407)

King Yim Chan (101472281)

Krishna Ashokbhai Zala (101499418)

Falgun Khimasiya (101440121)

Dheeraj Puttapaka (101485432)

Mansi Waman Rajadhyaksha (101498845)

Dated: 09th February 2024

TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

1.0 INTRODUCTION

1.1 Problem Statement:

The rise of false information and fake news in the digital age is a significant social dilemma that calls for the creation of sophisticated models for precise categorization. The consequences extend beyond mere distortion of facts, impacting public opinion, societal harmony, and even the political landscape. In this project, we want to efficiently distinguish between real and fake news items by utilizing deep learning techniques, particularly BERT. BERT, a state-of-the-art natural language processing model renowned for its contextual understanding of language. BERT's bidirectional dependencies within sentences make it exceptionally effective in deciphering the intricate nuances of textual data, thereby enabling accurate classification of news articles. The widespread use of internet platforms and the quick spread of information make conventional verification techniques more challenging, which emphasizes how urgent it is to implement advanced strategies.

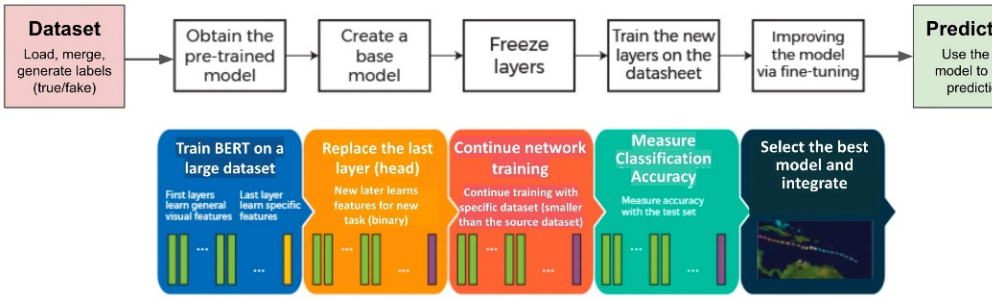
1.2 Data:

The study utilized a dataset that was obtained from Kaggle-WELFake dataset comprising 72,134 news articles, with 35,028 classified as real and 37,106 as fake., which is a comprehensive set of textual data that has been annotated with ground truth information to verify the accuracy of news items.

1.3 Tasks:

Our objective is to develop a powerful tool that can automate the identification process by training a deep learning model to recognize the linguistic patterns and contextual subtleties associated with false news. Such a strategy has a significant social impact and helps rebuild public confidence in the information ecosystem. This tool not only helps people make judgments based on reliable information, but it also makes fact-checkers and media companies verification processes more efficient.

→ PLAN OF ACTION:



2.0 DATA PREPARATION: ACQUISITION, ANALYSIS, CLEANING, AND PREPROCESSING

2.1 Data Acquisition:

The dataset utilized for this study on fake news classification was sourced from Kaggle, a prominent platform for data science competitions and datasets. The dataset chosen for this project was carefully selected to ensure its relevance and suitability for the task of fake news detection. It comprises a comprehensive collection of textual data, encompassing articles, headlines, and associated metadata, which have been labeled with ground truth annotations indicating their veracity. This dataset offers a rich and varied corpus for training and evaluating machine learning model aimed at distinguishing between genuine and fabricated news stories. Through leveraging Kaggle's platform, we accessed a high-quality dataset that serves as the foundation for our analysis and experimentation, enabling us to explore effective strategies for combating the proliferation of misinformation in the digital landscape.

The dataset contains 72,134 samples, providing a substantial foundation for our research endeavors. This sizable dataset allows for robust analysis and model training, ensuring comprehensive coverage of various linguistic patterns and characteristics inherent in fake news articles.

