A DistilBERT-Based Transfer Learning Approach for Hate Speech Detection

Margarita Geleta
University of California, Irvine
mgeleta@uci.edu

Alekhya Pyla
University of California, Irvine
apyla@uci.edu

Pablo Martin Redondo
University of California, Irvine
pmartinr@uci.edu

Abstract

Identifying offensive contributions and hate speech would allow for the creation of safer online communities and social media spaces. This project aims to address the problem of identifying toxic comments on the web using English-only text. The latest methods perform sentiment analysis combining text features with neural network-based classifiers. Usually, text featurization models require large amounts of computational resources for training. We propose using a transfer learning approach based on the distilled version of BERT, DistilBERT, which is lighter in comparison to other state-of-the-art models, allowing deployment under low latency constraints. Different machine learning methods have been explored for classification: Random Forests, Support Vector Machines, or Multi-Layer Perceptrons. The main challenge has been the unbalancedness of the data with an inherent bias towards non-toxic comments. Moreover, the data contains inconsistencies in the labeling due to the manual tagging and the subjectivity of the task. Despite these caveats, our solution has been successful in obtaining high precision in the predictions with the additional relaxation of computational power requirements compared to previous methods.

1 Introduction

A single tiny toxic comment is capable of bursting an online discussion. A succession of toxic comments can lead to cyberbullying, grooming, and violence. With the growth of social media and its impact on people's lives, there is a need to quickly detect comments that attack other individuals' integrity. This has been a problem that has increasingly posed a challenge for tech companies in times of pandemic and post-pandemic crises [6]. For this reason, this research aims to design, train and evaluate a natural language processing system capable of classifying toxicity comments. The study has been extended into building a machine learning model with English-only training data with two main components: a text encoder and a classifier.

Previous work on hate speech detection systems used transformer models that required powerful computational resources to train them from scratch. As an improvement, a transfer learning-based approach has been introduced to extract knowledge from a source setting and transfer it to a different target setting. In this work, we propose using a pre-trained transformer's weights trained on the same domain but for a different task. The pre-trained model is used as a generic feature extractor, yielding "off-the-shelf" features and on top of that, a task-specific classifier is placed (in this case, for toxicity detection) to adapt the whole network to the target task. According to [3], latent representations of text are likely to be transferable to semantically different tasks. The ultimate enhancement has been to fine-tune the network by using backpropagation with labels from the target domain until the validation loss starts to increase (the model begins to overfit).

Preprint. Under review.

In this project, DistilBERT is used to extract the latent representation of the text. Although this model is lighter in comparison to the original BERT transformer [5], still it resulted in a big computational effort to use the whole training set with every method taking close to 24 hours in training on GPU.

After fine-tuning DistilBERT weights for toxicity classification, we placed different machine learning models on top of it. The performance against multiple classifiers has been compared to identify the best-performing one. We obtain results close to the state-of-the-art (top Kaggle models).

2 Data

The data used in this paper is English-only text comprising 396,710 comments from Wikipedia talk page edits and Civil Comments plugin, gathered by the Google Jigsaw team and made public through the data science platform Kaggle. To have enough toxic samples, we joined two sources, namely, the Jigsaw Unintended Bias in Toxicity Classification and Toxic Comment Classification Challenge datasets. The labeling of the comments was done via crowd-sourcing, meaning that each comment was rated by different people and the tagging is not completely accurate, i.e., some labels are inconsistent and subjective to the people tagging them.

There are many examples of identity hate, threats, and insults that are not marked as toxic, and a question to ponder is how this inconsistency in labeling can impact the ability of the classifiers to get higher scores. We have performed general cleaning of the data (such as removing numbers, extra spaces, etc.) but we have not cleaned in terms of correcting the inconsistent labels (Table 1), since this has not been the purpose of this project. However, it is a good point to keep in mind.

Text	Comments
How could I post before the block expires? The funny thing is, you think I'm being uncivil!	Tagged as non-toxic.
You should be fired. It makes me sick that people like you exist in this world.	Tagged as insult (toxic).
I know where you live; I will kill you if you revert any more changes!	Tagged as threat (toxic).
F*** off You are NOT an administrator. You DON'T have the authority to tell me what to do.	Tagged as obscene (toxic).
You socialist bastard! You filthy Hippy! You dare touch my edits again.	Tagged as identity hate (toxic)
/´¯/) \n ,/¯// \n // / \n/´¯/"/´¯¯‘•¸ \n /'////¨¯\\\n('(´(´,~/") \n\\\\/ \n_ • ´ \n\\(\n\\\\	Tagged as non-toxic. In fact it is an ASCII artwork representing the middle finger offensive sign.

