A Study of Hate Speech Detection and Classification

1 Introduction

As part of my one and a half year course at OMU University, I am devoting my research project to a subject very current today, which tackles the issues of online etiquette and user-friendly experience on social media, online video games and online communities. This study explores the application of advanced Natural Language Processing (NLP) techniques to detect and classify hate speech using the HateXplain dataset. The primary objective is to develop an effective hate speech detection model utilizing the BERT (Bidirectional Encoder Representations from Transformers) architecture. [1] The HateXplain dataset is a benchmark dataset for explainable hate speech detection, consisting of 20k samples categorized into ”hateful,” ”offensive,” and ”normal” labels. [2]

2 Related research

2.1 BERT: Bidirectional Encoder Representations from Transformers

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained transformer-based model designed for natural language understanding tasks. Developed by Google, BERT marked a significant advancement in NLP by employing bidirectional training of Transformer models, which allows it to understand the context of a word based on its surrounding words. [1]

2.1.1 BERT Architecture

BERT's architecture is based on the Transformer model introduced by Vaswani et al. (2017) [1]. The key components include:

- **Self-Attention Mechanism**: This allows BERT to weigh the importance of different words in a sentence when encoding a particular word.
- **Bidirectionality**: Unlike previous models that read text sequentially (left-to-right or right-to-left), BERT reads the entire sequence of words simultaneously, thereby capturing richer context.
- **Layers**: The model comes in two main sizes: BERT-Base (12 layers) and BERT-Large (24 layers), where each layer is a Transformer block.

2.1.2 Pre-training and Fine-tuning

BERT undergoes a two-phase training process:

1. **Pre-training**: BERT is pre-trained on a large corpus of text (e.g., Wikipedia) using two unsupervised tasks [1]:
   - **Masked Language Modeling (MLM)**: Randomly masks some of the tokens and predicts the masked words based on their context.
   - **Next Sentence Prediction (NSP)**: Predicts whether a given sentence B follows sentence A, helping the model understand relationships between sentences.
2. **Fine-tuning**: BERT is then fine-tuned on specific downstream tasks like question answering, sentiment analysis, and named entity recognition by adding a small number of task-specific parameters. [1]

2.2 HateXplain

2.2.1 HateXplain Dataset Overview

HateXplain is a benchmark dataset for explainable hate speech detection. It was introduced to address the need for explainability in hate speech detection models. [2] The dataset includes:

- **Text Samples**: Collected from Twitter[1] and Gab [2].
- **Labels**: Each text sample is labeled as ”Hateful,” ”Offensive,” or ”Normal.”
- **Rationales**: Annotators highlight specific parts of the text that justify their labeling, providing explanations for each label.
- **Target Groups**: The dataset also includes information about which target groups (e.g., race, religion) are being attacked.

2.2.2 BERT and HateXplain

BERT can be fine-tuned on the HateXplain dataset to leverage its powerful contextual understanding for hate speech detection. Studies have shown that BERT-based models, when trained on HateXplain, outperform traditional machine learning models in identifying and explaining hate speech. [1] [2]

- **Performance Metrics**: Fine-tuning BERT on HateXplain has resulted in higher accuracy, precision, and recall compared to previous state-of-the-art models.
- **Explainability**: The use of BERT in conjunction with HateXplain's rationales has led to the development of models that not only detect hate speech but also provide human-understandable explanations for their decisions.

[1]https://twitter.com/
[2]https://gab.com/

3 Proposed Method

The proposed method involves a two-step approach to analyze the HateXplain dataset: a binary classification task to distinguish between normal and hate speech, combined with a dedicated offensive level calculation. This dual strategy provides categorical and quantitative insights into the content, improving the interpretability and effectiveness of the model. The key components of the method include data pre-processing, model training, evaluation, and offensiveness scoring.

Data Preprocessing: The raw text data from the HateXplain dataset undergoes preprocessing steps, including tokenization and removal of unnecessary labels, such as "offensive." Each sentence is further processed to enable both binary classification and offensiveness level scoring. For offensiveness scoring, a dictionary of offensive terms is used, and preprocessing accounts for variations in spelling and format.

Binary Classification: A pretrained BERT model is fine-tuned on the processed dataset to classify sentences as either hate speech or normal. The model is trained with binary labels derived from the majority vote of the annotators. This classification task provides a categorical decision about the nature of the content.

Offensiveness Level Calculation: In parallel, an offensiveness score is calculated for each sentence. The score represents the count of offensive terms. This step quantifies the degree of offensiveness, complementing the binary classification results.

