Report on Bias Detection in Legal Documents

Team 6 Bias detection in legal documents

Team Members:

Hamza Babukhan (SE22UARI208)

Ayaz Ahmed Ansari (SE22UARI024)

Sandeep Surapaneni (SE22UARI151)

Aryan Pawar (SE22UARI195)

Project Overview:

The Bias Detection in Legal Documents project aimed to address the critical issue of bias in legal texts using advanced Natural Language Processing (NLP) techniques and machine learning models. Legal language profoundly influences societal norms and justice delivery, but it can inadvertently reflect implicit or explicit biases influenced by factors such as race, gender, or socio-economic status. This project sought to develop a robust, automated system for detecting and analysing biases in legal documents. Leveraging a combination of preprocessing techniques, tokenization, dependency parsing, contextual embeddings, and pre-trained transformer models, the system aimed to identify biased versus unbiased documents with high precision and recall. The goal was to provide a scalable and effective solution for mitigating biases in legal documentation, aiding legal practitioners, policymakers, and researchers in promoting fairness and neutrality in legal systems. This report presents the comprehensive methodology followed during the project, including preprocessing steps, model training, evaluation, and analysis of results. Key insights and the project's potential impact are also discussed.

Prior Related Work

Several studies have explored bias detection in textual data, emphasizing societal implications. Techniques such as sentiment analysis, topic modeling, and neural network-based classifiers have been widely used. Research on using transformer models, like BERT and GPT, for bias detection has shown promising results in improving accuracy and interpretability. Additionally, preprocessing techniques such as lemmatization, stop-word removal, and tokenization have been crucial for achieving cleaner datasets and better model performance.

DATA SET :

https://www.kaggle.com/datasets/amohankumar/legal-text-classification-dataset

About Dataset

The dataset contains a total of 25000 legal cases in the form of text documents. Each document has been annotated with catchphrases, citations sentences, citation catchphrases, and citation classes. Citation classes indicate the type of treatment given to the cases cited by the present case.

The dataset has four columns, namely Case ID, Case Outcome, Case Title, and Case Text. The Case ID column contains a unique identifier for each legal case, the Case Outcome column indicates the outcome of the case, the Case Title column contains the title of the legal case, and the Case Text column contains the text of the legal case.

Methodology/Model

1. Preprocessing and Cleaning

• Objective: Prepare raw legal documents for effective downstream analysis by removing irrelevant and noisy elements.

• Actions:

- Cleaning Text: Removed unwanted characters, including special symbols, numbers, and extraneous whitespaces, to standardize the input.
-
- Case Standardization: Converted all text to lowercase to ensure uniformity across the dataset, especially for tokenization.
- Stopword Removal: Removed commonly used words that do not contribute to bias detection (e.g., "and," "the").

- Handling Missing Data: Ensured no critical information was missing or inconsistently formatted in the dataset.

2. Tokenization

• Objective: Break down the cleaned text into smaller components (tokens) while preserving contextual meaning for input into the NLP model.

• Actions:

- Used Hugging Face's AutoTokenizer, optimized for pre-trained transformer models.

• Models Used: Tokenizer from distilbert-base-uncased.

3.Stemming :

was applied to reduce words to their root forms, minimizing vocabulary size and computational overhead while retaining meaningful information.

• Models Used: Applied stemming algorithms (e.g., Porter Stemmer) to reduce inflectional forms of words (e.g., "arguing" to "argu").

4. Named Entity Recognition (NER)

• Objective:

Identify and categorize critical entities in the text, such as names, dates, organizations, and locations.

• Actions:

1. Leveraged pre-trained NER models to extract relevant entities.
2. Verified that extracted entities did not introduce biases (e.g., stereotypical associations).

Models Used: Pre-trained NER model from the spacy library

5. Dependency Parsing

• Objective: Understand the grammatical structure of sentences and relationships between words.

• Actions:

1. Used dependency parsing tools to analyse syntactic roles (e.g., subject, object) within sentences.
2. Identified patterns of dependency relationships that could influence biased phrasing.

• Models Used: Dependency parser from the spacy library.

6. Contextual Embedding

• Objective: Enhance the representation of words by capturing their context within the sentence.