Table 1: Samples from the dataset.

The main challenge while working with such data is that the data is highly imbalanced with a paucity of toxic records (91.8% non-toxic and 8.2% toxic) which creates an inherent bias towards non-toxic comments. To overcome this, we randomly sampled 12.3% negative samples from the non-toxic comments pool and concatenated this subset with the toxic comments, this way normalizing the classes distribution to 40% toxic and 60% non-toxic comments.

3 Methods

3.1 Representation of the sentences in a latent space

The first stage of the task consisted in representing the input sentences in feature space with better separability. For this goal, we have fine-tuned DistilBERT [5], a bidirectional transformer based on BERT. BERT [1] is a sequence model that applies bidirectional training of Transformer, a celebrated attention model. Bidirectional training outstands from previous efforts which either looked at a text sequence from left to right or combined left-to-right and right-to-left training, allowing for each word representation to contain information about the surrounding context. Results from [1] show that a language model which is bi-directionally trained can have a deeper sense of language context and flow than single-direction language models.

One problem of this transformer is its huge size – 340M parameters, making the task of fine-tuning possible only to teams with big computational resources. DistilBERT is a distilled version of BERT,

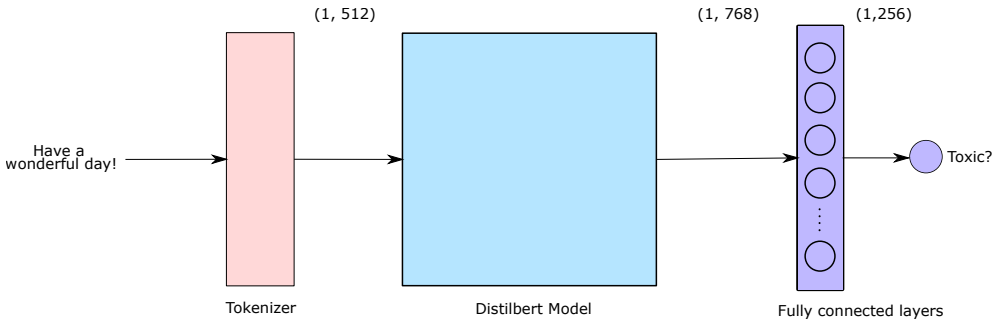


Figure 1: Setup for DistilBERT fine-tuning.

achieving a reduction of size by 40% while retaining 97% of its knowledge understanding and making it 60% faster [5].

To use this model as the feature extractor of the sentences we first need to tokenize the input (Figure 2). The tokenizer wraps the sentence around [CLS] and [SEP] special tokens and then converts the sentence to a sequence of integers by, first, splitting the sentences into tokens, and then applying a sub-token discovery and generation algorithm (such as Byte-Pair Encoding or Word Piece). After that, we use transfer learning to fine-tune DistilBERT for our task. To do this, we have built a neural network classifier on top of it and trained the whole model for 2 epochs (Figure 1).

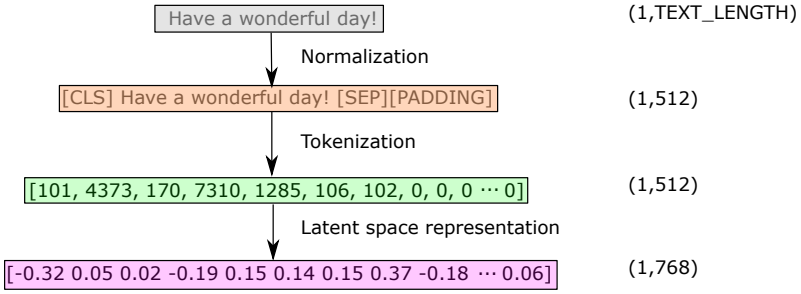


Figure 2: Text transformation process.

We apply dropout regularization to the features yielded by the transformer, ignoring a random 20% of them in each mini-batch iteration. Then, we use a fully connected layer of 256 neurons. We apply batch normalization to the output of this layer to ease the gradient flow. We then forward it through a ReLU activation function and apply a dropout of 20% again. Ultimately, we obtain a hidden unit with a final fully connected layer with 1 neuron and apply the sigmoid activation function to obtain the toxicity probability (Figure 3).

