H3: Multiclass Hate Speech Detection

Group number: 30 | TA: Palani

Apoorva Arya
2020032
apoorva20032@iiitd.ac.in
Siya Garg
2020577
siya20577@iiitd.ac.in
Srijan Arora
2020342
srijan20342@iiitd.ac.in
Utkarsh Pal
2020144
utkarsh20144@iiitd.ac.in

Abstract

With the increasing use of social media, factors like Hate Speech and Offensive language also increase. The classification and segregation of hate speech from other instances of offensive language is a significant challenge for automatic hate-speech detection on social media. The first step in cumbering of these things is the detection of Hate and Offensive comments/tweets. We have attempted to classify them through this project.

1 Problem Definition

The task at hand is Multi-class Hate Speech Detection. Hate speech is used to refer to ideas expressed on public platforms which encourage or instigate hate or violence against a particular individual or a group of individuals based on their color, race, sex, age, or other demographic factors. While a publicly expressed opinion is referred to as Offensive speech if it doesn't necessarily hateful or threatening towards an individual or community - something can offend someone without it being hateful.

The problem given focuses on text sourced from Twitter. These tweets have been classified into three categories: Hate, Offensive, and Neither. Our objective is to classify the tweets into these 3 categories and, more specifically, to differentiate between Hate and Offensive samples.

2 Related Work

2.1 Automated Hate Speech Detection and the Problem of Offensive Language (Davidson et al.)

The paper highlights that the separation of hate speech from offensive language on online platforms is one of the most prominent issues of automatic hate speech detection. Supervised learning algorithms often confuse these two categories leading to low accuracy. The authors use a hate speech lexicon sourced from Twitter and labeled by human volunteers as one of Hate, Offensive, and Neither. The authors extract unigram, bigram, and trigram features, each weighed by their TF-IDF, and use supervised learning algorithms for their classification.
They discovered that sexist tweets are often categorized as offensive while racist and homophobic tweets are frequently labeled as hate speech, and the presence of hateful words most often detects the presence of hate speech, and their absence makes detection very difficult.

2.2 HateXplain (Mathey et al.)

The paper provides a benchmark hate-speech classification dataset with the same 3 labels as the current problem, along with providing the target community and the rationale (which section of text is the basis of the decision) for each.

2.3 Improving Random Forest Method to Detect Hate speech and Offensive Word (Nugroho et al.)

This study explored hate speech generated on social media platforms like Twitter, MySpace, and Facebook. They used conventional Machine Learning models on the textual data obtained from these sources. It focuses on improving the performance of the Random Forest classifier in detecting hate speech and differentiating it from offensive speech. The results obtained were put in comparison with those obtained when AdaBoost and Neural Networks were applied to this. According to this paper, Random Forest had the best performance, with AdaBoost following closely and with significantly lesser accuracy using Neural Networks.

3 Data

We were given a Twitter dataset containing tweets, their labels, and indices corresponding to each. The data contains lots of noise, which was removed through pre-processing. The removal of usernames and URLs, followed by the splitting of hashtags, which were abundant. Since hashtags can contribute towards hate detection, we split the hashtags using greedy dictionary lookups to preserve as much information as possible. Several tokens, which have no semantic meaning and don't contribute to the classification problem, such as '>' converted to '<' and 'RT' for retweet token, were removed. Other removals include extra punctuation, spaces, characters, etc. The data was split into training and testing with an 80-20 split.

Dataset Split(Training Set)		
Label	Class	Count
0	Hate	885
1	Offensive	12318
2	Neither	2657

Dataset Split (Validation Set)		
Label	Class	Count
0	Hate	634
1	Offensive	3080
2	Neither	252

Dataset Split (Augmented Set)		
Label	Class	Count
0	Hate	5935
1	Offensive	5480
2	Neither	7814

4 Methodology

4.1 Pretrained models

The HuggingFace library and transformers were used for creating contextual embeddings for our tweets. We directly employed these pre-trained models for the creation of embeddings.

4.1.1 BERT

The *bert-base-uncased* is a pre-trained model on the English Language based on Vaswani's Transformer model. It is a bidirectional transformer that was pre-trained in a self-supervised fashion using both Masked Language Model (15% of the tokens are masked and the model is tasked with predicting the masked token) and Next Sentence Prediction (2 sentences are passed to the model, and the model detects whether the second sentence follows the first sentence)

4.1.2 RoBERTa

Roberta-base is a pre-trained model on the English Language based on BERT, mentioned above. It fine-tunes the learning objectives of BERT, and is trained on a much larger dataset to achieve better performance than BERT.

4.1.3 XLNet

The *xlnet-base-cased* model is a pre-trained model on the English Language based on a novel generalized permutation language modeling objective. Its transformer is based on the Transformer-XL. It achieves state-of-the-art results on various downstream language tasks. It aims at maximizing the expected log-likelihood over all permutations of factorization order. It implements autoregressive training, a proxy to autoencoding helping in removing pretrain-finetune discrepancies and independence assumptions seen in BERT model.