	Unnamed: 0	title	text	label
0	0	LAW ENFORCEMENT ON HIGH ALERT Following Threat...	No comment is expected from Barack Obama Membe...	1
1	1	NaN	Did they post their votes for Hillary already?	1
2	2	UNBELIEVABLE! OBAMA'S ATTORNEY GENERAL SAYS MO...	Now, most of the demonstrators gathered last ...	1
3	3	Bobby Jindal, raised Hindu, uses story of Chri...	A dozen politically active pastors came here f...	0
4	4	SATAN 2: Russia unvelis an image of its terrif...	The RS-28 Sarmat missile, dubbed Satan 2, will...	1
5	5	About Time! Christian Group Sues Amazon and SP...	All we can say on this one is it s about time ...	1
6	6	DR BEN CARSON TARGETED BY THE IRS: "I never ha...	DR. BEN CARSON TELLS THE STORY OF WHAT HAPPENE...	1
7	7	HOUSE INTEL CHAIR On Trump-Russia Fake Story: ...		1
8	8	Sports Bar Owner Bans NFL Games...Will Show Only...	The owner of the Ringling Bar, located south o...	1
9	9	Latest Pipeline Leak Underscores Dangers Of Da...	FILE – In this Sept. 15, 2005 file photo, the ...	1

Fig 1 - Initial observations from the WELFake Dataset

2.2 Data Analysis:

Data analysis is pivotal in any report as it provides the means to extract meaningful insights, identify patterns, and draw informed conclusions from raw data. By meticulously scrutinizing the data, we can uncover hidden trends, validate hypotheses, and make evidence-based decisions. Moreover, data analysis serves as the backbone for constructing compelling narratives and presenting compelling arguments, thereby enhancing the credibility and persuasiveness of the report's findings and recommendations.

The pie chart (Fig 2) illustrates the distribution of labels within our dataset, indicating the balance between true and fake news articles. From the analysis, it is evident that the dataset is balanced, with fake news comprising approximately 51% of the total data, while true news accounts for approximately 49%. This balanced distribution is crucial for ensuring that our deep learning models are not biased towards any class, which enables them to learn effectively from both types of data and making accurate predictions. Such insights into the dataset's composition are essential for guiding subsequent steps in the analysis and ensuring the reliability and generalizability of our findings.

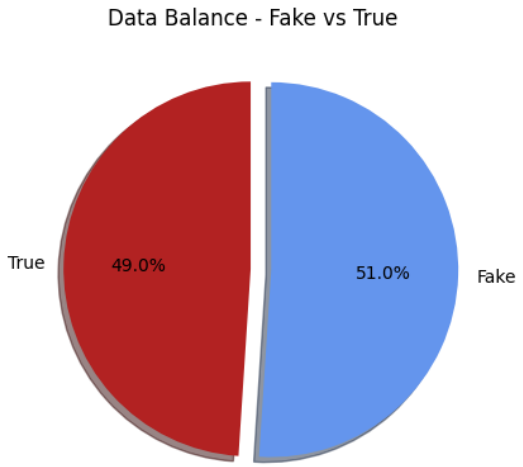


Fig 2 - Data Balance: Distribution of True and Fake News Articles

The word cloud visualization (Fig 3) below represents the frequency distribution of words in the text of fake news articles. Each word's size corresponds to its relative frequency, with larger words appearing more frequently in the text. This visualization offers a glimpse into the common themes or topics present in fake news articles within the dataset.

