Offensive Language Identification Using Ensembling

Ishan Pandey	Arunim Gupta	Mehul Singh	Mansi Sharma
2020304	2019025	2020081	2020444

ABSTRACT

In this new technology age, social media has become the new platform for political and social discourses and discussions. But anonymity offered online has led to a surge in profane and abusive language online, affecting everyone globally across all ages alike. This calls for automated text classification tasks, and we attempt to do this using fine-tuning pre-trained transformer networks like ALBERT. Though test-specific training data sets are generally fine-tuned in a supervised way, they can also be done unsupervised by pre-training the masked language modeling task(MLM). We compare the effectiveness of offensive text classification of pre-trained transformers with and without MLM fine-tuning.

RELATED WORK

Offensive language detection has always been a hot topic among researchers, especially in social media. Initial architecture for such detection started with basic techniques like n-gram, lexical and morphological analysis, and POS tagging. The introduction of improved hardware and algorithmic components onset the domination of machine learning techniques along with dense word embeddings, which improved immensely over the previous work.

The beginning of pretraining and rectification of vanishing and exploding gradients put Deep Learning techniques as the most successful mechanism of capturing text context. This includes Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), which have provided high precision in such performance. Where these models were still a little short on an application-based system, LSTM and GRU came out to be the cream layer of the models to make such systems applicable on large platforms. However, such models still had the scope for improvement to a great extent.

Fine-tuning of pre-trained models and modern embedding techniques have yielded baseline results for text classification, predominantly applied to such tasks. Fine-tuning is performed on task-specific training datasets in both supervised and unsupervised manner using the masked language modeling (MLM) task. Although such techniques have made a giant leap in this domain, these techniques still need to catch up in the application since Autoencoder learning does not capture the full extent of the masked text in such models.

Where BERT falls short of classification, variations of the improved BERT model, ALBERTA and RoBERTa, have aided in accuracy and latency. On the other hand, models such as multilingual-BERT and XLM-RoBERTa have significantly been used in multilingual and cross-lingual data representations.

These models are powerful, yet accumulation can compensate for weaknesses if we utilize a "voting system" type model, i.e., ensembling.

To capture the full extent of such a task, ensembling has been a very effective way to compensate for weaknesses in each model. Ensembling is done using a few DL models and capturing a majority vote using specific metrics to classify the text. Different BERT variations and XL transformer additions have been tested for such tasks. On many tasks, pre-training has been done on DL models, but the classification is done using classic ML models, which gives quite a result. But, an ensemble based on the ALBERT model achieves the best overall performance, as we have presented in our best score.

DATASET DESCRIPTION

The dataset for the given task is comprehensive data, comprising 33% OFF data and the rest NOT. The train set consists of 13,207 labeled tweets, and the test set consists of 849 tweets. 80-20% ratio has been maintained for the train and dev set for this task. NOT classification is done for neutral or positive tweets and OFF, being offensive, for offensive language.

We further augmented the training set using the OLID 2020 Task A training data.

Text preprocessing has been thoroughly done and cross-checked. We have used various libraries and algorithms to achieve clean and processable text to construct meaningful word embeddings.

Following are the pre-trained transforms models we have modified and worked upon:

1) **BERT**
   It stands for 'Bidirectional Encoder Representations from Transformers'. This has been the basis for numerous future models. This language model is based on the transformer's architecture. This model impressively uses the 'Attention' mechanism to find relations and the extent of dependencies between tokens in a sentence. It masks 15% of the original tokens in the sentence and then tries to predict them to train. This mechanism is known as 'Masked Language Modelling' (MLM). It also randomly takes a pair of sentences and tries to predict whether the second sentence is the successor of the first in the original document to train. This is known as the 'next Sentence Prediction' (NSP) task.

2) **RoBERTa**
   It stands for 'Robustly optimized BERT,' and its model is based on a transformer. It was trained on numerous datasets, including 'OPENWEBTEXT,' 'CC-NEWS,' and 'STORIES'. Its architecture is mostly the same as BERT, with the following significant differences:

a) Larger batch sizes with longer sequences and neater data are used for training.
b) Unlike BERT, we don't have the 'Next Sentence Prediction'(NSP) task.
c) We keep changing the masking pattern dynamically.

3) **XLMRoberta**
It is a multi-lingual model, a modified version of RoBERTa, and uses over a hundred languages to train. Its architecture is also transformer based. It has proved highly effective in cross-lingual tasks like classifying 'Hinglish' hate speech. It is more or less equivalent to RoBERTa, with the main difference being that it is trained on data from the 'CommonCrawl' dataset, which has texts from over 100 languages.

4) **ALBERT**
It stands for 'A lite BERT for Self-supervised Learning of Language Representations.' It is also a modified BERT with the primary objective of improving training time issues and decreasing memory losses. Its architecture is mostly the same as BERT, with the following significant differences:
a) Instead of BERT's simple NSP task, it uses 'Sentence Order Prediction' to evaluate inter-sentence coherence loss.
b) The embedding parameters are divided into smaller matrices, which are then separately projected to the hidden space.
c) The learned parameters are stabilized and made more efficient by sharing them with all layers.