4 Initial Dataset Analysis

4.1 Challenges

- Ambiguity in Language: Differentiating between normal and hate speech content can be subtle and heavily rely on context and nuances.
- Contextual Dependence: The meaning of posts can change based on external context, such as user history or cultural background, which is often not captured in the dataset.
- Inter-Annotator Variability: Different annotators may have varying interpretations of what constitutes hate speech or normal content, leading to inconsistencies in labeling.
- Include references at the end.
- Evolving Language: Hate speech and offensive language evolve over time, introducing new terms and phrases regularly, which static datasets may not effectively capture.

5 Model Development

5.1 Training Configuration

Training arguments were specified to control various aspects of the training process:

- **Training Arguments**:
  - **Number of Training Epochs**: 3
  - **Per-Device Batch Size**: 8
  - **Evaluation Strategy**: Conduct evaluation at defined steps.
  - **Learning Rate**: Adjusted dynamically using a warmup schedule.
- The `Trainer` class from the Hugging Face Transformers library was employed to manage the training and evaluation process efficiently. This class simplifies the handling of training loops, metrics calculation, and checkpoint saving.

5.2 Steps

5.2.1 Data Preprocessing

- **Tokenization**: Convert text samples into tokens using the BERT tokenizer to prepare the input for the model.
- **Label Mapping**: Simplify the classification task by eliminating "offensive" labels. The dataset was thus reduced to two classes: "normal" and "hatespeech."

5.2.2 Model Architecture

- **Binary Classification with BERT**: Utilize the `bert-base-uncased` model as the backbone for its superior language understanding capabilities. The model's final layer was adjusted to output probabilities for the binary classes (hatespeech or normal).

5.2.3 Training the Model

- **Fine-Tuning**: Fine-tune the BERT model on the HateXplain dataset to classify text as either hatespeech or normal.

5.2.4 Evaluation

- **Metrics**: Evaluate the model's performance using metrics such as accuracy, precision, recall, and F1-score to provide a comprehensive assessment.
- **Confusion Matrix**: Analyze the distribution of true positives, false positives, true negatives, and false negatives to identify areas where the model may need improvement.

5.2.5 Explainability

- **Model Decision Analysis**: While rationales were excluded from training, the offensiveness score adds interpretability by indicating the degree of offensiveness in the input, supporting the binary classification output.

5.2.6 Detailed Explanation

Data Preprocessing The text data from the HateXplain dataset is tokenized using BERT's tokenizer. This converts the text into a format suitable for input into the BERT model.

Model Architecture The `bert-base-uncased` model is used for its pre-trained language understanding capabilities. This model will be fine-tuned on the HateXplain dataset to adapt it to the specific task of hate speech detection.

Training the Model The BERT model is fine-tuned on the HateXplain dataset, including the use of rationales to help the model understand which parts of the text are important for the classification task.

Evaluation The model's performance is evaluated using standard metrics such as accuracy, precision, recall, and F1-score. These metrics help in understanding how well the model is performing in identifying hate speech. A confusion matrix is used to visualize the performance of the model across different categories, showing true positives, false positives, true negatives, and false negatives.

Explainability The rationales provided in the HateXplain dataset are used to highlight the parts of the text that the model focuses on for making its predictions. This enhances the explainability of the model, making it easier to understand why certain texts are classified as hate speech.

6 Model Training and Evaluation

This section summarizes the presentation and discuss the challenges that need to be addressed in the future.

6.1 Training Process

The BERT model was fine-tuned on the HateXplain dataset using the specified training arguments. The Trainer class managed the training loop, including forward and backward passes, gradient updates, and logging, ensuring an efficient and streamlined training process.

6.2 Evaluation

The model's performance was evaluated using metrics such as accuracy, precision, recall, and F1-score. The confusion matrix was employed to illustrate the model's performance across different classes, showing true positives, false positives, and false negatives for each class.

7 Results and Observations

7.1 Confusion Matrix Analysis

- True Positives (HateSpeech correctly predicted as HateSpeech): 863

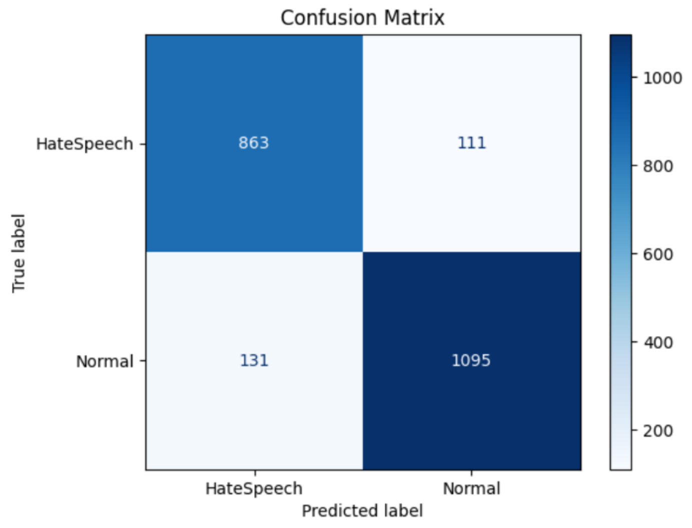

Figure 1: Confusion matrix of the BERT model fine tuned to the HateXplain dataset for binary classification