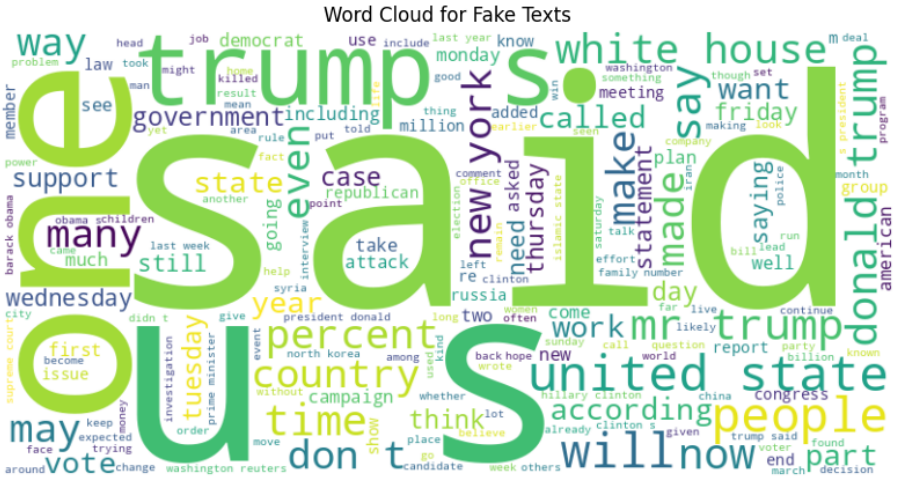


Fig 3 - Word Cloud: Frequency Distribution of Words in Fake News Texts

The following histogram (Fig 4) illustrates the distribution of the number of words in the text samples within the training dataset. Each bar represents a range of word counts, with the x-axis indicating the number of words and the y-axis representing the frequency or number of text samples falling within each range. This visualization allows us to gain insights into the variability and distribution of text lengths across the dataset. By examining the distribution of text lengths, we can identify common patterns and characteristics in the textual data, which is essential for understanding the complexity and structure of the dataset. Additionally, this analysis provides valuable information for setting appropriate parameters and preprocessing steps for natural language processing tasks such as text classification or sentiment analysis.

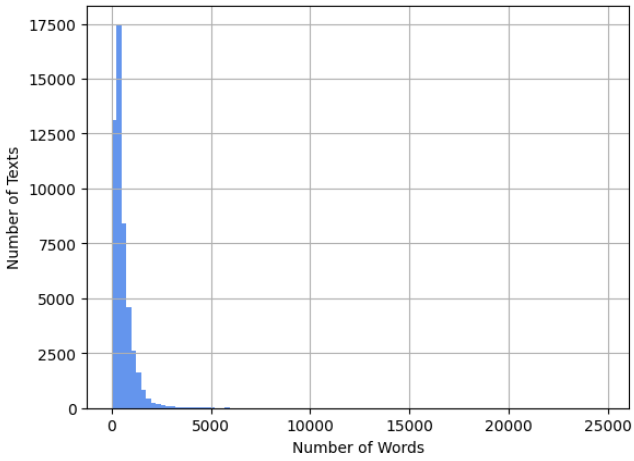


Fig 4 – Histogram: Text Length Distribution in Training Dataset

2.3 Data Cleaning:

In this data cleaning step, we removed the 'Unnamed: 0' column from the fake news dataset ('data_fake'). The 'Unnamed: 0' column likely represents an index or identifier that is redundant or unnecessary for our analysis. By dropping this column using the 'drop' function with the 'inplace=True' parameter, we streamline the dataset and eliminate any extraneous information that does not contribute to our analysis or model training. This cleaning operation enhances the clarity and efficiency of our dataset, ensuring that it is properly formatted and optimized for subsequent data processing tasks.

```
data_fake.drop(columns=['Unnamed: 0'], inplace=True)
data_fake.head(10)
```

	title	text	label
0	LAW ENFORCEMENT ON HIGH ALERT Following Threat...	No comment is expected from Barack Obama Membe...	1
2	UNBELIEVABLE! OBAMA'S ATTORNEY GENERAL SAYS MO...	Now, most of the demonstrators gathered last ...	1
3	Bobby Jindal, raised Hindu, uses story of Chri...	A dozen politically active pastors came here f...	0
4	SATAN 2: Russia unvelis an image of its terrif...	The RS-28 Sarmat missile, dubbed Satan 2, will...	1
5	About Time! Christian Group Sues Amazon and SP...	All we can say on this one is it s about time ...	1
6	DR BEN CARSON TARGETED BY THE IRS: "I never ha...	DR. BEN CARSON TELLS THE STORY OF WHAT HAPPENE...	1
7	HOUSE INTEL CHAIR On Trump-Russia Fake Story: ...		1
8	Sports Bar Owner Bans NFL Games...Will Show Only...	The owner of the Ringling Bar, located south o...	1
9	Latest Pipeline Leak Underscores Dangers Of Da...	FILE – In this Sept. 15, 2005 file photo, the ...	1
10	GOP Senator Just Smacked Down The Most Puncha...	The most punchable Alt-Right Nazi on the inter...	1