• Actions:

1. Implemented BERT-based embeddings to dynamically capture semantic nuances.
2. Ensured that the embeddings were fine-tuned to reflect the intricacies of legal language.

• Models Used: Pre-trained distilbert-base-uncased model.

7. Insertion of Bias Label in the Dataset

• Objective:

Annotate documents with labels (1 for biased, 0 for unbiased) to enable supervised learning.

• Actions:

1. The dataset used in this project did not include a verdict on whether the legal judgments were biased or unbiased. To train a supervised model, it was necessary to manually label the documents with bias labels.
2. Carefully reviewed the dataset to ensure high-quality annotations that accurately reflect biased versus unbiased judgments.
3. Maintained a balanced dataset to prevent model bias towards majority classes.

• Models Used: the bias labelling was a manual annotation task.

8. Splitting Data into Training and Testing Sets

• Objective: Divide the dataset for model training and performance evaluation.

• Actions:

1. Allocated 80% of the data for training and 20% for testing, ensuring equal representation of both classes.
2. Verified that the split did not inadvertently introduce bias into the training or testing sets.

9. Tokenizing Data for Model Training

• Objective:

Convert annotated text into numerical representations for model input.

• Actions:

1. Applied tokenization to generate input IDs, attention masks, and segment IDs.

2. Handled padding and truncation to standardize input sizes.

• Models Used: Tokenizer from distilbert-base-uncased.

Experiments

Tools Used:

- **Google Colab**: For computational resources and collaborative development.

- **Creating a PyTorch Dataset Class**
- Objective: Standardize the loading and batching of data for model training and evaluation.
- Actions:
- Developed a custom dataset class to efficiently map input IDs, attention masks, and labels.
- Enabled seamless batching and shuffling for optimized training performance.

- Models Used: this step involved PyTorch utilities.

Setting Up and Training a Pretrained Model

• Objective: Fine-tune a pre-trained transformer model for bias detection.

• Actions:

- Used distilbert-base-uncased for binary classification.
- Fine-tuned the model using annotated legal documents, adjusting learning rates and batch sizes for optimal performance.

• Models Used: distilbert-base-uncased model for binary classification.

Experiment Setup:

- **Objective**: Measure the initial performance of the trained model on unseen data.
- **Evaluation Metrics**: Accuracy, Precision, Recall, and F1-Score.
- **Baseline Comparison**: Compared transformer-based models with traditional machine learning methods.
- **Ablation Studies**: Analyzed the impact of preprocessing techniques on model performance.

• Models Used: distilbert-base-uncased.

Analysing Misclassifications

• Objective: Understand scenarios where the model made incorrect predictions.

• Actions:

o Analysed false positives and negatives to identify patterns or ambiguities in the data.

• Models Used: Analysis involved reviewing predictions from distilbert-base-uncased

Fine-Tuning Further

• Objective: Enhance the model's performance by refining hyperparameters and data inputs.

• Actions:

- Adjusted the number of epochs and regularization parameters for improved results.

• Models Used: distilbert-base-uncased.

Saving the Model

• Objective: Save the trained model for future use.

Results

Making Predictions on the Test Set

• Objective:

Generate predictions for unseen data and validate them against true labels

. • Actions: o Used the trained model to classify test documents as biased or unbiased.

• Models Used: distilbert-base-uncased.

Evaluating Model Performance

Achieved a high accuracy of 99.06%, precision of 99.05%, recall of 100%, and F1-score of 99.53%.

Evaluating Model Performance

```python
from sklearn.metrics import accuracy_score, precision_recall_fscore_support

# Calculate accuracy
accuracy = accuracy_score(true_labels, predictions)

# Calculate precision, recall, and F1-score
precision, recall, f1, _ = precision_recall_fscore_support(true_labels, predictions, average='binary')  # 'binary' if you have 0 and 1 labels

print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1 Score: {f1:.4f}")
```

```
Accuracy: 0.9906
Precision: 0.9905
Recall: 1.0000
F1 Score: 0.9953
```