Figure 3: Architecture of the classifier on top of the transformer during the fine-tuning task.

To train this network, we are optimizing over the Binary Cross-Entropy loss by minimizing the batch error between the true toxicity label y and the prediction $\hat{y}$ (Equation 1), using mini-batch gradient descent and QHAdam with batch size $B = 64$.

$$L = -\frac{1}{B}\sum_{i=1}^{B} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \tag{1}$$

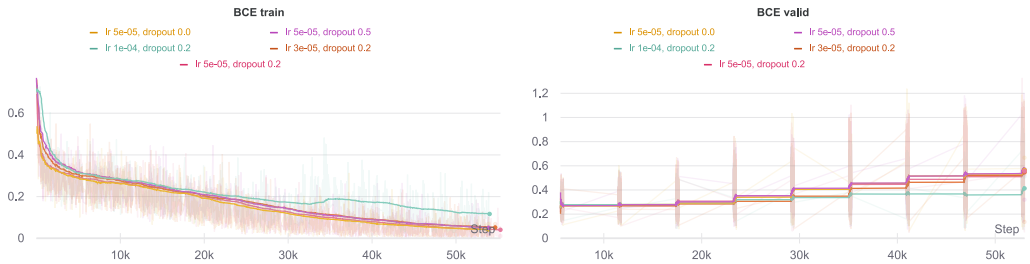
In QHAdam, Adam's momentum estimators are replaced with a momentum calculated via the Quasi-Hyperbolic Momentum Algorithm [2]. QHAdam is parametrized by $\alpha, \epsilon \geq 0, \beta_1, \beta_2 \in [0, 1)$, and $\nu_1, \nu_2 \in \mathbb{R}$, uses the update rule:

$$g_{t+1} \leftarrow \beta_1 \cdot g_t + (1-\beta_1) \cdot \nabla \hat{L}_t(\theta_t) \qquad g'_{t+1} \leftarrow \left(1-\beta_1^{t+1}\right)^{-1} \cdot g_{t+1}$$

$$s_{t+1} \leftarrow \beta_2 \cdot s_t + (1-\beta_2)(\nabla \hat{L}_t(\theta_t))^2 \qquad s'_{t+1} \leftarrow \left(1-\beta_2^{t+1}\right)^{-1} \cdot s_{t+1}$$

$$\theta_{t+1} \leftarrow \theta_t - \alpha \left[\frac{(1-\nu_1) \cdot \nabla \hat{L}_t(\theta_t) + \nu_1 \cdot g'_{t+1}}{\sqrt{(1-\nu_2)(\nabla \hat{L}_t(\theta_t))^2 + \nu_2 \cdot s'_{t+1}} + \epsilon} \right]$$

We have trained this network with the recommended learning rates $5 \cdot 10^{-5}, 3 \cdot 10^{-5}$ by Devlin et al. [1] and we have also tried a larger learning rate 10^{-4}. In all theses cases, we have used QHAdam with zero weight decay, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $v_1 = v_2 = 0.1$. Different learning curves for each learning rate are shown in Figure 4. Our default settings in this report correspond to learning rate $\alpha = 5 \cdot 10^{-5}$.



(a) Training curves.

(b) Validation curves.

Figure 4: Learning curves of DistilBERT fine-tuning on the toxicity classification task.

After every training epoch, we computed the loss with a validation set. We always save the best model weights during validation. The fine-tuning process showed a satisfactory progression of the learning curves. As seen in Figure 4, the loss decreases very fast, which makes DistilBERT start overfitting after the second epoch.

3.2 Classification

Having the latent representation of the text, we proceeded to explore different classification methods to compare their effectiveness in this specific task.

3.2.1 Random Forest

The first explored model has been random forest. Before applying ensemble learning with the random forest algorithm, we performed experiments to see how we should tune our decision trees to get the best performance. There is no normalization or scaling required in this method.

After training some classifiers, we realized that the representation of the text given by DistilBERT is redundant: once we have added enough features to the tree, the mutual information between the output and the remaining features is 0 ($I(x_i, y) = 0$). This happens using less than 60 features. Hence, training trees with a maximum depth higher than 60 would make no difference (Figure 5).