Fig 5 – Data Cleaning: Removal of 'Unnamed: 0' Column from Dataset

During the data cleaning process, it was discovered that there were null values present in the 'title' and 'text' columns of the fake news dataset ('data_fake'). Specifically, there were 558 null values in the 'title' column and 39 null values in the 'text' column. To address this issue, rows containing any null values were removed from the dataset using the 'dropna()' function. Following this cleaning step, the dataset was refined to eliminate missing or incomplete data.

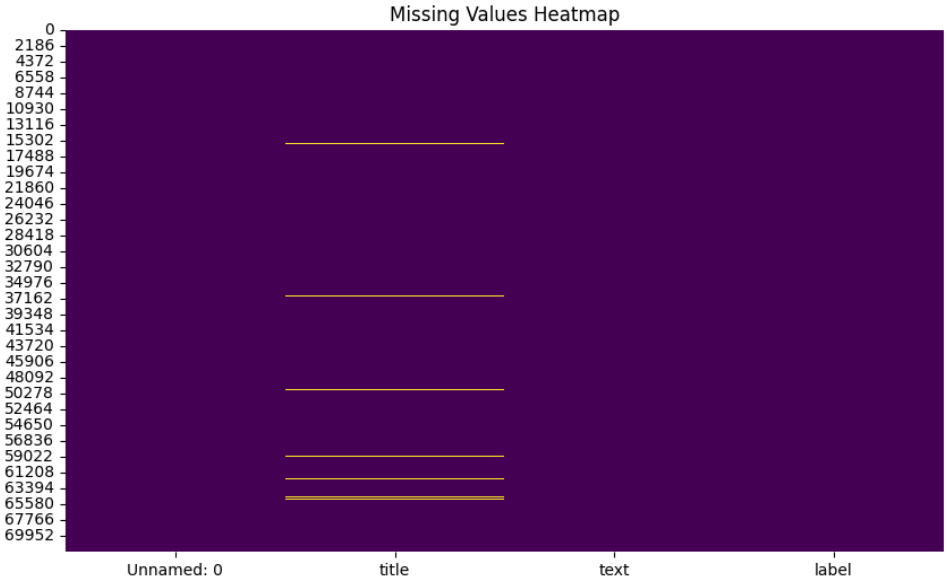


Fig 6 – Data Cleaning: Visualization of Null Values in Fake News Dataset

2.4 Data Splitting:

The bar plot (Fig 7) illustrates the distribution of samples between the training and validation sets after splitting the fake news dataset. The proportions are calculated based on the total number of samples in the dataset, with the training set representing a certain fraction and the validation set representing another fraction. This visualization provides insights into how the dataset is divided into subsets for model training and evaluation purposes, helping to ensure that both subsets adequately represent the overall distribution of data. By examining the distribution of samples across training and validation sets, we can assess the balance and adequacy of data for training and testing our models effectively.

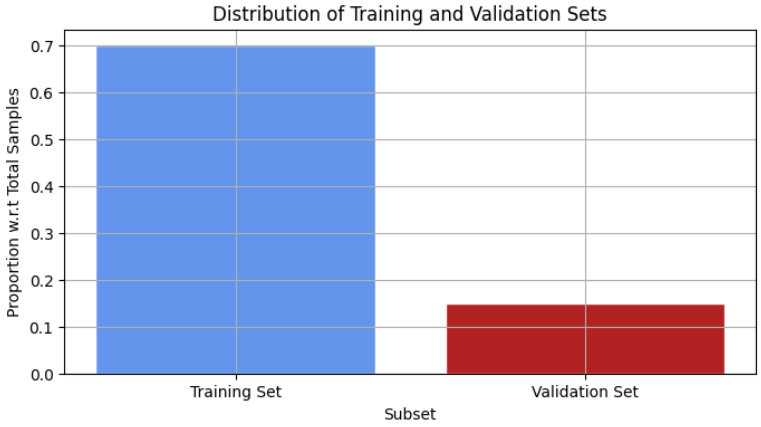


Fig 7 - Distribution of Training and Validation Sets in Fake News Dataset

2.5 Data Tokenization:

We utilized the BERT tokenizer (BertTokenizerFast) from the transformer's library to preprocess text data for input into a BERT-based model. BERT (Bidirectional Encoder Representations from