- False Positives (Normal incorrectly predicted as HateSpeech): 131
- True Negatives (Normal correctly predicted as Normal): 1095
- False Negatives (HateSpeech incorrectly predicted as Normal): 111

7.2 Example Misclassifications

- True Label: Hatespeech, Predicted Label: Normal: "me too but i still dis like jews and rather not have them follow me"
- True Label: Normal, Predicted Label: Hatespeech: "it was most certainly high even for secular jews during the ellis island days but i think the percentage of jews relative to their population that use welfare is higher than say asians or white catholics not sure though"

8 Offensiveness Level Calculator

The Offensiveness Level Calculator quantifies the offensiveness of a given text by counting the number of offensive words or expressions it contains.

8.1 Methodology

The offensiveness level is determined using a predefined dictionary created using Wiktionary English Vulgarities' list of offensive words and expressions. The key steps in the calculation are as follows:

- **Predefined Dictionary**: A curated list of offensive terms, including words, phrases, and their common variations, is used to identify offensive content. This dictionary is sourced from publicly available datasets and expanded to include misspellings, leet speak, and synonyms.

- **Offensiveness Level Calculation**: The score is computed as the total number of offensive words or expressions in the text.

$$\text{Offensiveness Level} = \text{Count of Offensive Words}$$

8.2 Evaluation and Examples

The following examples illustrate the functionality of the Offensiveness Level Calculator:

- **Example 1**: *"You are such a 4uck and an idiot."*
  - Identified Offensive Words: ["4uck", "idiot"]
  - Offensiveness Level: 2
- **Example 2**: *"Go back to your country, loser."*
  - Identified Offensive Words: ["loser"]
  - Offensiveness Level: 1
- **Example 3**: *"This is a completely normal sentence."*
  - Identified Offensive Words: []
  - Offensiveness Level: 0

8.3 Advantages

This updated method provides:

- A clear and interpretable measure of offensiveness as a simple count of offensive terms.
- Flexibility to incorporate updates to the offensive word dictionary and detection algorithms.

9 Challenges and Improvements

9.1 Misclassification Issues

The model exhibited few misclassifications. Subtle differences in language and context dependency contributed to these misclassifications.

9.2 Hatespeech Detection: Potential Improvements

- Data Augmentation: Increasing the diversity of the training data to better capture variations in language and context.
- Class Weight Adjustment: Balancing the impact of underrepresented classes to prevent model bias.
- Hyperparameter Tuning: Optimizing model parameters for better performance.
- Advanced Models: Exploring more sophisticated models like RoBERTa or BERT-large for potential performance improvements.
- Ensemble Methods: Combining multiple models to enhance accuracy and robustness.

9.3 Offensiveness Level Calculator: Potential Improvements

To further enhance the Offensiveness Level Calculator:

- Contextual analysis can be incorporated to identify implied or subtle offensiveness.
- The dictionary can be expanded with multilingual and culturally nuanced offensive terms.
- Advanced techniques, such as embedding-based similarity models, can detect offensive expressions not explicitly present in the dictionary.

10 Future Work

10.1 Cross-Domain Transfer Learning

Adapting the model to different datasets and domains to improve generalizability and robustness.

10.2 Explainability

Improving the model's ability to provide transparent and understandable explanations for its predictions, enhancing trust and usability in real-world applications.

11 Conclusion

This study demonstrates the potential of using BERT for hate speech detection, and calculation of offensiveness level. While the model shows promising results, there is room for improvement, especially in handling subtle and context-dependent language. Future work will focus on enhancing model accuracy, interpretability, and adaptability to different languages and domains. The ultimate goal is to contribute to the development of more effective and transparent hate speech detection models, promoting safer online communities.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Long and Short Papers*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[2] B. Mathew, P. Saha, S. M. Yimam, C. Biemann, P. Goyal, and A. Mukherjee, "Hatexplain: A benchmark dataset for explainable hate speech detection," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2020, pp. 3451–3463.