Model Comparison for Bias Detection in Textual Content
Sentients

Sri Chaitanya Nulu, Shromana Kumar, Srinayani Mankali, Manogna Sai Kolluri, Permareddy Hemanth Sagar Reddy
{srichaitanyanulu,shromana.kumar,smankali,m.kolluri,permareddy.h}@ufl.edu

Abstract

There has been a growing concern regarding the prevalence of hateful and discriminatory language on online platforms. Social media companies have been working to implement automated bias detection in textual data to address this issue. We conducted this study focusing on classifying bias in text data. We evaluate the performance of Transformer and Non-Transformer models on this task and compare their effectiveness. This research aims to provide insight into the most effective methods for detecting bias in text, which could have practical implications for online platforms to improve their ability to identify and mitigate discriminatory language and determine the most effective approach for identifying biased content.

1 Introduction

1.1 Background on textual bias analysis

Textual bias analysis is a branch of NLP that focuses on detecting biased language in text data, such as racial, gender, political, or religious bias. It can impact how people perceive and interact with each other online.

1.2 Significance of the research question

"Textual Bias Analysis: Transformer vs Non-Transformer Models - Which model performs best when detecting biased textual content?"

This research question is significant because it addresses an important issue in natural language processing (NLP). Bias in textual content can be harmful, perpetuating stereotypes and reinforcing discrimination. Therefore, it is crucial to develop effective tools to detect and mitigate bias in textual content.

The use of transformer models, such as BERT and GPT, has revolutionized NLP and led to significant improvements in various NLP tasks. However, it is not clear whether transformer models are better than non-transformer models, such as traditional machine learning algorithms or rule-based approaches, at detecting biased textual content. This research question is significant because the answer could inform the development of more accurate and effective tools for detecting and mitigating bias in textual content.

1.3 Hypothesis

Statement: *Transformer model BERT performs better than Non-Transformer Model LSTM.*
This hypothesis is based on the fact that transformer models have shown superior performance on a variety of NLP tasks. Additionally, transformer models are designed to handle long-term dependencies in language, which is important for detecting bias that may be spread out over multiple sentences or paragraphs.

On the other hand, while LSTM is a popular and powerful model for natural language processing, it may not be as effective at detecting bias as transformer models due to its inability to capture long-term dependencies in language. LSTM is designed to handle sequential data but may struggle with understanding complex relationships between different parts of a text.

If the hypothesis is supported, it could lead to the development of more accurate and effective tools for detecting and mitigating bias in textual content. This has significant implications for various industries, such as journalism, advertising, and social media, and ultimately, for creating a more equitable and just society.

2 Literature Review

2.1 Previous research on textual bias analysis

During our literature survey on bias detection in textual data, we reviewed several research papers that provided valuable insights. Among these papers were [1] Study on BERT Model for Hate Speech Detection, [2] Text Classification Research Based on Bert Model and Bayesian Network,

[4] BERT-based Ensemble Approaches for Hate Speech Detection, [5] Detection of Hate Speech using BERT and Hate Speech Word Embedding with Deep Model, [7] Offensive Language and Hate Speech Detection with Deep Learning and Transfer Learning, and [8] Research on Text Classification Method Based on LSTM Neural Network Model

The BERT paper helped us better understand the various BERT-based models (such as ROBERTa and ALBERT) and their architectures. Meanwhile, the papers on LSTM-based models for text classification provided valuable insights into semantic information extraction from text using word embeddings. Lastly, the paper on BiLSTM helped us understand the unique architecture of this model, which involves two LSTMs - one taking the input as it is, and the other taking a backward direction copy of the sequence, thus offering better predictions. The author demonstrates that BiLSTM outperforms traditional LSTM models in identifying English fake news, achieving a higher level of accuracy.

Overall, these papers provided us with a deeper understanding of the models and methods used in bias detection on textual data.

2.2 Comparison of transformer and non-transformer models

In recent years, NLP has seen significant advances, resulting in the development of powerful deep-learning models for text classification. Non-transformer models such as traditional machine learning models and RNN-based models have been widely used for text classification tasks, but they have limitations in handling more complex language and sentence structures.

The transformer-based models such as BERT and GPT have emerged as a new SOTA approach for text classification. These models are pre-trained on large amounts of text data and utilize self-attention mechanisms to capture contextual relationships between words. They have demonstrated remarkable performance gains over non-transformer models in various text classification tasks, including sentiment analysis and natural language inference.

While non-transformer models can still be useful as a baseline for comparison or for simpler text datasets, transformer-based models should be considered the preferred approach for more complex text classification tasks. The ability of transformer-based models to capture contextual relationships between words has made them highly effective in handling the nuances of language and sentence structures, resulting in superior performance.