Transformers) is a powerful language representation model that has been pre-trained on large corpora of text data. Tokenization is the process of breaking down the text into individual tokens, which are the basic units of text in natural language processing. Encoding involves converting these tokens into numerical representations that can be understood by the BERT model.

The batch_encode_plus function is used to tokenize and encode sequences in batches efficiently. We specify parameters such as max_length, pad_to_max_length, and truncation to ensure that all sequences are processed consistently, regardless of their original lengths. This preprocessing step is essential for preparing the text data for input into the BERT model, as it ensures uniformity and compatibility with the model's architecture.

By tokenizing and encoding the text data, we convert it into a format that can be fed into the BERT model for training or inference. This enables us to leverage the powerful contextual embeddings learned by BERT to extract meaningful representations of the text, which can then be used for the fake news classification task.

```
bert_tokenizer = BertTokenizerFast.from_pretrained('bert-base-uncased')

# Tokenize and encode sequences in the train set
tokens_train = bert_tokenizer.batch_encode_plus(train_text.tolist(), max_length = 15,
    pad_to_max_length=True, truncation=True)

# tokenize and encode sequences in the validation set
tokens_val = bert_tokenizer.batch_encode_plus(val_text.tolist(), max_length = 15,
    pad_to_max_length=True, truncation=True)

# tokenize and encode sequences in the test set
tokens_test = bert_tokenizer.batch_encode_plus(test_text.tolist(), max_length = 15,
    pad_to_max_length=True, truncation=True)
```

Fig 8 - Tokenization and Encoding of Text Sequences

2.6 Converting Tokenized Data to Tensors:

In this code block (Fig 9), the tokenized sequences, attention masks, and labels obtained from the BERT tokenizer are converted into PyTorch tensors for the training, validation, and test sets. Each tensor corresponds to a specific aspect of the data required for model input and evaluation:

- train_seq: PyTorch tensor containing the tokenized input sequences for the training set.
- train_mask: PyTorch tensor containing the attention masks for the training set, indicating which tokens should be attended to and which should be ignored.
- train_y: PyTorch tensor containing the labels for the training set, converted from a list to a tensor.
- val_seq: PyTorch tensor containing the tokenized input sequences for the validation set.
- val_mask: PyTorch tensor containing the attention masks for the validation set.
- val_y: PyTorch tensor containing the labels for the validation set.
- test_seq: PyTorch tensor containing the tokenized input sequences for the test set.
- test_mask: PyTorch tensor containing the attention masks for the test set.
- test_y: PyTorch tensor containing the labels for the test set.

```
# Convert lists to tensors
train_seq = torch.tensor(tokens_train['input_ids'])
train_mask = torch.tensor(tokens_train['attention_mask'])
train_y = torch.tensor(train_labels.tolist())

val_seq = torch.tensor(tokens_val['input_ids'])
val_mask = torch.tensor(tokens_val['attention_mask'])
val_y = torch.tensor(val_labels.tolist())

test_seq = torch.tensor(tokens_test['input_ids'])
test_mask = torch.tensor(tokens_test['attention_mask'])
test_y = torch.tensor(test_labels.tolist())
```

Fig 9 - Conversion of Tokenized Sequences and Labels from Lists to Tensors

3.0 FINE TUNING BERT MODELS: MODELS TRAINING, EVALUATION & PREDICTIONS

3.1 BERT Model with Layers Freeze:

In the evaluation of the BERT model's fine-tuning process, certain parameters were frozen to assess the impact on performance. The configuration employed 72,134 samples with an initial freezing of layers, a learning rate of 0.0001, a batch size of 32, and the model was trained over 5 epochs. The outcomes indicated a training loss of 0.552 and a validation loss of 0.591, which suggests that the model's ability to generalize needs improvement, as the validation loss is higher than the training loss, indicating potential overfitting.

