Fake News Detection

Santhosh Reddy Chilaka
Department of Computer Science
University at Buffalo
schilaka@buffalo.edu

Sri Harsha Vardhan Raju Namburi
Department of Computer Science
University at Buffalo
namburi@buffalo.edu

Harshavardhan Sivadanam
Department of Computer Science
University at Buffalo
hsivadan@buffalo.edu

Abstract

In the digital age where fake news proliferates rapidly, our project explores the use of advanced NLP techniques for effective detection and differentiation of true vs. false narratives. With the rise of social media and the widespread dissemination of information, the spread of fake news has become a significant issue in society. Fake news often has characteristics such as sensational headlines, emotionally charged language, lack of credible sources, and confirmation bias. Our project aims to showcase the effectiveness and efficiency of both models in fake news detection, providing insights into their performance and potential applications in automated fact-checking. We harness the strengths of LSTM (Long Short-Term Memory) and Transformer models, renowned for their ability to process sequential data and large datasets, respectively, in the context of complex text analysis. We were able to get around 62% accuracy for our models.

1 LIAR Dataset

1.1 Dataset description

The LIAR dataset comprises thousands of labeled statements from `PolitiFact.com`, a political fact-checking website. Each statement in the dataset is labeled with a truthfulness rating. The dataset contains 12,800+ short statements drawn from various contexts and topics within the political domain. Pants fire, false, mostly true, half true, mainly true, and true are its six categories of labeling. The statements mainly cover the years 2007 through 2016. Republicans and Democrats coexist among the speakers, and each one has a wealth of metadata, including past totals for the number of erroneous claims made by them. Also includes additional metadata like the speaker's credit history, the context of the statement, and the speaker's job title which provides richer information for the model to analyze the veracity of statements. We divided the dataset into 80% training, and 20% for validation and testing.

1.2 Data Analysis

- We found out there are 3827 unique subjects in the dataset, and these subjects are not very useful for the detection of fake news. We displayed just the top 20 subjects based on count just for visualization, It shows the variety and frequency of topics addressed, highlighting the dataset's diversity in terms of political subjects. This diversity is crucial for training models to recognize fake news across a broad spectrum of political discourse.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

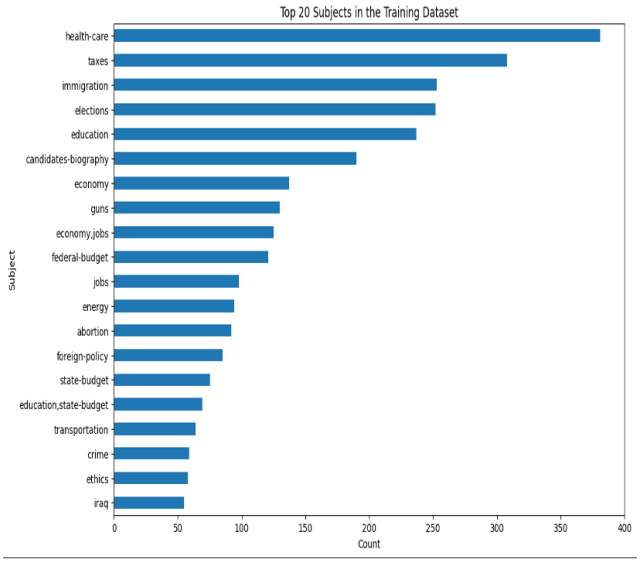
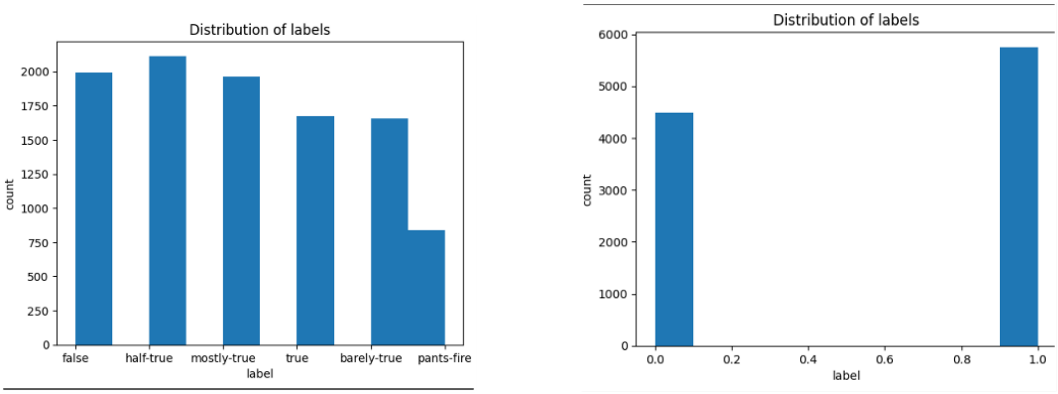


Figure 1: Top 20 Subjects

- We wanted to know about the distribution of labels so that we can do any type of data augmentation technique to solve the case of class imbalance. The labels represent the truthfulness ratings of the statements, ranging from true to false, including various degrees of truth. This distribution is key to understanding the inherent class imbalances in the dataset.

- Labels are converted into binary values(true and mostly true labels as true, remaining all labels as fake). It seems like a binary classification task where the labels are transformed into binary values, such as true and fake. This step is crucial for training a binary classification model.