4.1.4 Electra

Electra is a pre-training approach where two transformer models are trained: the generator and the discriminator. The generator replaces tokens randomly in a sequence and is thus trained as a masked language model. The model of interest, the discriminator, attempts to identify which tokens in the sequence were replaced by the generator. This approach significantly reduces the training time compared to previous language models like BERT.

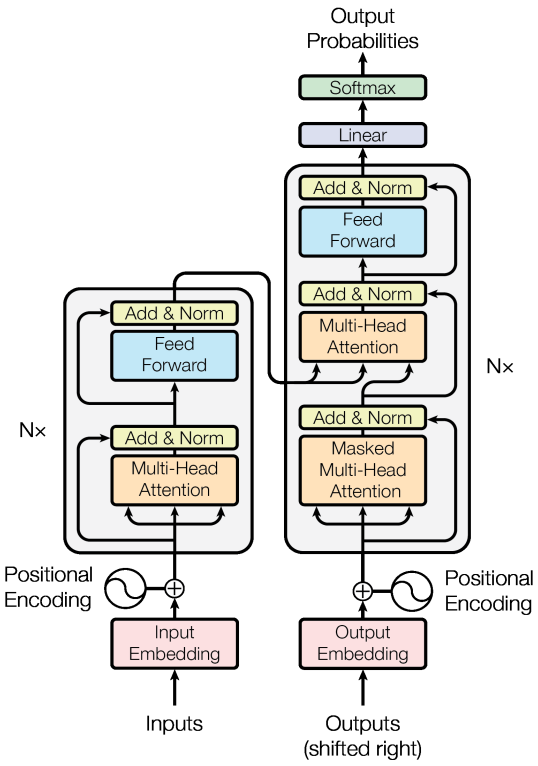

Figure 1: The transformer architecture which is used in all the pre-trained models.

4.2 Feed Forward

The output of the last hidden layer received from the pre-trained models are then passed to a fully connected layer. We tried different combinations of hidden layers and activations for fine-tuning. The optimizer used is Adam with learning rate of 2e-5 as recommended by the authors of BERT, with epsilon value of 1e-8. Dropout is applied to each layer with probability 0.2 to prevent overfitting of data. This network finally classifies the data into 3 classes: Hate, Offensive, and Neutral. Loss function used is Cross-Entropy loss.

4.3 Machine Learning Models

For final classification, we also tried to implement some machine learning algorithms like Random-Forest and AdaBoost on the contextual embeddings obtained our pre-trained models.

4.3.1 Random Forest

Random Forests are implemented by further developing the CART (Classification and Regression Tree) method by applying bagging and random feature selection in the algorithm. We apply the Random Forest classifier to training data, which is used with stratified k-fold cross-validation.

4.3.2 Adaptive Boosting (AdaBoost)

Adaboost is an ensemble method and, thus, gives more weight to weak classification. It combines a number of weak learners to form one strong classifier - here, the first model is built to learn the dataset while the others are built upon it to correct the errors present in the first one. Adaboost is used with stratified k-fold cross-validation and the results obtained are close to the ones we get using the Random Forest classifier.

4.4 Playing with Data

While training the model and running detections, we noticed misclassification in the Hate class was more than in the Offensive. The model was classifying the hate tweets as offensive tweets which led to lower accuracies. To create some bias we tried 2 methods to avoid this: Oversampling and Data Augmentation

4.4.1 Oversampling

In order to slightly mitigate the class imbalance between the offensive and the hate-speech classes, we re-sampled the hate-speech samples and put them again in our training data.

4.4.2 Data Augmentation

We took a publicly available dataset, HateXplain which has a multilabel classification similar to our problem. The dataset, similar to the one provided, contains text samples generated from public forums like Twitter and classified into the same 3 labels. We discard the target community and rationale parts as they do not directly correlate with the current task. The dataset is available for public use here.

5 Experimental Results

We deployed four different pre-trained models for our project with and without over-sampling and data augmentation. For the final label classification we used both feed-forward and machine learning models. The results are given in the table present.

6 Analysis

6.1 Base Models

The four contextual pre-trained models worked well on our dataset giving an accuracy of around 89-90% and Macro-F1 of around 0.75 - 0.76.
The BERT model is the base model for comparison with the other pre-trained models used. Roberta

Model	Epochs	Accuracy	Macro-f1	Oversampling	Augmentation
BERT base	5	0.900907	0.75409	False	False
BERT oversampled	1	0.91023	0.783254	True	False
BERT augmented	3	0.9041855	0.754608	False	True
RoBERTa base	3	0.903933	0.7683188	False	False
RoBERTa oversampled	2	0.893343	0.769643	True	False
RoBERTa augmented	4	0.895108	0.748825	False	True
XLNet base	4	0.8961169	0.7680548	False	False
XLNet oversampled	2	0.908976	0.776081	True	False
XLNet augmented	2	0.899825	0.768959	False	True
Electra base	7	0.9054462	0.7686989	False	False
Electra oversampled	4	0.901412	0.77263	True	False
Electra augmented	5	0.90242	0.759187	False	True
Random Forest	7	0.7252	0.7133	True	False
AdaBoost	4	0.6489	0.7081	True	False