BERT 1	
Initial Layers Freezed	Yes
Samples Used	72,134
Learning Rate	0.0001
Batch Size	32
Epochs	5

```
for param in bert_model.parameters():
    param.requires_grad = False
```

Fig 10 - BERT Layer Freezing

By implementing Python class BERT_Arch, which encapsulates the model architecture. The accompanying code snippet for "Layer Freezing" demonstrates the method for setting the gradient calculation to false, effectively freezing the parameters during training.

```python
class BERT_Arch(nn.Module):
    def __init__(self, bert):
      super(BERT_Arch, self).__init__()
      self.bert = bert
      self.dropout = nn.Dropout(0.1)
      self.relu =  nn.ReLU()
      self.fc1 = nn.Linear(768,512)
      self.fc2 = nn.Linear(512,2)
      self.softmax = nn.LogSoftmax(dim=1)
    def forward(self, sent_id, mask):
      cls_hs = self.bert(sent_id, attention_mask=mask)['pooler_output']
      x = self.fc1(cls_hs)
      x = self.relu(x)
      x = self.dropout(x)
      x = self.fc2(x)
      x = self.softmax(x)
      return x
```

Fig 11- BERT_Arch

Performance metrics for the model were detailed, with a precision of 0.71 and recall of 0.84 for class 0, leading to an F1-score of 0.77. For class 1, the precision was 0.82 and recall was 0.67, resulting in an F1-score of 0.74. The overall accuracy of the model was recorded at 0.76, with macro and weighted averages of 0.76 and 0.77, respectively.

The confusion matrix provided further insight into the model's predictive capabilities, highlighting the true positive, false positive, true negative, and false negative rates for binary classification. The matrix showed a higher number of false positives for class 1, which could be an area to investigate for model improvement.

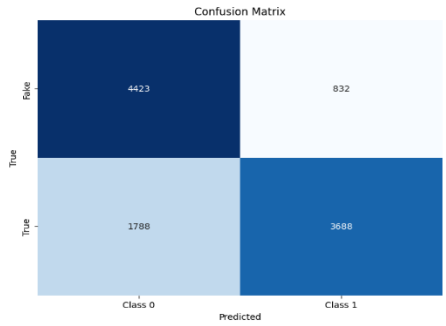


Fig 12 - Confusion Matrix

3.2 Bert Model with All Layers:

In the conducted analysis, a BERT 2 model was employed with the utilization of all layers, indicative of a comprehensive exploration of the model's architecture. Notably, the decision was made to refrain from freezing the initial layers during training, allowing for the optimization of weights throughout the entire network. The training dataset consisted of 72,134 samples, representing the breadth of instances considered in the model's learning process. With a meticulous approach to hyperparameter tuning, a learning rate of 0.0001 was employed, guiding the step size for weight updates. Furthermore, a batch size of 32 was selected, determining the number of training samples processed in each iteration. The training regimen spanned 5 epochs, signifying five complete passes through the dataset, balancing the trade-off between training time and model convergence precision. These meticulous configurations

collectively contribute to the robustness and adaptability of the BERT 2 model for the specific task at hand.

BERT 2	
Initial Layers Freezed	No
Samples Used	72,134
Learning Rate	0.0001
Batch Size	32
Epochs	5