(a) (b)

Figure 2: Labels before and after encoding

- A visual representation of text data where the size of each word indicates its frequency or importance within a given text corpus. In this case, the corpus appears to be related to political topics, with words like "federal budget," "health care," "elections," and "education" prominently displayed. These larger words suggest that these topics are more frequently mentioned or hold more significance in the dataset from which the word cloud was generated.

Figure 3: Word Cloud of true and fake news

1.3 Data Pre-processing

Data Loading: You load the dataset from three separate TSV (Tab-Separated Values) files: training data, test data, and validation data. These files likely contain information about news articles, including the text of the articles and corresponding labels.

Text Cleaning: Text cleaning is performed to preprocess the textual data and remove unnecessary information.

- Regular expressions are used to match and remove specific patterns in the text. This can include removing special characters, digits, or any other patterns that are not relevant to the analysis.
- Patterns matching URLs are removed from the text.
- If the text contains HTML tags, they are removed to extract only the plain text.
- Punctuation is often removed to simplify the text and focus on the words.
- Common stop words are removed. These words are often frequent but carry little semantic meaning.
- Accents in words are normalized using the unidecode library. This helps handle variations in accent marks.

Tokenization: Tokenization is the process of converting a sequence of text into individual units called tokens.

- Bert Tokenizer from the transformers library is used. This tokenizer is specifically designed for models like BERT and handles WordPiece tokenization, which is beneficial for capturing subword information.
- TF-IDF (Term Frequency-Inverse Document Frequency) vectorizers are also used for tokenization. TF-IDF represents the importance of each word in the context of the entire dataset.
- The tokenized sentences are then padded or truncated to a maximum length of ‘128‘.

Label Encoding: Labels are converted into binary values(true and mostly true labels as true, remaining all labels as fake). It seems like a binary classification task where the labels are transformed into binary values, such as true and fake. This step is crucial for training a binary classification model.

Overall, these preprocessing steps aim to clean and transform the raw text data into a format suitable for training machine learning models. They help in reducing noise, handling variations, and extracting meaningful features from the text, ultimately preparing the data for the model training process.

2 Model Description

2.1 Bi-LSTM

The architecture presented in the diagram provides a structured approach to natural language processing tasks, specifically designed for classifying text inputs. Initially, the raw text is subjected to a

series of preprocessing steps: the data is loaded, cleansed of noise such as unwanted characters and spaces, and then tokenized using BERT's sophisticated tokenization process. This step is essential for breaking down the text into a form that a neural network can work with. Following tokenization, label encoding is applied to translate categorical labels into numerical values, preparing the data for the machine learning model.

Once preprocessed, the text data enters the deep learning phase of the pipeline. An embedding layer first converts the tokenized text into dense vectors that are capable of capturing complex word relationships and semantic meanings. These vectors then pass through multiple LSTM layers, which are specifically chosen for their ability to process sequences and capture temporal dependencies. The LSTM layers, potentially numerous as indicated by "N layers", allow the model to learn from the context within the text over different ranges, from immediate succession to long-range patterns.

Finally, the outputs from the LSTM layers are concatenated to form a holistic feature set, which is then refined through a dropout layer to prevent overfitting a common challenge in deep learning models. The concatenated and regularized features feed into a fully connected (FC) layer, which serves to distill these features into a format that can be used for classification. The process concludes with a sigmoid function in the output layer, providing a probability score between 0 and 1. This score is the model's prediction of the likelihood that the input text belongs to a particular class, thus achieving the classification objective.

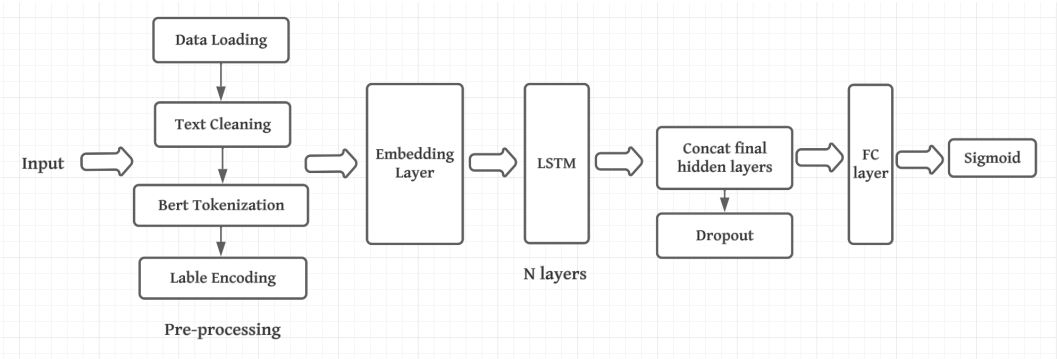


Figure 4: Bi-LSTM Model Architecture

2.2 Bi-LSTM with Attention

The architecture encapsulates a sequence-to-sequence neural network designed for the classification of text data. Starting with raw text input, the data is preprocessed through steps of loading, cleaning, BERT tokenization, and label encoding, transforming the text into a format suitable for machine learning. An embedding layer then assigns vectors to the tokenized text, which are processed by multiple bidirectional LSTM layers. Three stacked bidirectional LSTM layers with a hidden dimension of 256 facilitate comprehensive temporal understanding, extracting features from both past and future states are augmented by a self-attention mechanism that assigns varying levels of importance to different parts of the input sequence, enhancing the model's ability to understand context and relationships within the data.