Table 1: Experimental Results for base models and datasets used

shows a little improvement from BERT as it employs more efficient training methods.
XLNet uses bi-directional contextual embeddings which shows minor improvements from BERT and RoBERTa. These results are in line with the published literature. XLNet took a long time to run and requires higher computational power.
Electra is a newer model that significantly reduced the model's training time, taking less than half the time required for BERT and RoBERTa. Also, it gave good results.

6.2 Machine Learning Models

The results of machine learning models were not as good as feed-forward neural network. It has very low F1 and accuracy scores. Moreover, they took a long time for training and high computational power. Here, Random Forest gives us better results when compared to AdaBoost. AdaBoost, with its tendency to emphasize wrongly classified points, is able to give comparable F1 scores to Random Forest, despite having much less accuracy.

6.3 Oversampling

When comparing the results with and without oversampling, all models show an improvement in their F1 scores with the oversampled dataset. The improvement in the Bert model is the most significant at about 0.03, and is the only that can be considered an improvement beyond the margin of error. Oversampling worked well with our dataset and facilitated our aim of achieving higher F1 scores.

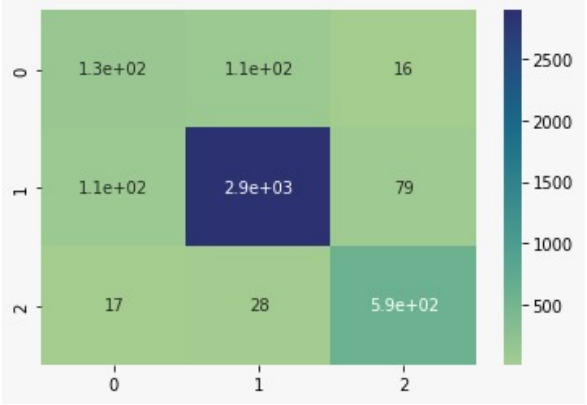

Figure 2: Confusion matrix for BERT with oversampled data

6.4 Data Augmentation

The augmented dataset gives results comparable to our original dataset, despite having nearly double the amount of training samples present. This implies that the datasets are not exactly compatible, and using both together is likely to only result in much higher training times. We concluded, data augmentation won't work on our dataset and contribute much towards the project solution.

6.5 Confusion matrix analysis

As observable in Figure 2, classes 1 and 2 are mostly predicted correctly. However, this is not the case with class 0, Hate speech. Prediction of

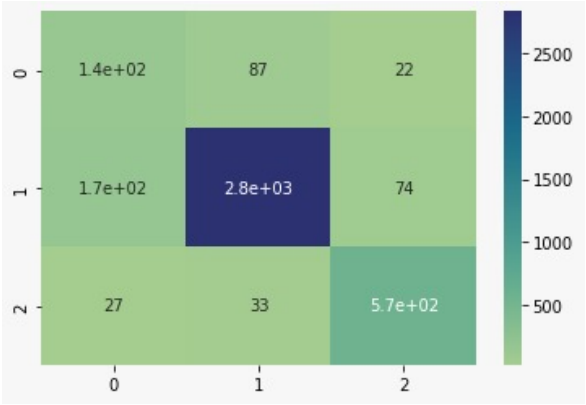

Figure 3: Confusion matrix for XLNet without oversampled data

class 0 is split evenly between classes 0 and 1, indicating the difficulty in correctly identifying hate speech from offensive speech. These results are in line with the literature mentioned.

As shown in Figure 3, XLNet mirrors the trend seen in BERT. However, it is better able to better classify offensive labels than BERT.

Contributions

The four members worked together to complete this project and have equal contributions towards it. Utkarsh Pal performed all the preprocessing of the given noisy data, obtained the augmented dataset and ran the models. Srijan Arora and Siya Garg brainstormed on the usage of various pretrained models for classification and writing its code. Apoorva Arya tried various ML models to be used on the top of contextual embeddings for classification and running the models.

References

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. ArXiv:1703.04009 [cs].

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2022. HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection. ArXiv:2012.10289 [cs].

Kristiawan Nugroho, Edy Noersasongko, Purwanto, Muljono, Ahmad Zainul Fanani, Affandy, and Ruri Suko Basuki. 2019. Improving Random Forest Method to Detect Hatespeech and Offensive Word. In *2019 International Conference on Information and Communications Technology (ICOIACT)*, pages 514–518, Yogyakarta, Indonesia. IEEE.

(Davidson et al., 2017) (Nugroho et al., 2019) (Mathew et al., 2022)