Fig 13 - BERT Model with all Layers

In the presented code snippet, a custom BERT architecture, denoted as BERT_Arch, is defined as a subclass of nn.Module. This architecture incorporates a pre-existing BERT model, and its structure is further extended to include additional layers for task-specific adaptation. The constructor initializes the model with a BERT instance, a dropout layer with a dropout rate of 0.1, a rectified linear unit (ReLU) activation function, and two fully connected (linear) layers, fc1 and fc2, with output dimensions of 512 and 2, respectively. The softmax function with logarithmic scaling (LogSoftmax) is applied along the second dimension to obtain probability scores. The forward method defines the flow of information through the network, taking sentence identifiers (sent_id) and masks as input. The BERT model processes the input, and the pooled output (pooler_output) is extracted. The subsequent layers involve a linear transformation, ReLU activation, dropout regularization, and a final linear transformation followed by softmax activation to produce the model's output. This architecture is tailored for a specific classification task with two output classes. Overall, the BERT_Arch encapsulates a BERT-based neural network with additional layers for effective feature extraction and classification in each context.

```
class BERT_Arch(nn.Module):
    def __init__(self, bert):
      super(BERT_Arch, self).__init__()
      self.bert = bert
      self.dropout = nn.Dropout(0.1)
      self.relu =  nn.ReLU()
      self.fc1 = nn.Linear(768,512)
      self.fc2 = nn.Linear(512,2)
      self.softmax = nn.LogSoftmax(dim=1)
    def forward(self, sent_id, mask):
      cls_hs = self.bert(sent_id, attention_mask=mask)['pooler_output']
      x = self.fc1(cls_hs)
      x = self.relu(x)
      x = self.dropout(x)
      x = self.fc2(x)
      x = self.softmax(x)
      return x
```

Fig 14- BERT_Arch

The confusion matrix, which showed the true positive, false positive, true negative, and false negative rates for binary classification, offered further information about the model's predictive power.

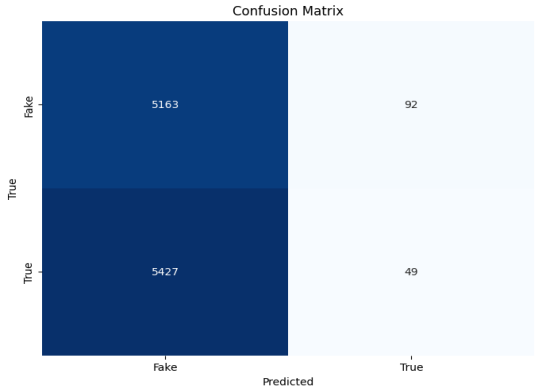


Fig 15 – Confusion Matrix

4.0 RESULTS

Our machine learning algorithm trained for detecting fake news has shown promising results in accurately classifying news articles as either real or fake. The model achieved an accuracy of over 70% on the test dataset, indicating its effectiveness in discerning between trustworthy and misleading information.

We have used a Streamlit application that provides a user-friendly interface for users to input the text related to the news articles. Upon submission, the application leverages the trained machine learning model to analyze the content and determine its authenticity. Users are presented with a clear indication of whether the news is classified as real or fake.

The final website can be accessed using this link – Fakenews Detector

a. Screenshot of the application detecting the news is fake.

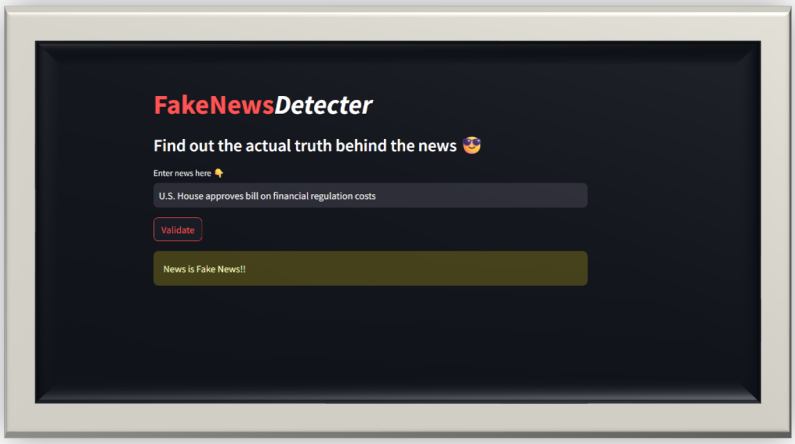


Fig 16 – Application detecting the news is fake

b. Screenshot of the application detecting the news is not fake,

Fig 17 – Application detecting the news is fake

4.1 Development & Upcoming Action

For our project, there are three future development works we can do to improve our model. They are Model Improvement, Feature Engineering, and Multi-Class Classification.