Following the sequential processing, the concatenated outputs from the LSTM layers, rich with contextual information, are subjected to a ReLU activation function, introducing non-linearity to the feature set. A dropout layer is then applied to prevent overfitting by randomly disabling a subset of neurons during training. The network's culmination is a fully connected layer that integrates these features into a final output, which is subsequently passed through a sigmoid activation function. This final step produces a probability score, denoting the likelihood of the input text belonging to a particular class, which finalizes the binary classification process. The architecture is thus tailored to extract and utilize the intricate patterns within text data, enabling accurate classification decisions.

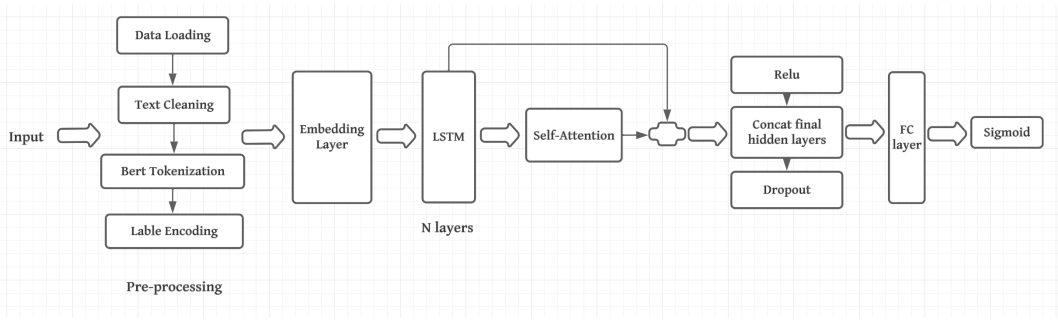


Figure 5: Bi-LSTM with Attention Model Architecture

2.3 Transformer

This Transformer-based model begins with text data that is methodically cleaned and vectorized using TF-IDF, ensuring that the most significant words within the text are emphasized numerically. The model then embeds these vectors into a higher-dimensional space and integrates positional encoding, a critical component that allows the transformer to understand the order and relevance of words in a sequence.

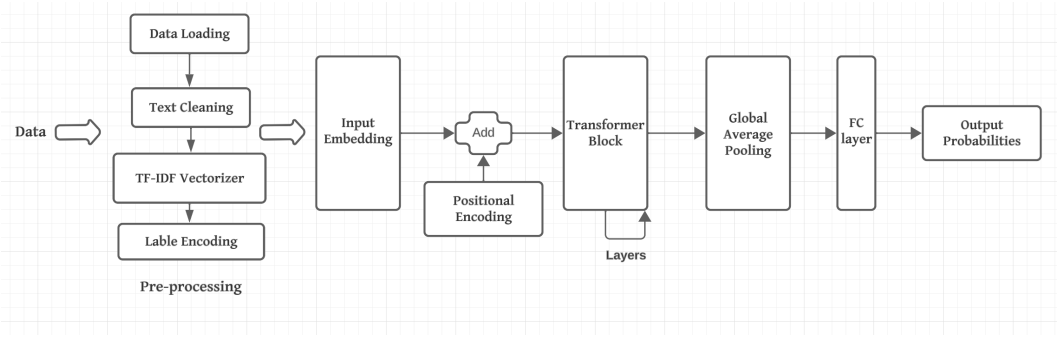


Figure 6: Transformer Model Architecture

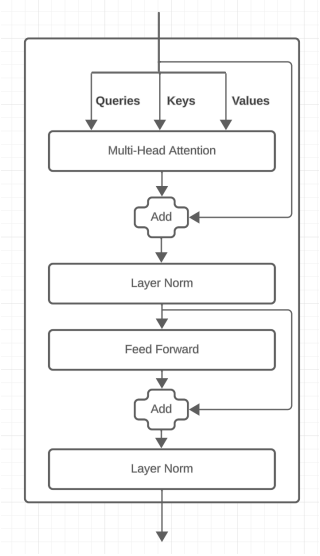


Figure 7: Transformer Block

The self-attention mechanism within the Transformer block is pivotal, enabling the model to dynamically assign importance to different segments of the input data, a process that is inherently more global compared to the local focus in LSTM architectures. Transformer blocks (figure 7), employing multi-head attention and feedforward layers, capture intricate contextual dependencies, allowing the model to discern the relevance of different words within a statement. The inclusion of layer normalization stabilizes training, mitigating issues like vanishing or exploding gradients. Global average pooling reduces spatial dimensions, and the fully connected layer produces the final output for binary classification. The softmax or sigmoid output layer provides the probability scores for each class, giving a quantifiable measure for classification decisions. Unlike the sequential focus of LSTM models, this architecture leverages the parallel nature of Transformers to efficiently handle long-range dependencies and complex patterns in text data, making it highly suitable for sophisticated natural language processing tasks.

2.4 Bert

Through cooperative conditioning on both left and right contexts in every layer, BERT is intended to pretrain deep bidirectional representations from unlabeled text. Consequently, state-of-the-art models for a variety of tasks, including question answering and language inference, can be created by fine-tuning the pre-trained BERT model with just one extra output layer and without requiring significant task-specific architecture adjustments.