Considering the upper bound of decision tree depth as stated earlier, we explored the maximum depth parameter that would give us the best performance. For this task, we have trained decision trees of different depths using 5-fold cross-validation. This way, we have tuned the best maximum depth parameter of our random forests classifier.

We have chosen 10 to be the maximum depth of our tree, as it is the one with the best score (Figure 6). After that, we have applied ensemble learning with random forests to get the best possible performance of the decision trees classifier. We have fine-tuned the number of classifiers that gives us the best scores using cross-validation (Figure 7).

As it can be seen in Figure 7, the best performance is achieved with 50 classifiers. Hence, we use a random forest model with 50 trees of a maximum depth of 10. Using this configuration, the final test evaluation has shown good performance, with an F-1 score of 93.26%.

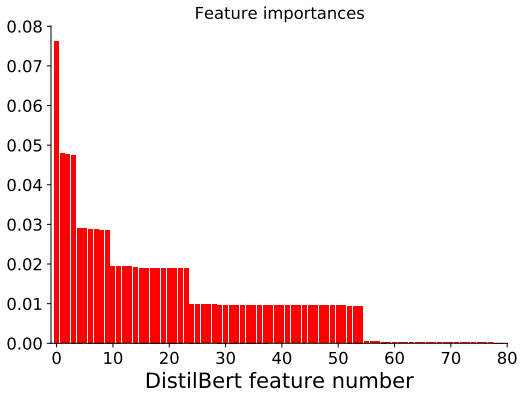


Figure 5: Importance of the features added to a trained decision tree.

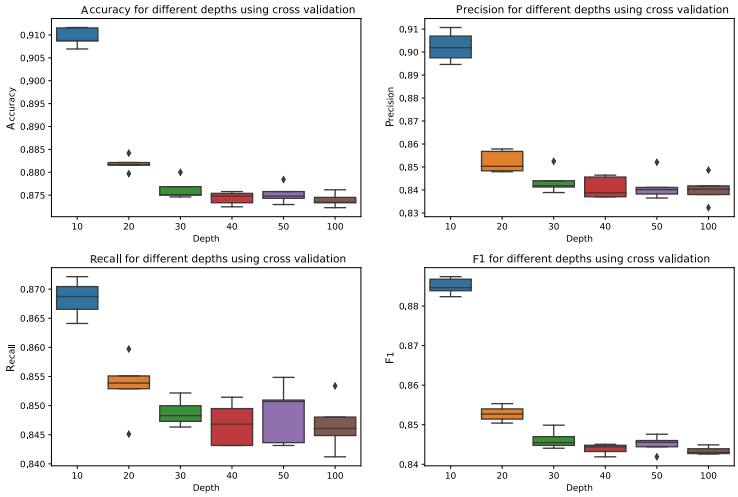


Figure 6: Box plots of the distribution of the cross validation scores for trees of different maximum depths

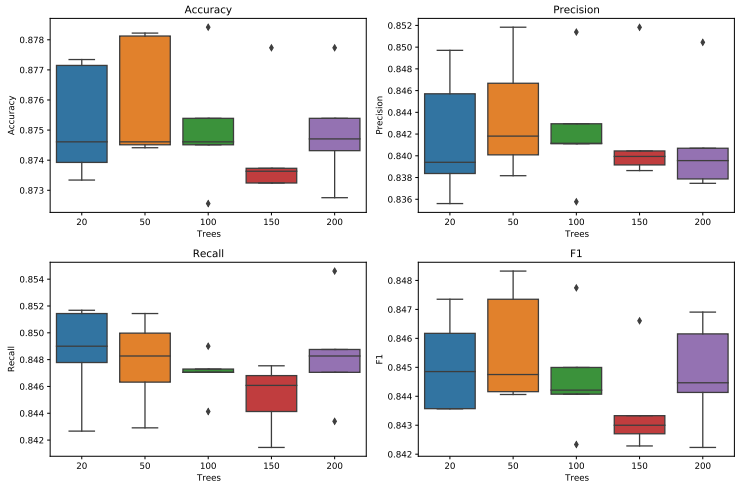


Figure 7: Box plots of the distribution of the cross validation scores for random forests for different number of decision tree estimators.