2.3 Advantages and disadvantages of each model type

Transformers models such as the BERT (Bidirectional Encoder Representations from Transformers), have become popular due to their ability to capture contextual relationships between words in a sentence, resulting in better performance in NLP tasks such as sentiment analysis. They achieve this by using self-attention mechanisms that allow the model to process the sequence of words simultaneously, allowing the transformer model to learn the context of the words from both directions in the sequence. This makes them particularly effective for tasks such as machine translation, text classification, and text generation.

However, transformers require large amounts of computational resources as they are computationally expensive to train, limiting their usage to a few applications. They may also struggle with rare or out-of-vocabulary words, which can negatively impact performance. Additionally, transformers require a large amount of training data to perform well, making them less effective for tasks with limited labeled data.

Whereas, non-transformer models such as LSTM and BiLSTM, are commonly employed in NLP applications because of their simple design and shorter training periods. One of the advantages of these models is their simplicity and efficiency, making them particularly effective for tasks such as sentiment analysis and text classification. And also, they can be effective for tasks with limited labeled data, making them a good option for smaller datasets.

Though the non-transformer model identifies the relationship between the words, it does not take context into account. They may struggle to capture long-range dependencies in sequential data, making them less reliable for tasks that require understanding the full context of a sentence. Additionally, they may require more feature engineering to perform well, making them less adaptable and flexible than transformers. Therefore, these models give a little less accuracy when compared to transformer models.

2.4 Research gaps and limitations

Firstly, We observed that there is a lack of practical implementations of multi-label classification for bias detection in text data. To address this gap, it is essential to develop models that can accurately identify multiple types of biases by handling multiple labels simultaneously. Secondly, the availability of diverse and large datasets for this type of classification remains limited. Furthermore, biases can be observed not only in textual data, but also in other data formats such as images, audio, and videos. Finally, this further highlights the need for developing robust evaluation metrics to ensure that the models are performing optimally. Overall, It is crucial to address these gaps to advance the field of multi-label classification and make it more applicable in various domains.

3 Methodology

In this section, we describe the specific models and techniques used in our experiments, including data preprocessing, model architectures, and evaluation metrics.

3.1 Description of the dataset

The dataset - 'Jigsaw Unintended Bias in Toxicity Classification' from Kaggle was utilized for this research. It contains comments, which are essentially textual data, and bias values ranging from 0 to 1 for categories like gender, sexual orientation, race, religion, and more. It has roughly 45 columns. The fact that the dataset has already been divided into training, testing, and validation datasets made our lives a little simpler. The longest remark we found was 149 words long, while the average length of the comments was around 31 words. Additionally, we saw a 92% lexical diversity, which we thought was really intriguing.

3.2 Preprocessing steps

3.2.1 Non Transformer Models

Preprocessing is a crucial step in building a text classification model. The raw text data is first cleaned by removing irrelevant characters, digits, and punctuation marks. The text is then tokenized into words or subwords, and stop words are removed to reduce the dimensionality of the feature space. After tokenization, the text is transformed into numerical feature vectors using text vectorization techniques. These techniques represent each text sample as a vector of word frequencies or weights, respectively. The resulting feature vectors are often normalized to ensure consistent scale across features. The preprocessed data is then split into training, validation, and test sets with appropriate stratification to ensure balanced class distributions in each set.

3.2.2 Small BERT

The Small BERT transformer model is designed to process sequential data. It starts by converting input tokens to continuous vectors using an input preprocessing layer, which consists of a token embedding layer and a position embedding layer. The token embedding layer maps each token to a fixed-size vector, and the position embedding layer adds information about the token's position in the sequence.

To make the model even more powerful, it's pre-trained on a large corpus of text using a technique called masked language modeling (MLM). This involves randomly replacing some of the input tokens with a [MASK] token and training the model to predict the original token based on the context around it. By doing this, the model learns to understand the meaning of the text at a more contextual level. Pre-training the model using MLM is crucial for boosting its performance, particularly when fine-tuning it for specific tasks.

3.2.3 BERT

The pre-processing step typically involves tokenizing the input text from the dataset into individual tokens, which are then converted into numerical representations that can be fed into the model. The BERT model uses the BertTokenizer module in Transformers, which breaks words into sub-word units and assigns each unit a unique numerical id. Once tokenized, the input text is padded or truncated to a fixed length, typically 512 tokens for BERT Base Uncased. Input IDs are a sequence of numerical ids representing the tokens in the input text; these are then generated for each example in the dataset. To ensure that the model attends to only relevant tokens in the input sequence, attention masks are also generated, which are binary vectors indicating the position of the padded tokens. We then divided these input IDs and attention masks into training and validation sets, which are fed to the pre-trained bert base uncased model inorder to fit the model according to the dataset.

