NLP Project Proposal

Sophia Johanna Schlosser
s.schlosser@mia.hertie-school.org

Samuel Ribanský
s.ribansky@mpp.hertie-school.org

April 6, 2020

1 Introduction

The purpose of this project is to build on previous papers that use Byte Pair Encoding (BPE) to improve the performance of hate speech recognition NLP models. More specifically, we intend to apply Bodapati et al.'s [1] approach to using BPE models in order to improve hate speech recognition on Wikipedia and Twitter to Reddit threads [2]. The documentation associated with this research project, including the Python notebook, the data, and appendices can be retrieved from our publicly available GitHub repository[1]

With over $1.3 billion$ visitors a month, reddit is the 19th most visited website in the world[2]. The website is considered to be the largest discussion forum in the world. Despite having a clearly defined *Content Policy*[3], one of its founders described Reddit's philosophy as *"each individual is responsible for his or her moral actions"*[4], alluding to the rather decentralised hands-off system of moderation employed by Reddit. Hence, it came as no surprise that subreddits (theme-based discussion threads) containing gravely inappropriate content such as child pornography, abusive threads towards racial, religious, and ethnic minorities proliferated. Consequently, Reddit was soon plagued by scandals, including content of threads such as r/incels, r/altright, r/Braincels and similar.

Yet despite the aforementioned occurrences, available information suggests that Reddit content moderation still largely relies on human agents rather than state-of-the-art NLP approaches[3]. According to Singh, human moderators are assisted by a Reddit-specific tool called Automods (for Automatic Moderators), which rely on specific input and varies from sub-reddit to sub-reddit[2]. As such, Reddit moderation tools are unable to respond to innovative methods of communicating hate speech, such as through the use of obfuscated spelling, where letters are replaced by numeric characters.

Therefore, we set out to test and fine-tune available tools, such as BPE algorithms, that enable sub-word character decomposition capable of addressing obfuscated spelling. Consequently, fine-tuned models should be able to identify emerging sub-reddits with illicit content and bring them to the attention of human moderators before they proliferate and are unable to be contained. In doing so, we rely on two papers: *Bodapati et al.: Neural Word Decomposition Models for Abusive Language Detection* [1] and *Sennrich et al.: Neural machine translation of rare words with sub-word units* [4]. We elaborate on the relevance specifics of these papers in Section 2.

2 Motivation and Literature review

2.1 Motivation: Liberty vs Security online

Our motivation for undertaking a project where we attempt to identify hate speech on Reddit is two-

[1] https://github.com/s-ribansky/RedditProject_NLPDL

[2] https://www.similarweb.com/website/reddit.com

[3] https://www.reddit.com/wiki/contentpolicy

[4] https://www.ft.com/content/dbc303c8-c345-11e9-a8e9-296ca66511c9

fold. First, BPE is still a relatively novel approach to textual analysis, going beyond standard bag-of-words and word embedding approaches to textual analysis. Analysing texts based not only on word but sub-word structures can significantly improve the learning potential of hate speech identifying models.

Second, as the previous section already suggested, Reddit is arguably the largest discussion forum, yet it continuously relies on sub-par outdated content moderation methods. This has foreseeable consequences in terms of enabling the proliferation of hate-speech and cyber-bullying online. Such content inadvertently places Reddit in the cross-hairs of internet censors, imperiling an otherwise useful discussion platform. Consequently, being able to increase the speed and accuracy with which hate-speech and sub-reddits containing hate-speech can be identified can protect the platform and its purpose as a harbour for free speech, while defending it from ill-intentioned users.

2.2 Neural Decomposition and Byte-Pair Encoding

Before BPE application, NLP models relied on dictionary approaches to analyse textual data. BPE refers to a simple data compression technique that *iteratively replaces the most frequent pair of bytes in a sequence with a single, unused byte*[4] the new byte comprising of the frequent pairs. In the field of Neural Machine Translation (NMT), this enables a given translator to deal with languages with many agglutinate and compound elements (such as German or Turkish). Consequently, where a dictionary-based translator would have issues translating rare vocabulary and would yield errors, BPE enables a model to decompose an agglutinate word into smaller components. According to Sennrich et al., this enables 1-1 translations not possible with dictionary-based models. Most importantly for our paper, BPE allows for the identification of *morphologically complex words, including words containing multiple morphemes, for instance formed via compounding, affixation, or inflection, may be translatable by translating the morphemes separately.*[4] Perhaps the most important finding of Sennrich at al, is that BPE methods outperform sophisticated dictionary approaches with word-bases upwards of 50,000 words and perform neural decomposition into sub-words much more effectively. In sum, BPE enables decomposition of words into their "byte-sized" components.