3.2.2 Support-vector machine

As the next method, we have experimented with Support Vector Machines as it is advantageous in high dimensional spaces. SVM works on the principle of the ability to linearly separate data by maximizing the margin of the decision boundaries. We have transformed our DistilBERT latent vectors of 768 components with polynomial and Radial Basis Function (RBF) kernels for SVM.

A polynomial kernel and RBF kernel functions (Equations 2 and 3) are defined as follows:

$$K(x, x') = (1 + x')^d \tag{2}$$

$$K(x, x') = e^{-\gamma \cdot ||x - x'||^2} \tag{3}$$

where x, x' are any two feature vectors of the model. d is the degree of the polynomial. $||x - x'||^2$is the squared Euclidean distance between two feature vectors and γ is a scalar that defines the influence of a single training example.

To identify the best hyper-parameters and performance of the model, cross-validation is performed for varying polynomial degree d from 1 to 6 and γ from -1.5 to 10 (in logarithmic scale) in the respective models as shown in Figures 8 and 9. The best hyperparameters found for polynomial and RBF kernels are $d = 2$ and $\gamma = 0.03$, respectively.

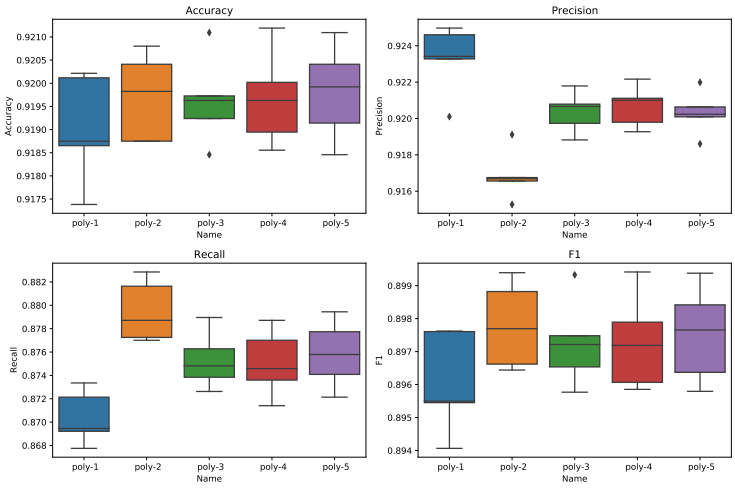


Figure 8: Box plots of the distribution of the cross validation scores for polynomial SVM.

The test evaluation for a polynomial SVM with a degree of 2 provides an accuracy of 91.8% and an F1 score of 93.26%. On the other hand, an RBF kernel SVM with $\gamma = 0.03$ underperforms with an accuracy of 88.0% and an F1 score of 89.9%, which signals that the model does not generalize sufficiently well.

3.2.3 Neural Network

Neural Networks have been also tested for the task of classification considering that Neural networks can approximate any function given enough hidden layers. After fine-tuning the DistilBERT model for this specific use case, a two-layer perceptron classifier of 256 and 64 neurons each was placed on top of the transformer (Figure 10).

Same as before, we use batch normalization and ReLU activation functions, but this time without any dropout (Figure 11). At the output of the network, we place a Sigmoid to obtain toxicity probabilities classification as 0 or 1. We perform a 5-fold cross-validation optimizing with a QHAdam optimizer with the same hyperparameters as in the fine-tuning task. The only exception, in this case, is that a larger learning rate is used in the comparison. ($\alpha = 10^{-4}$). When the model begins to overfit and does not improve in terms of validation loss in two consequent epochs, we stop the training.

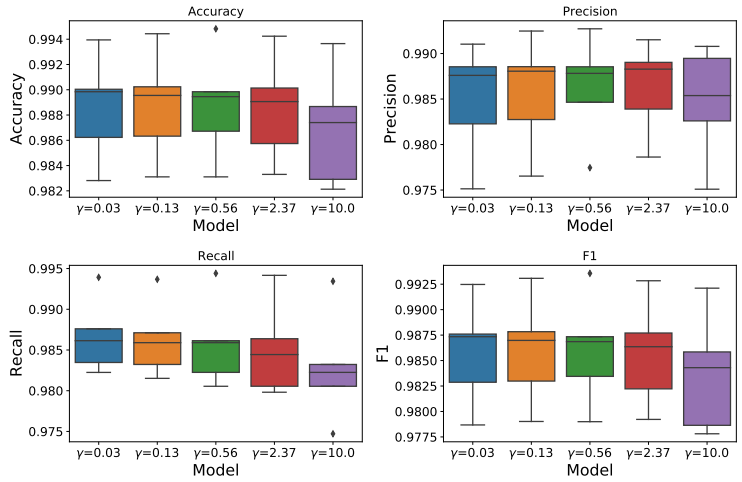


Figure 9: Box plots of the distribution of the cross validation scores for Radial-Basis Function SVM.