3.3 Model architectures

3.3.1 BERT

BERT is a transformer model, which is pre-trained on English Wikipedia and Book Corpus. From different versions of BERT, we chose BERT base uncased version for our research. The "uncased" part of its name means that the model is trained on lower-cased text and does not differentiate between uppercase and lowercase letters. BERT Base Uncased has 12 transformer layers, 110 million parameters, and can handle a maximum sequence length of 512 tokens. The model takes in tokenized text input, and each token is embedded into a 768-dimensional vector space. The embedding is then passed through the transformer layers, which perform multi-head self-attention and pointwise feedforward operations on the input sequence.

Each transformer layer contains two sub-layers: the multi-head self-attention layer and the pointwise feedforward layer. The multi-head self-attention layer allows the model to attend to different parts of the input sequence simultaneously and capture contextual relationships between words. This is done by calculating attention scores between all pairs of words in the input sequence and weighting each word's representation by its corresponding attention score. The pointwise feedforward layer applies a non-linear transformation to the output of the multi-head self-attention layer to capture complex interactions between the input tokens. Each transformer layer also includes layer normalization, residual connections, and dropout regularization to improve model performance and prevent overfitting. The output of the last transformer layer is then used for various downstream natural language processing (NLP) tasks such as text classification, question answering, and language translation. Overall, the architecture of the BERT Base Uncased model is highly effective in capturing contextual relationships between words and has demonstrated state-of-the-art performance on various NLP benchmarks.

3.3.2 Small BERT

We have also used Small Bert (uncased_L-12_H-768_A-12), which is a smaller version of the original BERT model, but retains its core features and capabilities, and also works well in resource-constrained environments.

The architecture consists of a stack of 12 transformer blocks, each with 768 hidden units. It has a total of 109 million parameters that it can adjust during training to minimize the loss function, except for one parameter that remains fixed at the beginning and does not change during training. This bert model version is based on the Transformer architecture proposed in the paper [6] Attention Is All You Need by Vaswani et al. (2017). It consists of a self-attention layer, a feedforward layer, and a residual connection, followed by layer normalization.

In simple terms, the self-attention layer in the model helps to identify important words in a sentence and prioritize them. This helps the model to better understand the meaning of the text and make more accurate predictions. The feedforward layer adds complexity by applying a non-linear function to the self-attention output, which makes the model smarter at capturing the nuances of the text. The residual connection ensures that the input can still flow through the block, which helps the model learn faster by preventing the gradient from disappearing. Lastly, layer normalization is applied after each block to keep the output values in check and ensure they are consistent and reliable.

3.3.3 LSTM

The architecture of our model is based on the Long Short-Term Memory (LSTM) neural networks, which have demonstrated remarkable performance in various natural language processing tasks. LSTMs are a type of recurrent neural network (RNN) that can process sequential data by maintaining a memory of past inputs and selectively remembering or forgetting information at each time step.

To preprocess the text data, we utilized the powerful Tokenizer module from the Keras library to tokenize the comments and convert them into sequences of integers. Subsequently, we applied padding to the sequences, which were fixed to a length of 149, to ensure that all inputs to the LSTM model had the same shape.

Moreover, to address the class imbalance issue in the dataset, we applied an undersampling technique to the majority class using the RandomUnderSampler module from the imblearn library. This approach randomly selects samples from the majority class to reduce its size and balance the number of samples in each class. This way, our model can learn from a balanced dataset and make unbiased predictions for all classes.

The above preprocessing and balancing tech-

niques are also utilized for the other non-transformer model architectures we used.

The LSTM model consists of an embedding layer and three LSTM layers with 32, 32, and 16 units, respectively. Dropout regularization with a rate of 0.5 was applied after each LSTM layer to prevent overfitting. The final output layer consisted of a single unit with a sigmoid activation function to predict whether the given comment is biased or unbiased.

3.3.4 BiLSTM

Compared to traditional unidirectional LSTMs, BiLSTMs process the input sequence in both forward and backward directions, allowing the model to capture dependencies from past and future inputs. We used a BiLSTM model in our specific case due to the inherent complexity and context-sensitivity of identifying toxic comments. By utilizing a BiLSTM, our model can more effectively understand the nuances of language and better capture the underlying meanings of the comments.

The architecture of the BiLSTM model consists of an embedding layer, followed by two bidirectional LSTM layers with 32 and 16 units, respectively. A dropout layer with a rate of 0.5 is added after the first and second LSTM layers to prevent overfitting. The model ends with a dense layer with a sigmoid activation function to output the predicted probability of a comment belonging to the toxic class.

Using a bidirectional LSTM allows the model to learn from the context of the entire comment rather than just the previous words. It helps the model capture long-range dependencies and improve the performance of the classification task.

Overall, the BiLSTM model architecture proved effective in addressing the class imbalance issue and achieving high accuracy on the task of biased comment classification.