BERT is the first representation model based on fine-tuning that outperforms numerous task-specific designs and reaches state-of-the-art performance on a wide range of sentence and token-level problems. Here, we have two stages to complete which are pre-training and fine-tuning. The model was tested on unlabelled data across several pre-training tasks during pre-training. The pre-trained parameters are used to initialize the BERT model for fine-tuning, and labeled data from the downstream jobs is used to adjust every parameter.

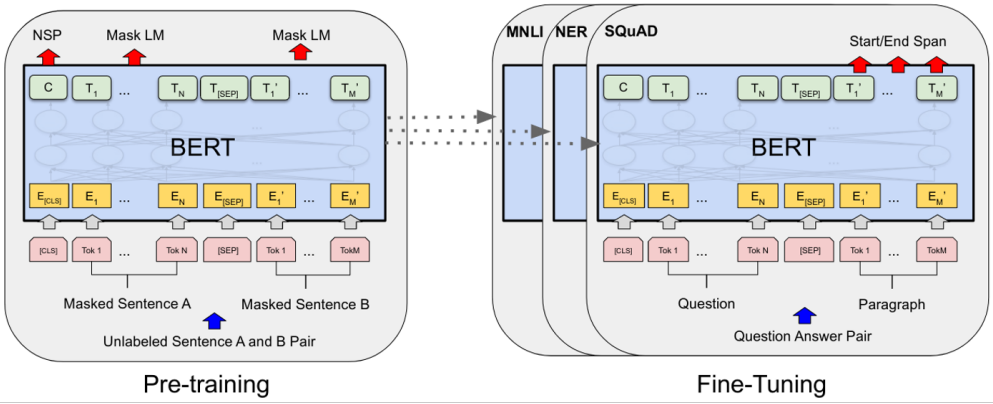


Figure 8: Bert Architecture

2.5 Which Model is Best?

LSTMs may slightly lead in accuracy for fake news detection because they are adept at handling the sequence and context in text, which is crucial for pinpointing the authenticity of news. This can make them a preferable option when the primary goal is achieving the highest possible accuracy in identifying fake news.

However, when it comes to the risk of overfitting, transformers tend to be superior. Despite LSTMs' accuracy, transformers, with their advanced architecture, are better at avoiding the trap of overfitting to the training data, which means they might maintain their performance more reliably when dealing with new, unseen data. This balance between high accuracy and lower overfitting risk makes transformers particularly valuable in practical applications of fake news detection.

3 Loss Function

Chosen Loss Function - Focal Loss: The decision to utilize Focal Loss in the model was driven by its adeptness in handling imbalanced data, a frequent challenge in the domain of fake news detection. This loss function amplifies the focus on challenging classifications, thereby enhancing the model's precision in differentiating between true and fake news. It's particularly beneficial in scenarios where accurate identification of less frequent, but impactful, false news is crucial.

Other Loss Functions Tried - BCE Loss and SmoothL1 Loss: In addition to Focal Loss, the model also explored Binary Cross-Entropy Loss (BCE Loss) and SmoothL1 Loss. BCE Loss is a staple in binary classification models, focusing on the prediction accuracy of dichotomous outcomes. SmoothL1 Loss, straddling the line between L1 and L2 losses, is designed to be more resilient to outliers in data, offering a balanced approach to error measurement.

Innovation on the Loss Function: The application of Focal Loss in the realm of fake news detection represents a notable innovation. This approach not only tailors the model to address the specific challenges of an imbalanced dataset but also boosts the model's effectiveness in identifying nuanced or difficult cases of fake news. Such innovation underscores the ongoing efforts to adapt and refine machine learning techniques for complex real-world problems like fake news detection.

3.1 Description of Loss Functions:

- **Focal Loss**: Designed to address class imbalance by focusing more on the hard-to-classify samples. It adjusts the standard cross-entropy loss such that it down-weights the loss assigned to well-classified examples, emphasizing difficult or misclassified cases.
- **Binary Cross-Entropy Loss**: Measures the performance of a classification model with outputs between 0 and 1. It's ideal for binary classification tasks, penalizing the distance of the predicted probability from the actual label.
- **SmoothL1 Loss**: A combination of L1 and L2 losses, it's less sensitive to outliers than mean squared error. It uses a squared term if the absolute element-wise error falls below a threshold and an L1 term otherwise.

3.2 Which Loss Function is Best?

Focal Loss stands out in the analysis for fake news detection. Its design addresses class imbalance by placing greater emphasis on challenging-to-classify samples. This is particularly beneficial in fake news detection where the differentiation between true and false news is subtle yet crucial. Focal Loss's ability to prioritize hard cases improves the model's sensitivity and accuracy, making it highly effective for this complex task.

4 Optimization Algorithms:

Chosen Optimization Algorithm: The primary optimization algorithm used is Adam (Adaptive Moment Estimation). Adam is renowned for its efficiency and adaptability in handling sparse gradients and varying learning rates, making it highly suitable for deep learning models, particularly in complex tasks like fake news detection.

Other Optimization Algorithms Tried: The presentation also discusses the usage of other optimization algorithms such as AdamW, RMSProp, and SGD (Stochastic Gradient Descent). AdamW modifies Adam by handling weight decay differently for better regularization. RMSProp is designed for adaptive learning rates, ideal for non-stationary objectives. SGD, a more traditional approach, is known for its simplicity and effectiveness in various machine-learning scenarios.