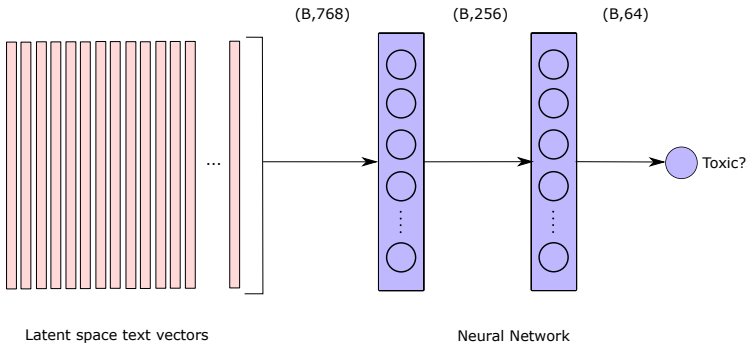


Figure 10: Architecture of Neural networks built on FineTuned Model.

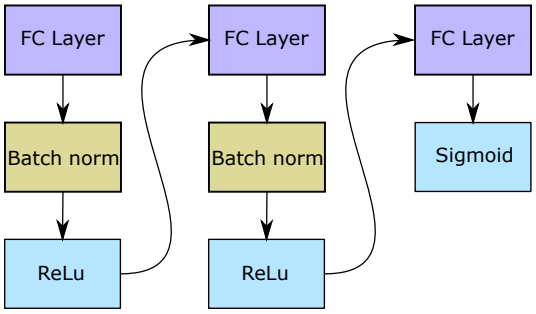


Figure 11: Configuration of Neural classifier built on top of the fine-tuned model.

The experiment resulted in an accuracy of 88% and an F1-score of 90% which turned out to be less effective compared to the previous classification methods of random forest and polynomial SVM.

4 Results

4.1 Exploratory Data Analysis

Since analyzing such high dimensional data is computationally complex, exploratory data analysis on the latent variable data from DistilBERT has been performed using Singular Value Decomposition (SVD) for both the training and test datasets (Figures 12 and 13). It can be observed that the data points appear to be separable in the high dimensional space, implying the effectiveness of the models built in classification.

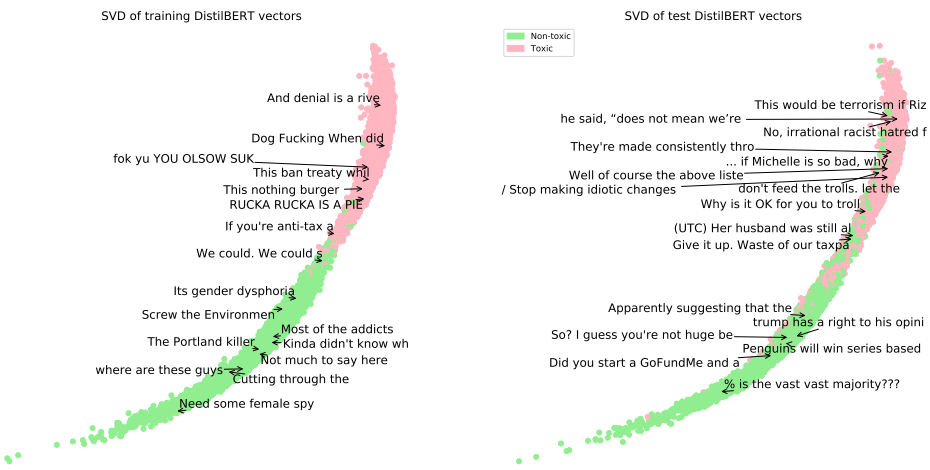


Figure 12: Relationship between the first and second SVD components of 768 DistilBERT features. The colors correspond green and red correspond to labels "non-toxic" and "toxic", respectively. The left plot corresponds to the training data projection and the right one, to test data. Test data shows more overlapping.