Confusion Matrix

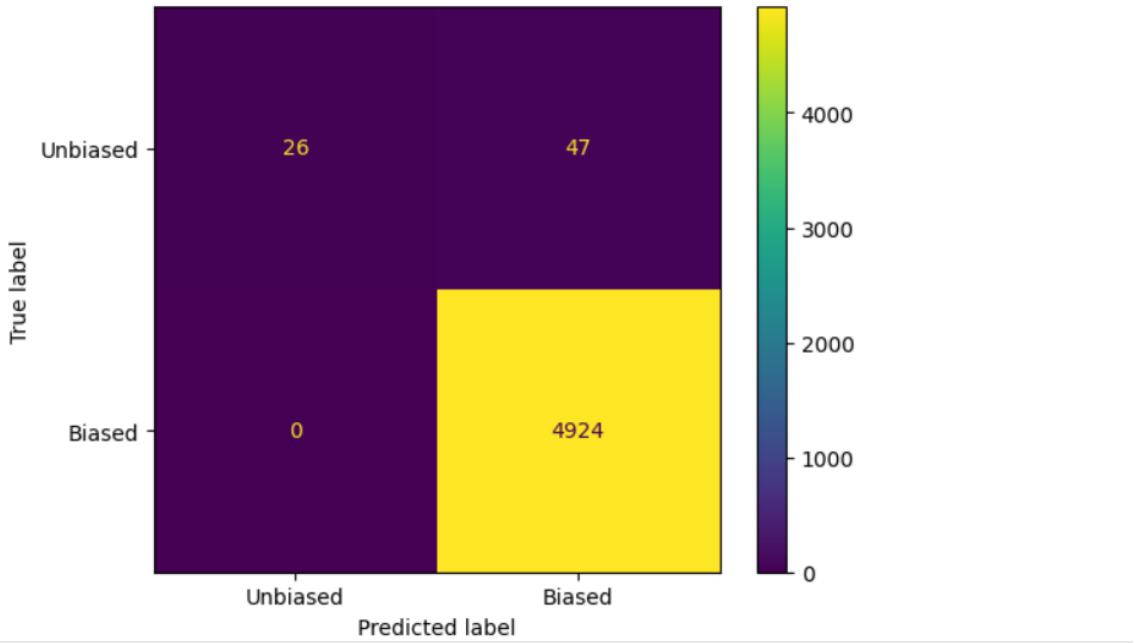
- Objective: Visualize classification results.
- Actions:
    - Constructed a confusion matrix to examine true positives, true negatives, and misclassifications.
- Models Used: distilbert-base-uncased.

Confusion Matrix

```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

cm = confusion_matrix(true_labels, predictions, labels=[0, 1])
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=['Unbiased', 'Biased'])
disp.plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x78ffadca10f0>
```



ROC-AUC (Receiver Operating Characteristic - Area Under Curve)

• Objective: Measure the model's ability to distinguish between biased and unbiased documents.

• Actions:

- Calculated an ROC-AUC score of 0.6781, highlighting areas for improvement in nuanced discrimination.

• Models Used: distilbert-base-uncased.

Analysis & Conclusion

The project achieved significant results, demonstrating the feasibility of automating bias detection in legal documents. The following metrics summarize the model's performance:

• Accuracy: 99.06%, indicating the model's overall correctness in predictions.

• Precision: 99.05%, reflecting its ability to correctly identify biased documents without misclassifying unbiased ones.

• Recall: 100%, showing the model successfully identified all biased documents in the test set.

• F1-Score: 99.53%, signifying a strong balance between precision and recall.

• ROC-AUC Score: 0.6781, indicating challenges in distinguishing subtle nuances of bias, though still useful for overt bias detection.

These results indicate that the model effectively detects clear instances of bias but may require further fine-tuning or domain-specific features to capture subtler forms.

Conclusion:

The Bias Detection in Legal Documents project successfully developed a machine learning pipeline capable of detecting bias with high precision and recall. The system showcases the power of advanced NLP techniques and pre-trained transformer models in addressing realworld challenges in the legal field. By automating bias detection, the project provides a scalable solution to promote fairness and neutrality in legal documents. Future work could focus on incorporating external knowledge sources, expanding the dataset, and fine-tuning contextual embeddings to improve the system's ability to capture subtle biases. This project highlights the transformative potential of AI in enhancing justice and equity in society.