Innovations on Optimization Algorithm: While specific innovations in the optimization algorithms are not detailed, the presentation indicates a comprehensive approach by experimenting with various optimizers. This experimentation reflects an understanding of how different optimization techniques can impact model performance, particularly in the nuanced task of detecting fake news.

4.1 Description of Optimization Algorithm:

- **Adam:** Combines the benefits of two other optimization algorithms, AdaGrad and RMSProp, making it efficient for handling sparse gradients and adapting learning rates. It's widely used in training deep learning models due to its robustness and effectiveness.
- **AdamW:** A variant of Adam, it introduces modifications in handling weight decay, providing better regularization which is crucial for preventing overfitting in complex models.
- **RMSProp:** Focuses on adaptive learning rates, making it well-suited for non-stationary objectives and datasets with variable characteristics.
- **SGD (Stochastic Gradient Descent)**: A traditional and simple optimization method that updates parameters in the opposite direction of the gradient, effective across a wide range of machine learning problems.

4.2 Which optimization is Best?

Adam is identified as the most effective optimizer. It combines the strengths of AdaGrad and RMSProp, offering efficient handling of sparse gradients and adaptable learning rates. These features are vital in training deep learning models, especially for tasks like fake news detection that require nuanced understanding and processing of large datasets. Adam's robustness and adaptability make it a preferred choice for optimizing the performance of complex NLP models.

5 Metrics and Experimental Results

5.1 Bi-LSTM

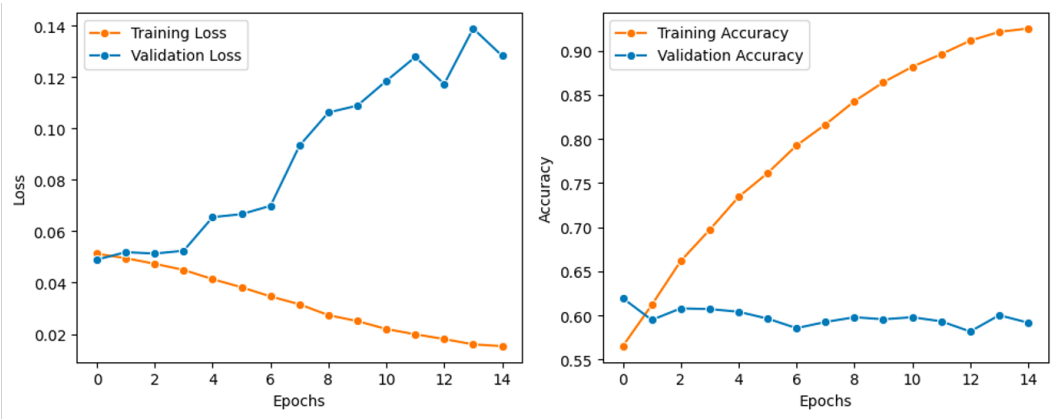


Figure 9: Loss and Accuracies over epochs

In Fig.9, we can see the losses and accuracies of Bi-LSTM model using Focal Loss function and Adam optimizer. The graph shows variations in loss and accuracy throughout the epochs with the max_length of sentence of 128, LSTM layers of 3 and batch size of 16.The training loss shows a desirable downward trend, demonstrating that the model is learning and improving its predictions on the training data. A sharp increase in validation loss after an initial decline indicates the model is not generalizing well and is likely memorizing the training data rather than learning underlying patterns.

Training accuracy improves consistently, which normally would suggest successful learning, reaching near-perfect levels by the final epochs. Validation accuracy does not follow the same trend; it plateaus and then declines, which is a sign that improvements in the model are not translating to unseen data. The divergence between training and validation loss as epochs increase is a strong indicator of overfitting in the model.

And we got a test accuracy 65.5% suggests that the model is able to predict the correct outcomes for approximately two-thirds of the test dataset. While this is above random chance (which would be 50% for a binary classification task), depending on the complexity of the task and the baseline or state-of-the-art results, there may be room for improvement.

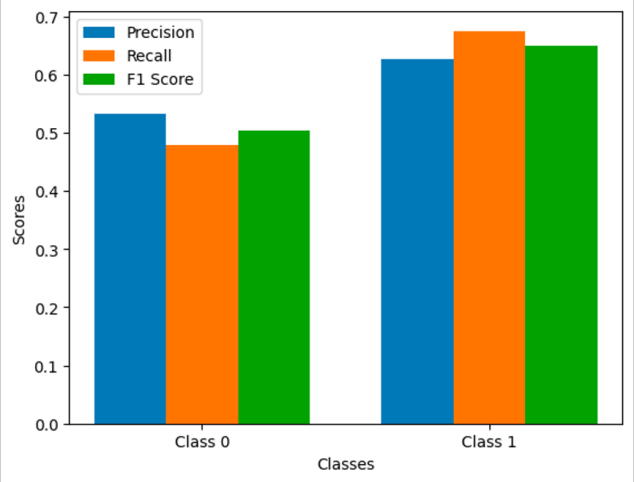


Figure 10: Essential Metrics of Real and fake classes

5.2 Bi-LSTM with Attention

The Hyper-parameters for this model is same as above model except loss functions and optimizers. We tried with 3 different loss functions (Focal loss, BCE loss, Smooth L1 loss) and 3 different optimizers (Adam, AdamW, RMSProp).