3.3.5 Hybrid

We have tried several hybrid models, such as BILSTM+CNN [3] Text Sentiment Classification Based on Improved BiLSTM-CNN, BILSTM+attention, and Siamese BILSTM +attention. But, we finalized the siamese BILSTM+attention model. The hybrid model in the further sections refer to the siamese BILSTM+ attention model.

In classifying whether a comment is biased or unbiased, the Siamese BiLSTM+Attention model has an added advantage over other models, such as BiLSTM+Attention and BiLSTM+CNN, as it is capable of handling input pairs rather than just individual texts. We can compare two comments to determine their similarity and classify them accordingly. Hence, this approach allows for a more nuanced analysis of text data and ultimately results in more accurate and robust classification performance.

We constructed a Siamese BiLSTM model with a self-attention layer, using an embedding layer with 128 output dimensions to convert the text data into a numerical representation. The model mainly consisted of three BILSTM layers, each with decreasing units 64,32,16 respectively, followed by a self-attention layer to weigh the importance of each word in the input sequence to improve the model's ability to detect patterns and relationships between different text .and a dropout layer to avoid overfitting. The final output from the last BILSTM layer was passed through a GlobalAveragePooling1D layer to get a single feature vector. Further, this vector was fed into a dense layer with a sigmoid activation function to predict the output. This architecture with GlobalAveragePooling1D is particularly suitable for text classification tasks, as it allows the model to learn important features from variable-length input sequences and produce accurate output.

3.4 Model training and testing

3.4.1 BERT

As the BERT base uncased model is pre-trained, the model has to be configured and trained according to our chosen dataset. The model is trained using the ADAM optimizer with a learning rate of 0.00002, which is a popular optimization algorithm used for deep learning models. The CategoricalCrossentropy loss function is used for training, which measures the difference between the predicted probabilities and the true labels. The SparseCategoricalAccuracy metric is used to evaluate the model's accuracy on the test dataset, which measures the percentage of correctly classified examples. During training, the ModelCheckpoint callback is used to periodically save the weights of the model and monitor its performance on a validation dataset. We have trained the BERT model using a training dataset and validation dataset for 3 epochs and a batch size of 64.

As hyperparameters such as learning rate, number of epochs, and batch size significantly impact

the model's performance, we tuned the model by modifying these values which enhanced the working of the BERT model.

After finetuning and training the BERT model, we evaluated the performance of BERT by testing it using the test dataset. On evaluation, we have obtained an accuracy of 83%.

3.4.2 Small BERT

For training this model we have used the binary cross-entropy loss function, which is widely used in binary classification problems. The model's performance during training is evaluated using the accuracy metric. To ensure a balanced dataset, we perform oversampling using the RandomOverSampler from the imblearn package. Furthermore, the dataset is divided into training, validation, and testing sets utilizing the train_test_split function from the scikit-learn library.

To optimize the model's performance, we employ the AdamW optimizer with a learning rate of 3e-5. During training, the optimizer gradually increases the learning rate to the desired value with warmup steps. The model is trained for 3 epochs with a batch size of 64 while being assessed on the validation set to monitor performance and prevent overfitting. Due to the size of the BERT model, the training process takes a significant amount of time (1̃0 hours).

Following training, we evaluated the model's performance on the testing dataset using the standard accuracy metric. Our model achieved a high accuracy of 95.51%, indicating its effectiveness in classifying comments as biased or unbiased.

3.4.3 LSTM

During training, we defined the LSTM model with the above architecture and further utilized binary cross-entropy loss as the loss function and the Adam optimizer with a learning rate of 0.001 to optimize our model's weights. We incorporated class weights in the training process to handle the class imbalance problem. These weights were computed using the compute_ class_ weight function from the scikit-learn library, which calculates each class's inverse proportion of samples.

The model was trained for 20 epochs using a batch size of 32. Moreover, we employed early stopping to prevent overfitting, which stops the training process when the validation loss stops improving. We used the EarlyStopping callback from Keras, with a patience value of 3, which means that the training would stop if the validation loss did not improve for three consecutive epochs. Additionally, we saved the best model using the ModelCheckpoint callback to retrieve the model with the lowest validation loss.

Further, we tuned the model by changing the hyperparameters of the model such as epochs,embedding layers, dropout rate, optimizer and number of LSTM units.

Finally, the performance of the model was evaluated on the testing set, and we achieved an accuracy of 70.34%.The achieved results were not promising as they failed to consider the contextual meaning in a few scenarios.

3.4.4 BiLSTM

During training, we defined the BILSTM model with the above architecture and further utilized binary cross-entropy loss as the loss function and the Adam optimizer with a learning rate of 0.001 to optimize our model's weights.