METHODOLOGY AND RESULTt:

Text Augmentation
We experimented with text genie for augmenting every 5th sentence being augmented (t5 paraphrasing) and found no improvement in results, so we proceeded without text augmentation.

We explore two questions related to the fine-tuning of pre-trained transformer networks for offensive language detection: which pre-trained model performs best on the task, and how much language model fine-tuning on in-domain data before classification fine-tuning improves the performance of the best model?

Model Selection of Transformer Networks
To answer these questions, we evaluate the performance of the BERT (Bidirectional Encoder Representations from Transformers) model (Devlin et al., 2019), which uses an attention mechanism to learn contextual relations between (sub-)words in a text sequence.

Masked Language Model

Sun et al. (2019) demonstrated that additional pre-training of BERT with the masked language model task could enhance subsequent supervised task-specific fine-tuning. We investigated within-task, in-domain,

and cross-domain further pre-training. Results showed that the first strategy is prone to overfitting the training set and may reduce classification performance. The last method is not beneficial since BERT is already trained on general-domain data. In-domain further pre-training, however, can improve later classification performance if there is considerable overlap in language characteristics between further pre-training and supervised training data. Consequently, we only do MLM pre-training on the original data. We remove URLs and user mentions from tweets, eliminate duplicates and randomly select 5 % of the OLID 2020, i.e., 436.123 tweets, for further pre-training (Using external data).

We pre-trained the assumed optimal model RoBERTa-large (Liu et al., 2019b) for one epoch (batch size 16 and learning rate 5e-5), but the result was not the best. Therefore, we proceeded with our final approach.

Ensembling

We tested BERTbase and BERT-large (uncased), RoBERTa-base and RoBERTa-large, XLM-RoBERTa, and four different ALBERT models (large-v1, large-v2, xxlarge-v1, and xxlarge-v2) on the official test set submission as team Performance (in %). We fine-tuned the models with the training data using the corresponding test data for validation and trained each model for six epochs with a learning rate of 5e-6, a maximum sequence length of 128, and a batch size of 16. After every epoch, the model was evaluated on the validation set, and the results of the baselines, single models, and ensemble models on the test set were recorded.

The best-performing epoch was saved for ensembling. We tested two approaches: a majority vote from all models and a majority vote from one model type with different parameter sizes, such as BERT-base and BERT-large. To learn from the entire dataset and to reduce the instabilities of predictions from random effects during model training, we also aggregated predictions using every epoch of training. This involved fine-tuning the MLM pretrained RoBERTa-large model each time with 90% of the OLID data for training and the remaining 10% as a validation set. The table below shows that all ensembles consistently perform better than the individual models. The ensemble averaging the predictions from the two ALBERTxxlarge models achieved the highest F1-score of 92.25%. Post-submission experiments revealed that the ALBERT-based ensembles would have beaten the first-ranked submission.

Model	Learning Rate	F1 Score
Bert Base	5e-5	0.855
Roberta Large	5e-5	0.869
Albert Large	5e-6	0.920
Albert XXLarge-V2	5e-6	0.938
Albert Ensemble Max Vote	5e-6	0.944

Contributions:

Ishan Pandey: Methodology and training
Mansi Sharma: Data Augmentation, Preprocessing, and External Data research
Mehul Gupta: Literature review and Model training
Arunim Gupta: Hyperparameter Tuning and Model training

References:

Ansari, M., Beg, M., Ahmad, T., Khan, M., and Wasim, G. (2021). Language identification of hindienglish tweets using code-mixed bert.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. CoRR, abs/1907.11692.

Marzieh Mozafari, Reza Farahbakhsh, and Noel Crespi. 2019. A BERT-based transfer learning approach for hate ¨ speech detection in online social media. In Hocine Cherifi, Sabrina Gaito, Jose Fernendo Mendes, Esteban ´ Moro, and Luis Mateus Rocha, editors, Proceedings of the 8th International Conference on Complex Networks and their Applications, pages 928–940, Lisbon, Portugal.

J. Devlin, M.-W. Chang, K. L. K. T. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. [Lan et al.] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. CoRR, abs/1909.11942.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval). In Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval), MN, USA. ACL

Nayak, R. and Joshi, R. (2022). L3cubehingcorpus and hingbert: A code mixed hindi-english dataset and bert language models. arXiv preprint arXiv:2204.08398.

Pitsilis, G. K., Ramampiaro, H., and Langseth, H. (2018). Effective hate-speech detection in twitter data using recurrent neural networks, applied intelligence.

Srivastava, V., S. M. (2021). Hinge: A dataset for generation and evaluation of codemixed hinglish text. Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems, pages 200—-208.