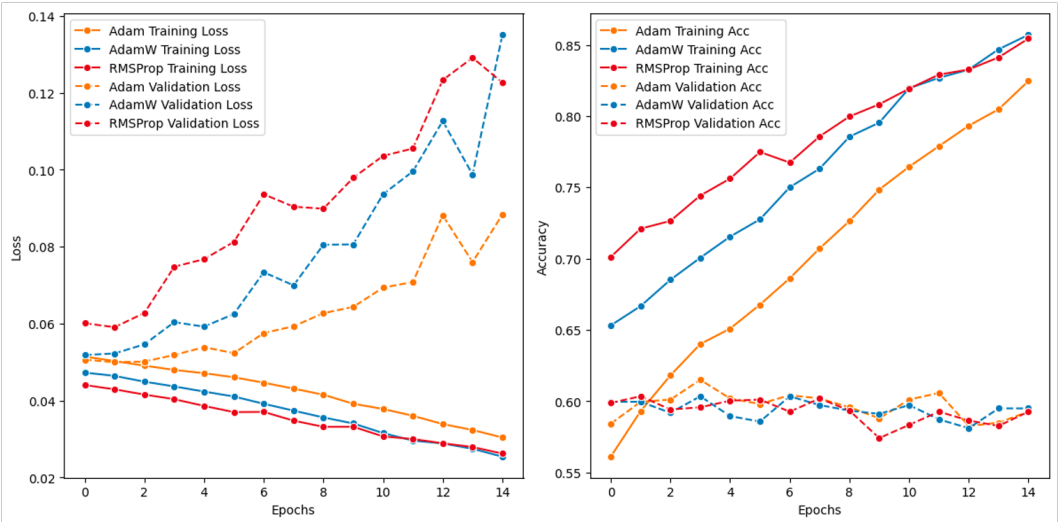


Figure 11: Focal Loss vs optimizers

From Fig.11, The training loss for all optimizers decreases as epochs increase, suggesting learning improvements. Validation loss decreases more consistently for AdamW, implying it generalizes better than Adam and RMSProp, which show spikes indicating possible overfitting. AdamW demonstrates the lowest validation loss by the final epoch, indicating the highest effectiveness in minimizing loss on unseen data.

Training accuracy improves steadily for all optimizers, indicating successful learning from the training data. Validation accuracy for AdamW is higher and more stable across epochs compared to Adam and RMSProp, pointing to its superior performance on validation data. The volatility in validation accuracy for Adam and RMSProp, especially after the initial increase, could be a sign of the models' struggle to maintain consistent generalization across different data samples.

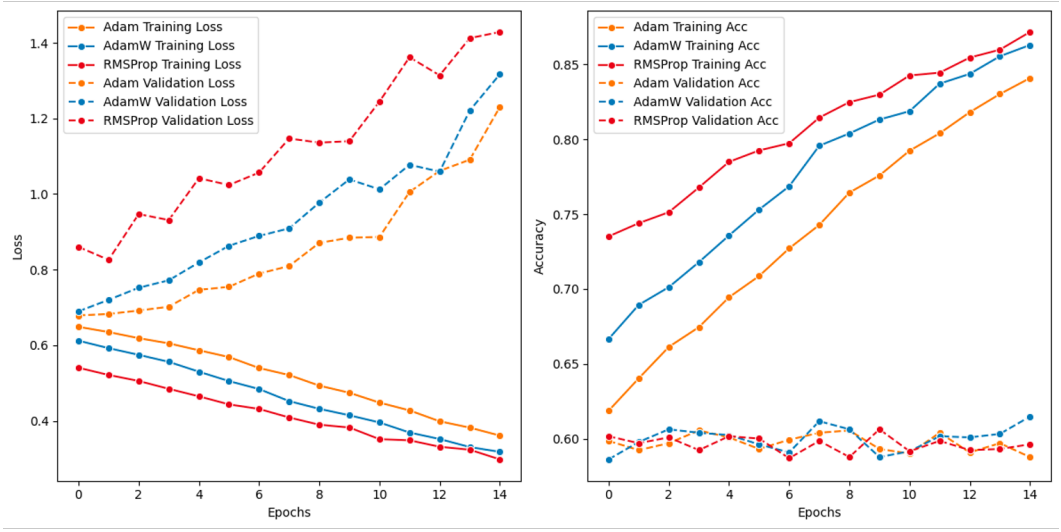


Figure 12: BCE loss vs optimizers

From Fig.12, AdamW excels with a stable decrease in training loss and steady validation loss, indicating effective learning and generalization without overfitting. Adam shows a decrease in loss that eventually increases, suggesting a risk of overfitting by learning irrelevant patterns from the training data. RMSProp's sharply rising validation loss in later epochs signals strong overfitting, performing well on familiar data but failing to generalize.

Accuracy-wise, AdamW steadily improves in both training and validation, showcasing its robust learning. Adam maintains a smaller gap between training and validation accuracy, suggesting decent generalization, while RMSProp's erratic validation accuracy points to difficulties in generalizing learned information to new data.