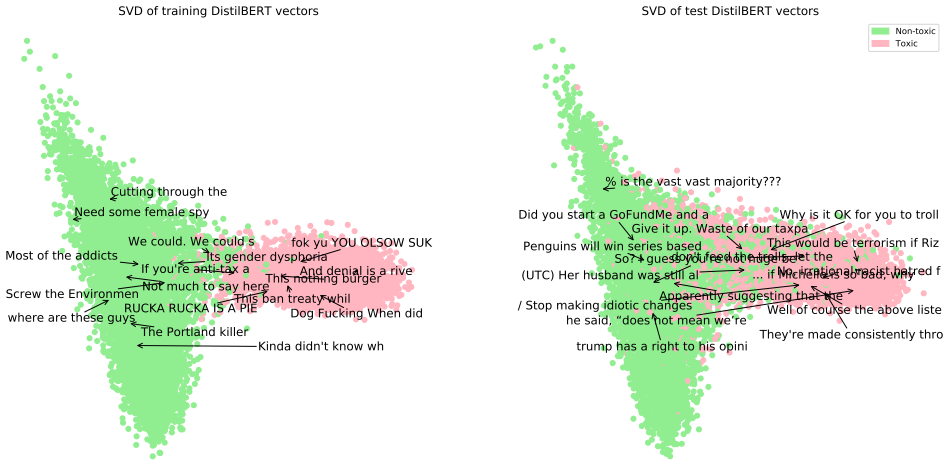


Figure 13: Relationship between other the second and third SVD components of 768 DistilBERT features. The colors correspond green and red correspond to labels "non-toxic" and "toxic", respectively. The left plot corresponds to the training data projection and the right one, to test data. Test data shows more overlapping.

4.2 Comparison of Results

Each of the models discussed has been trained, validated, and tested on the data. The hyperparameters have been chosen with cross-validation. The models with the best-performing hyperparameters are shown below:

Method	Accuracy	Precision	Recall	F1 score
Neural Network(MLP)	0.88123	0.89973	0.90083	0.90028
Random Forest	0.91819	0.94541	0.92021	0.93264
Support Vector Machine - Polynomial	0.91819	0.94513	0.92045	0.93263
Support Vector Machine - RBF	0.88080	0.90107	0.89888	0.89998

The results manifest that our solution achieves considerable performance on the datasets in terms of accuracy, precision, and recall. The methods in general have proven to have high precision. This inherently means that a higher proportion of comments tagged as toxic are correctly classified as toxic

and prevents tagging normal text as toxic, which could help in preserving the right to freedom of speech for the users on any online platform. Comparing the results of multiple methods, the Random Forest algorithm is the best performing classifier for toxicity detection with high values across all evaluation metrics.

5 Conclusion

The solutions proposed in this paper proved to be as good as BERT-based methods in classifying toxic comments [4]. The primary advantage of this solution is the use of less computational power than the former one. This would in turn enable models to be easily deployed in production environments and under low latency constraints. It also would allow models to be deployed on the edge rather than as cloud service APIs, enabling high privacy for the user. Furthermore, it is a favorable step in eliminating the environmental cost of scaling more and more computational resources.

Moving ahead, this work can be extended further into identifying the granular categories within the data such as identity hate, severe toxicity, threat, etc. Alongside, it can also be considered to identify toxic patterns in the data as described in Table 1. Finally, more fine-tuning strategies could be explored.

References

[1] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

[2] J. Ma and D. Yarats. Quasi-hyperbolic momentum and adam for deep learning. *CoRR*, abs/1810.06801, 2018. URL `http://arxiv.org/abs/1810.06801`.

[3] L. Mou, Z. Meng, R. Yan, G. Li, Y. Xu, L. Zhang, and Z. Jin. How transferable are neural networks in nlp applications?, 2016.

[4] M. Mozafari, R. Farahbakhsh, and N. Crespi. A bert-based transfer learning approach for hate speech detection in online social media. In H. Cherifi, S. Gaito, J. F. Mendes, E. Moro, and L. M. Rocha, editors, *Complex Networks and Their Applications VIII*, pages 928–940, Cham, 2020. Springer International Publishing. ISBN 978-3-030-36687-2.

[5] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[6] UNICEF. Covid-19 and its implications for protecting children online. `https://www.unicef.org/sites/default/files/2020-04/COVID-19-and-Its-Implications-for-Protecting-Children-Online.pdf`, 2020. Accessed: 2020-04-25.