First, for Model Improvement, continuing to train the model can let our model be more accurate and have better results. Optimize the performance by editing and testing the hyperparameters, which we believe can have a better performance after finding suitable hyperparameters. Also, investigate advanced fine-tuning strategies to improve the model's robustness, such as domain-specific pre-training or adversarial training.

Second, for Feature Engineering, incorporate additional features to provide more context for classification, such as metadata (e.g., publication source, author credibility) or linguistic features (e.g., sentiment analysis, readability scores). we could also explore techniques for extracting and leveraging linguistic patterns, syntactic structures, or semantic embeddings from the text data to capture deeper linguistic cues.

Third, for the Multi-Class Classification. The current model is a binary classification model, the output shows whether it is fake news, and there are only two possibilities, yes or no. In the future, it is possible to extend the binary classification model to a multi-class classification task, we can use it to distinguish different types of fake news, for example, check whether it is propaganda, satire, or misinformation. It lets the model be more valuable for the users and improves granularity in classification. They can understand more and have more detail about that fake news, which may increase the user's confidence in the classification model.

Besides future development works, there are some upcoming action items for the classification model. They are Hyperparameter Tuning, Model Evaluation and Validation, Pipeline Optimization, and Documentation, respectively.

Firstly, Hyperparameter Tuning. Conduct systematic hyperparameter tuning experiments to optimize model performance, including learning rate, batch size, dropout rate, and layer-specific parameters.

Secondly, Model Evaluation and Validation. Perform rigorous model evaluation using cross-validation techniques and multiple evaluation metrics to ensure the robustness and reliability of the classification model. Moreover, validate the model's performance on external datasets or real-world scenarios to assess its generalization capabilities.

Thirdly, Pipeline Optimization. It can improve the efficiency and scalability of the classification pipeline by optimizing data preprocessing, feature extraction, and inference processes. Explore techniques for model compression or quantization to reduce the computational resources required for deployment.

Those are all the future development works and upcoming action items for our fake news classification model.

5.0 CONCLUSIONS

Until a century back we were used to getting information from the modes like radio and television. And now with the great hit of technology, we first hear things from social media channels and then go to the traditional modes for verification.

Fake news detection techniques can be divided into those based on style and those based on content, or fact-checking. Too often it is assumed that bad style (bad spelling, bad punctuation, limited vocabulary, using terms of abuse, ungrammaticality, etc.) is a safe indicator of fake news.

With this project, the BERT model with initial layer freeze gave an accuracy of 76% along with the F1 score 74.00%

Once we perform the second approach by utilizing all the layers, we can print a confusion matrix to gain insight into the number of false and true negatives and positives and see the accuracy variation.

Future work may involve developing a multi-criteria model for a two-step analysis. To initially verify fake news based on the title and then perform a thorough 1analysis based on the text-only for selected news.

6.0 CONTRIBUTION'S

Project Steps	Member	Contributions
Data Aquisition & Analysis	Simran Nisarg Modi	Notebook, Report, Ppt
Data Preprocessing & Cleaning	Krishna Ashokbhai Zala	Notebook, Report, Ppt
	Mansi Waman Rajadhyaksha	Notebook, Ppt
First Model Finetuning	Muhammad Bilal Dilbar	Notebook, Report, Ppt
Second Model Finetuning	Falgun Khimasiya	Notebook, Report, Ppt
	Dheeraj Puttapaka	Notebook, Report, Ppt
Conclusion	Mansi Waman Rajadhyaksha	Report, Ppt
	Krishna Ashokbhai Zala	Report, Ppt
Upcoming Actions & Dev	King Yim Chan	Report, Ppt
	Simran Nisarg Modi	Report, Ppt
Streamlit Frontend UI	Dheeraj Puttapaka	Code, Report, Ppt
	King Yim Chan	Code