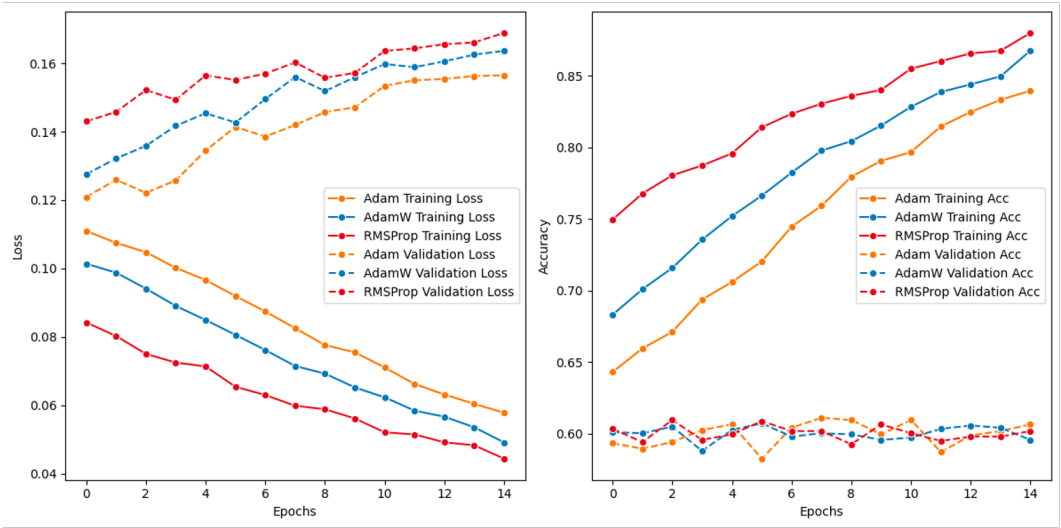


Figure 13: Smooth L1 loss vs optimizers

From Fig.13, it is apparent that the AdamW optimizer is effective in reducing training loss and shows steady improvement in validation loss, highlighting its strong generalization capabilities. On the other hand, Adam and RMSProp exhibit more variability in validation loss, with RMSProp displaying significant fluctuations that could signal concerns about the model's stability.

In terms of accuracy, AdamW demonstrates consistent gains in both training and validation, suggesting robust learning without overfitting. While Adam achieves good training accuracy, it reaches a plateau in validation accuracy, indicating a potential halt in performance gains on unseen data. RMSProp's performance is erratic, with its validation accuracy notably inconsistent, which may affect the model's generalization.

Table 1: Comparsion of Test Accuracies for Bi-LSTM with Atention Model

	Focal Loss	**BCE Loss**	**Smooth L1 Loss**
Adam	60.2%	62.2%	61.01%
AdamW	59.9%	60.6%	61.3%
RMSProp	61.4%	61.2%	61.4%

According to model and from Table 1, Binary Cross-Entropy (BCE) Loss demonstrates optimal synergy when paired with the Adam optimizer, whereas AdamW exhibits a marginally superior performance in conjunction with SmoothL1 Loss. Conversely, RMSProp maintains a notably consistent performance across all three loss functions. These findings can provide valuable insights for selecting an appropriate optimizer and loss function combination in similar tasks or models. In comparison to the previously discussed model, the accuracies are nearly equivalent, however, the incidence of overfitting is considerably reduced.

5.3 Transfromer

For Transfromer model we set dropout as 0.1, batch size of 64, 6 layers, and dimensions of 256. We have choosen accuracies as evaluation metrics which are shown later in the report (Table 2).

The constant drop in training loss, as shown by the transformer model, suggests that the Adam optimizer offers consistent and reliable learning.In Fig.15, RMSProp and SGD, on the other hand, perform less consistently; RMSProp displays oscillations, while SGD has a larger variance in validation loss, which may indicate overfitting. AdamW appears to have a better capacity to generalise by striking a compromise between validation and training accuracy. Compared to RMSProp and SGD, Adam and AdamW perform better, especially when it comes to retaining a smaller validation loss a critical factor for assessing the performance of the model on fresh data. Overall, Adam is exceptional at learning consistently, and AdamW is also good.

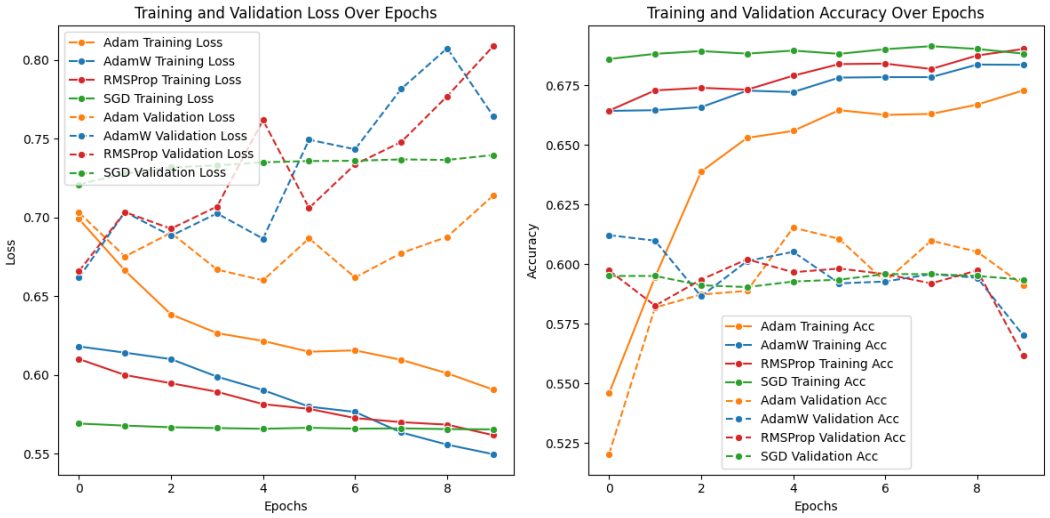


Figure 14: Loss and Accuracy over epochs

From Table 2, for the given task using BCEWithLogits loss, SGD slightly outperforms the other optimizers, with RMSProp being a close second. Adam and AdamW have similar performance levels