During the training process, we used a batch size of 32 and trained the model for 20 epochs. We also implemented early stopping with a patience value of 3 to prevent overfitting by stopping the training process when the validation loss stopped improving. To retrieve the model with the lowest validation loss, we utilized the ModelCheckpoint callback to save the best model during the training process. It ensures that we have the best model saved, which can be used later for evaluation.

Further, we tuned the model by changing the hyperparameters of the model such as epochs, embedding layers, dropout rate, optimizer, and number of BILSTM units.

Finally, the performance of the model was evaluated on the testing set, and we achieved an accuracy of 70.59%.

3.4.5 Hybrid

During training, we defined the BILSTM model with the above architecture. The model was compiled with binary cross-entropy as the loss function and the Adam optimizer. We used class weights to account for the imbalanced data during training. Early stopping and model checkpoint callbacks were used during training to prevent overfitting and save the best model. The model was trained for 20 epochs using a batch size of 32 and evaluated on a testing set. The best model was saved and loaded to evaluate the testing set.

We further tuned the model by changing the hyperparameters of the model such as the number of epochs, optimizer, number of BILSTM layers, attention layers and etc.

Finally, the performance of the model was evaluated on the testing set, and we achieved an accuracy of 73.04%.

3.5 Evaluation metrics

Initially, in our research, we considered utilizing conventional measures such as accuracy, F1 score, recall, precision, and similar metrics. However, we recognized that these measures may not provide sufficient insights into the performance of a model. After extensive exploration of various metrics and devising new ones, we decided to use the following standard metrics to evaluate all the models:

- **Accuracy:** a measure of how often the model correctly predicts the label of a sample.
- **False Positive Rate:** The proportion of actual non-biased cases that are incorrectly predicted as biased.
- **False Negative Rate:** The proportion of actual biased cases that are incorrectly predicted as non-biased.

As we were developing the hybrid model, we chose to incorporate certain metrics that were helpful in evaluating and enhancing non-transformer models. These metrics are as follows:

- **Mean Absolute Mismatch Error:** the average absolute difference between the predicted and actual values, calculated only for samples where the predicted and actual values are not equal.
- **Bias Amplification Factor:** the average ratio of predicted value to the actual value, calculated only for samples where the actual value is non-zero.
- **Bias Proportion Ratio:** The ratio of the proportion of predicted biased cases to the proportion of true biased cases.

4 Results

In this section, we will evaluate the accuracy and performance of transformer and non-transformer models. Our research has shown that transformer models are more effective than non-transformer models for textual classification tasks. This is because transformer models process information bidirectionally, allowing them to capture contextual relationships between words more accurately.

To further illustrate this point, we have conducted a comparison between the transformer model BERT and the bidirectional LSTM, LSTM. As shown in the table below, BERT outperforms the bidirectional LSTM and LSTM in terms of accuracy and efficiency.

4.1 Performance Comparison of Non-Transformer and Hybrid Models using Special Metrics

We visualized certain special metrics of non-transformer models to improve our understanding of how non-transformer models can be improved. We evaluated LSTM and BiLSTM models, along with the Siamese BiLSTM Model with an Attention Layer. By analyzing these metrics, we were able to assess the models' performance.

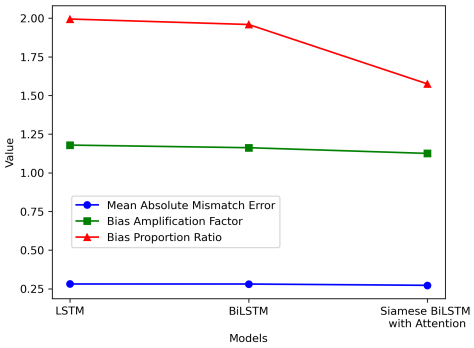

Figure 1: Performance comparison of non-transformer and hybrid models using special metrics.

Figure 1 shows the performance comparison of the non-transformer and hybrid models using the special metrics. The x-axis represents the model names, while the y-axis represents the value of the metrics.

As you can see in the figure, our analysis revealed that the LSTM and BiLSTM models performed similarly, with Mean Absolute Mismatch Errors of 0.28 for both models. The Bias Amplification Factor was slightly lower for the BiLSTM model at 1.16 compared to the LSTM model's 1.18, indicating that the BiLSTM model was less biased toward specific target values. Similarly, the Bias Proportion Ratio was slightly lower for the BiLSTM model at 1.96 compared to the LSTM model's 1.99.

The Siamese BiLSTM model (with attention layer) performed slightly better than both LSTM and BiLSTM models, with a Mean Absolute Mismatch Error of 0.27. The Bias Amplification Factor was also lower for this model at 1.13, indicating that it was the least biased towards specific target values. The Bias Proportion Ratio was the weakest for the Siamese BiLSTM model (with attention layer) at 1.58, indicating less bias toward particular target values.

Hence, we can conclude that the Siamese BiLSTM model (with attention layer) outperforms the LSTM and BiLSTM models regarding mean absolute mismatch error, bias amplification factor, and bias proportion ratio.