but are slightly lower than RMSProp and SGD. The overall differences between the optimizers are relatively small, indicating that the choice of optimizer might not have a drastic impact on the model's accuracy in this case. However, the slight edge of SGD could be significant in scenarios where even minor improvements in accuracy are crucial. This could inform decisions on optimizer selection depending on the task's specific needs and the nuances of the dataset.

Table 2: Comparsion of Test Accuracies for Transformer Model

	Adam	**AdamW**	**RMSProp**	**SGD**
BCEwithlogits Loss	59.8%	59.9%	60.2%	60.4%

5.4 Bert

The performance data of a pretrained BERT model optimized with AdamW and BCELoss shows an effective learning trajectory, with training loss falling sharply before plateauing, suggesting a rapid adaptation to the training data, and a decreasing validation loss that indicates good generalization. Training accuracy significantly improves, and validation accuracy also rises, then levels out, demonstrating the model's ability to generalize. However, the test accuracy stands at around 65.5%, which, while indicative of learning and generalization, also points to potential areas for enhancement.

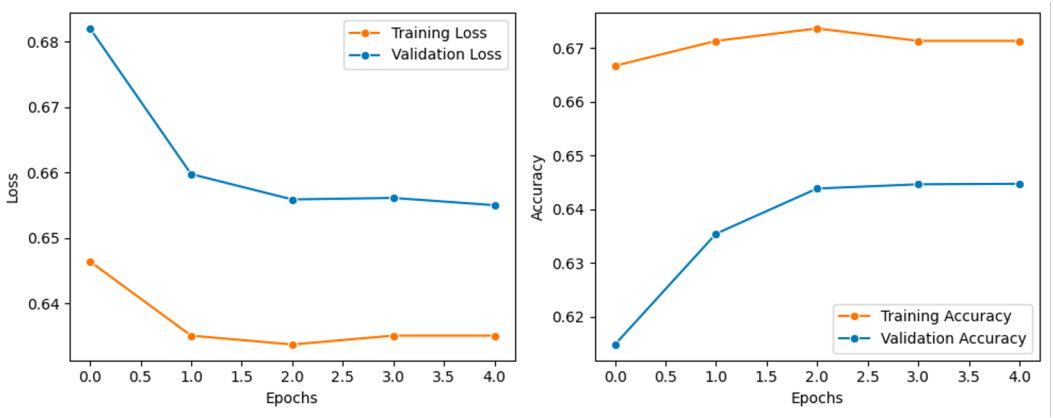


Figure 15: Loss vs Accuracy

6 Conclusion

In conclusion, the presentation adeptly illustrates the use of advanced NLP techniques, specifically LSTM and Transformer models, for fake news detection. The employment of the LIAR dataset and BERT tokenizers marks a noteworthy stride in discerning true from false narratives. Future improvements could involve improving the quality and diversity of the training dataset, including a wider range of topics and sources, which can help the models learn more nuanced patterns of misinformation. Also exploring deeper learning algorithms for nuanced context understanding and implementing cross-lingual capabilities to tackle fake news in various languages, broadening the models' applicability and effectiveness globally.

7 Contributions and GitHub

- **Sri Harsha Vardhan Raju Namburi:** Worked on Bi-LSTM with attention model, model architecture, code, and results.
- **Santhosh Reddy Chilaka:** Worked on Transformer model, model architecture, code, and results.

- **Harsha Sivadhanam:** Worked on pre-trained BERT model and Bi-LSTM model, model architecture, code, and results.
- **Team(All Three):** Worked on data pre-processing, data analysis, and web application.
- **Github link:** https://github.com/santhoshchilaka/Fake-News-Detection

References

[1] M. Qazi, M. U. S. Khan and M. Ali, "Detection of Fake News Using Transformer Model," 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 2020, pp. 1-6, doi: 10.1109/iCoMET48670.2020.9074071.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL-HLT (1) 2019: 4171-4186.

[3] Alghamdi, Jawaher et al. "Towards COVID-19 fake news detection using transformer-based models." Knowledge-Based Systems 274 (2023): 110642 - 110642.

[4] Bahad, Pritika et al. "Fake News Detection using Bi-directional LSTM-Recurrent Neural Network." Procedia Computer Science (2019): n. pag.

[5] Nida Aslam, Irfan Ullah Khan, Farah Salem Alotaibi, Lama Abdulaziz Aldaej, Asma Khaled Aldubaikil, "Fake Detect: A Deep Learning Ensemble Model for Fake News Detection", Complexity, vol. 2021, Article ID 5557784, 8 pages, 2021.