4.2 Performance Comparison of BERT, LSTM, and hybrid models on the dataset using Evaluation Metrics

We used visualizations to analyze and compare standard metrics such as Accuracy, False Positive Rate (FPR), and False Negative Rate (FNR) across all the models. While accuracy is a crucial metric for measuring a model's performance, FPR and FNR provide additional insights into a model's behavior in different scenarios. By examining these metrics, we gained a deeper understanding of the strengths and weaknesses of each model and were able to draw conclusions about their suitability for specific tasks.

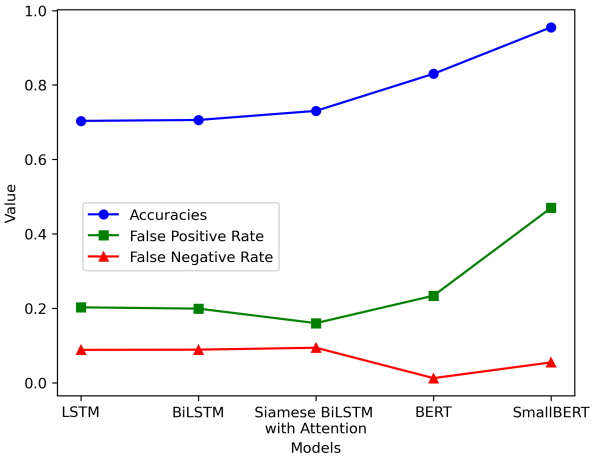

Figure 2: Performance comparison of all models using standard metrics.

The non-transformer models, specifically LSTM, BiLSTM, and Siamese BiLSTM, performed consistently, with low FPR and FNR. LSTM and BiLSTM had identical FPR and FNR, both having an FPR of 0.20 and an FNR of 0.09. Siamese BiLSTM performed slightly better, with an FPR of 0.16 and an FNR of 0.09. On the other hand, BERT had a higher FPR of 0.23 but a significantly lower FNR of 0.01.

When considering accuracy, small BERT had the highest accuracy of 95.51%, followed by BERT with an accuracy of 83 %. Siamese BiLSTM had an accuracy of 73.04%, followed by BiLSTM with an accuracy of 70.59% and LSTM with an accuracy of 70.34%. It should be noted, however, that Small BERT had a considerably higher FPR of 0.47 and a relatively high FNR of 0.05, indicating that it may not be the best choice for tasks where minimizing false positives is important.

Thus, the transformer model BERT, which has a significantly lower FNR than the other non-transformer models and also Small BERT, suggests that it may be more suitable for tasks where minimizing false negatives is critical. Additionally, BERT had a relatively high accuracy compared to the deep learning models, indicating its potential for high performance in a broader range of tasks. Although BERT had a higher FPR than some of the deep learning models, its lower FNR and higher accuracy make it a better choice in scenarios where both FNR and overall accuracy are essential metrics.

4.3 Examples

In this section, we will assess the classification performance of each model on biased comments, which will provide insights into how various types of comments affect model performance. Here are some of the examples[1] we will be discussing:

- ***"So he's a woman now? Mental Illness should be kept discreet. This is too much information and we did not need to know this. WHY is this news?"***
  Result: Biased

Model	Prediction
LSTM	0.17 (Unbiased)
BiLSTM	0.13 (Unbiased)
Siamese BiLSTM	0.18 (Unbiased)
BERT	1 (Biased)
Small BERT	1 (Biased)

Transformer models like BERT and Small BERT can detect bias in the text by under-

[1]Disclaimer: These examples are for illustrative purposes only and are not meant to offend anyone.

standing the meaning and context of words. This was demonstrated in the comment's negative attitude toward a person's gender transition. Non-transformer models like LSTM lack the context to identify such bias, highlighting the importance of transformer models in bias detection and mitigation.

- ***"1- All that silliness about people mysteriously dying is probably fun to think about but it is silliness. 2- Ms Clinton HAS been fact-checked, here and in the source Another Reader provided. Can find 10-20 more, if you like. There is no shadow army, there is no killer ray the Clintons use on folks who disagree with them, and no cool scary dude or dudette waiting to get ya ... I know, it's no fun to have to live with boring old reality. sigh." Result: Unbiased***

Model	Prediction
LSTM	0.03 (Unbiased)
BiLSTM	0.04 (Unbiased)
Siamese BiLSTM	0.08 (Unbiased)
BERT	1 (Biased)
Small BERT	0 (UnBiased)

The given comment contains a dismissive attitude towards conspiracy theories about the Clintons. While LSTM and BiLSTM models predicted a low level of bias in the comment, Siamese BiLSTM predicted a slightly higher level of bias. However, the BERT model predicted a high level of bias in the comment, possibly due to its ability to detect subtle forms of bias in the text. In contrast, the Small BERT model predicted no bias in the comment.

- ***"Why equal immigration just with welfare, crime and terrorism? I'm pretty sure I've seen stories about immigrants being pedofiles, devil worshippers and... overall determined to rape our daughters and wifes - heck even your bigotted ass (they like homosex, you know). This is why we need responsible citizens like yourself, willing to get on a horse, dressed in white sheets... right?" Result: Biased***

Model	Prediction
LSTM	0.90 (Biased)
BiLSTM	0.92 (Biased)
Siamese BiLSTM	0.90 (Biased)
BERT	1 (Biased)
Small BERT	1 (Biased)

All the models were able to detect bias in this comment. The LSTM, BiLSTM, and Siamese BiLSTM models predicted that the comment was biased, possibly due to their focus on the overall sentiment of the comment. Similarly, BERT and Small BERT models were able to detect bias, likely due to their ability to identify specific phrases and language patterns associated with bias and prejudice.

- ***"You can't handle the troof!!!!" Result: Unbiased***

Model	Prediction
LSTM	0.14 (Unbiased)
BiLSTM	0.15 (Unbiased)
Siamese BiLSTM	0.14 (Unbiased)
BERT	0 (Unbiased)
Small BERT	0 (Unbiased)

All models predicted the comment as unbiased. The LSTM, BiLSTM, and Siamese BiLSTM models may have classified it as unbiased due to the absence of negative sentiment towards any specific group. BERT and Small BERT models likely identified the lack of specific phrases or language patterns associated with bias and prejudice in the comment. The comment's lack of explicit bias or prejudice likely contributed to all models predicting it to be unbiased.

5 Discussion

Bias detection is a critical task in various fields, including natural language processing, machine learning, and artificial intelligence. It is important to ensure that the models developed are fair and unbiased towards different groups, and do not perpetuate or amplify any existing biases in the data.

5.1 Summary

In this context, the analysis and comparison of different models using various metrics are crucial for identifying the strengths and weaknesses of each model and choosing the most appropriate one for

a specific task. As we have seen in the example provided, the use of visualizations and standard metrics such as accuracy, FPR, and FNR can provide valuable insights into a model's behavior and its suitability for different scenarios.

The Siamese BiLSTM model (with attention layer) performed the best in terms of mean absolute mismatch error, bias amplification factor, and bias proportion ratio, indicating that it is less biased towards specific target values and can provide more accurate predictions. However, it is essential to consider other metrics such as FPR, FNR, and overall accuracy when choosing a model.

The BERT transformer model outperformed the non-transformer models in terms of FNR and accuracy, indicating its potential for high performance in a broader range of tasks. Although it had a higher FPR than some of the deep learning models, its lower FNR and higher accuracy make it a better choice in scenarios where both FNR and overall accuracy are essential metrics.

5.2 Justification of Hypothesis

The hypothesis for this study is that the Transformer model, specifically BERT, will outperform the Non-Transformer Model, specifically the Siamese BiLSTM model, in detecting biased textual content. This hypothesis is based on the fact that Transformer models, such as BERT, have been shown to have superior performance in a variety of natural language processing tasks, including text classification and sentiment analysis.

The main advantage of Transformer models is their ability to capture the contextual relationships between words in a sentence or text, which can be especially important in detecting biased language where certain words or phrases may have different meanings or connotations depending on the context. On the other hand, Non-Transformer Models, such as BiLSTM, are known to have limitations in capturing long-range dependencies and may struggle with complex sentence structures.

Therefore, based on these known differences in performance between Transformer and Non-Transformer models, it is reasonable to hypothesize that BERT will perform better than Siamese BiLSTM in detecting biased textual content.

5.3 Implications of Research

The results of this study carry significant implications for practical use in real-life scenarios. Firstly, they shed light on the importance of considering multiple metrics to assess the effectiveness of transformer and non-transformer models in detecting biased content. Secondly, they emphasize the need to combine the strengths of both transformer and non-transformer models to achieve the best possible outcomes. Finally, it highlights the crucial role of integrating bias detection into the development of models that use a transformer and non-transformer architectures to ensure fairness and minimize any harm caused by biases towards individuals or groups.

Our study also justifies the hypothesis that transformer model BERT performs better than non-transformer model LSTM when detecting biased textual content. This insight is critical for researchers and practitioners working in natural language processing, machine learning, and artificial intelligence to develop fair and unbiased models. It also opens up opportunities for further research into the development of hybrid models that can leverage both transformer and non-transformer architectures to achieve optimal performance in bias detection tasks. Overall, this study's findings are a step forward toward developing more ethical and fair machine learning systems that can better serve and benefit society.

5.4 Data vs. Model - Which is more important?

When training an NLP model, both the data and the model architecture are important factors that can significantly impact the performance of the model. The quality and quantity of the data used to train the model can have a significant impact on its accuracy and generalizability. In our project, we have observed that both transformer and non-transformer models can produce unexpected results when trained on unbalanced datasets. Specifically, when trained on a dataset consisting of around 90% records classified as 0, we have observed instances where both transformer and non-transformer models have produced a classification output of all 0s. This unexpected result highlights the importance of carefully considering the class distribution of the training data when training NLP models, as unbalanced datasets can introduce bias and negatively impact the performance of the model.

While data quality is certainly an important factor when training NLP models, the choice of model architecture and hyperparameters should not be overlooked as they can also significantly impact

the performance of the model. After balancing the dataset, we observed a clear difference in the performance of the transformer and non-transformer models. Specifically, the transformer model BERT emerged as the clear winner, highlighting the importance of selecting an appropriate model architecture and hyperparameters to achieve optimal performance in NLP tasks.

5.5 Multilabel Classification & Limitations

Our multilabel classification study used models like BERT (83.14%) and BiLSTM (66.63%) to predict multiple labels for comments. They were achieving comparable performance to their single-label equivalents but with slightly lower accuracy. However, we encountered missing values where certain labels were not applied to comments that should have had them. We examined the dataset and found potential bias in the labeling of comments, which may have contributed to the issue. Also, the multilabel classification values we obtained did not always make sense due to the issue of missing labels.

To improve the accuracy and fairness of toxicity classification models, we needed more diverse and comprehensive datasets that are less susceptible to bias. Additionally, exploring different models and techniques to address missing values in multilabel classification is crucial.

Despite recognizing the issue of missing values in our study, we were, unfortunately, unable to fully address it due to limitations in the dataset and time constraints. While we experimented with different techniques and models to mitigate the issue, we ultimately were unable to achieve a significant improvement. Nonetheless, we believe that our study provides valuable insights into the potential of multilabel classification for toxicity classification models and highlights the need for further research in this area.

6 Future Scope

As of now in our research, we have not taken empathy into consideration as a factor in identifying hateful, discriminatory speech or bias. However, we wanted to address the question of whether empathy could be a valuable part of a solution to the problem is an interesting one. Empathy, which is the ability to understand and share the feelings of others, maybe prove to be useful for recognizing and addressing bias. By training language models to recognize the emotional tone of the input text and empathetically understand the perspective of different groups, it may be possible to develop more effective methods for detecting and addressing hateful and discriminatory speech. Incorporating empathy into the process of bias detection could lead to more nuanced and human-centered approaches that better reflect the complexities of language and social interactions.

We also aim to explore different techniques for performing multilabel classification accurately. This will involve acquiring or creating a suitable dataset and adapting our existing models used for single-label classification. We will then fine-tune the hyperparameters to enhance the performance metrics.

Overall, our investigation into multilabel classification could pave the way for the development of more advanced and effective machine-learning models. This has the potential to open up new avenues for research and practical applications in several fields.

7 Conclusion

Our research comparing Transformer and Non-Transformer models for detecting biased language in textual data has revealed the strengths and weaknesses of each model. Initial results suggest that the Transformer model performs better overall in identifying instances of the biased language across different types of bias, but the Non-Transformer model may be better suited for certain types of bias.

Given the significance of detecting biases in comments, we believe that minimizing the false negative rate (FNR) is a crucial aspect to consider when selecting a suitable model. It can help to minimize the chances of missing comments that contain biases, which can have serious implications. Therefore, we would like to stress the importance of choosing BERT, as it outperformed the other models in terms of FNR, indicating its potential for better performance in identifying biased comments.

8 Code

You can find the link to our entire code repository at here. Our code is organized into various folders such as models, preprocessing, datasets, and so on.

References

[1] Shailja Gupta, Sachin Lakra, and Manpreet Kaur. 2020. Study on bert model for hate speech detection.

[2] Songsong Liu, Haijun Tao, and Shiling Feng. 2019. Text classification research based on bert model and bayesian network.

[3] Ruixin Ma, Shoryu Teragawa, and Zhanjun Fu. 2020. Text sentiment classification based on improved bilstm-cnn.

[4] Khouloud Mnassri, Praboda Rajapaksha, Reza Farahbakhsh, and Noel Crespi. 2022. Bert-based ensemble approaches for hate speech detection.

[5] Hind Saleh, Areej Alhothali, and Kawthar Moria. 2021. Detection of hate speech using bert and hate speech word embedding with deep model.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

[7] Bencheng Wei, Jason Li, Ajay Gupta, Hafiza Umair, Atsu Vovor, and Natalie Durzynski. 2021. Offensive language and hate speech detection with deep learning and transfer learning.

[8] Yanbo Zhang. 2021. Research on text classification method based on lstm neural network model.