2.3 BERT: BPE and hate-speech recognition

Based on the research done by Sennrich et al., Bodapati et al. hypothesise that using neural decomposition approach using BPE can lead to models that will easily outperform word-based hate speech identifying methods. Bodapati et al. posit that word-based models fall short of picking up hate-speech, where users use algorithm defeating methods such as obfuscated spelling. The reason is that even advanced NLP models are trained using standard text (pre-edited corpora) and hence perform poorly on noisy user-generated text (i.e. spelling "women" as "w0m3n").

However, they argue that BPE algorithms can decompose misspelled words and instead of identifying them as out-of-vocabulary (OOV), they can identify them to the key terms associated with hateful and abusive language.

Bodapati et al. test their hypothesis using three datasets, two from Wikipedia and one from Twitter. They then resort to multiple text- and word-decomposition models. Baseline is set using the fastText algorithm, an algorithm that "*performs mean pooling on top of the word embeddings to obtain a document representation.* Sub-word, Joint Word and Character Embedding, and End-to-End Character Embedding models are subsequently used as the state-of-art models for neural decomposition of noisy user-generated text. Lastly the authors train, fine-tune, and apply the *Bidirectional Encoder Representations from Transformers* (BERT) model to the aforementioned Twitter and Wikipedia datasets. Next, using Weighted F-1 indicators, Bodapati et al. find that BERT outperforms both the word-based and sub-word neural decomposition models in being able to identify obfuscated and deliberately misspelled words conveying abusive language. Perhaps the most important finding is that there is an insignificant performance loss with respect to cross-domain application of the BERT model. That is to say that whether applied to Wikipedia or Twitter text, finetuned BERT performs extremely well. Consequently, we are curious to find out whether equally impressive performance of the model will be observed when we apply the model as an abusive language classifier on Reddit data.

3 Proposed method

As mentioned in the previous section, we turn to BPE NLP models to see whether we can train a model to identify reddit comments containing hate-speech elements. More specifically, we turn to the BERT model (especially its computationally lighter smaller brother DistilBERT) to test the effectiveness of BPE models in hate-speech recognition. To this end, we rely on hugging-face transformers[5] that are easy to apply for various types of classifications and do not require a construction of a model from scratch. Moreover, hugging-face transformers allow the model user to 'open the hood' of a given model and adjust the hyperparameters to improve/degrade the chosen model's performance. Consequently, hugging-face transformers enable users with limited background in programming and data science to utilise state-of-the-art deep learning technology for NLP tasks.

While the initial intention of this project was to use the state-of-the-art model, the BERT model, computational limitations forced the authors to resort to less computationally intensive methods. More specifically, we turned to DistilBERT. DistilBERT is a 'distilled' version of the BERT model with a reduced number of embedding layers (capped at 512) and reduced vocabulary. However, according to available information, the model performs to about 97% of the standard BERT model, while reducing computational time by up to 60%.

Architecture

With respect to our architecture, we predominantly rely on hugging-face transformers and minimal architecture presented in this GitHub repository.[6] The reason for not opting for a more complex architecture is that we are looking to test model version easily accessible to uniformed lay audiences, as it would be unreasonable to assume advanced knowledge of PyTorch architecture from Reddit subreddit moderators. Moreover, BERT-family models and their respective pre-trained embeddings help save time and computational costs.

In more detail, our point of departure is the minimal start for binary classification architecture, where we specify the model (DistilBERT) and embeddings (DistilBERT base uncased) and allow the model to assume default settings. DistilBERT's default settings are that maximum sequence layers (alluding to the size of the embeddings) is limited to 128 (retaining less information than the full 512 version), 1 learning epoch, low learning rate (4e-5), and relatively low batch size of 8. It is unsurprising that under such conditions, the model's performance is not up to the standards described above.

Baseline

Using a skeleton version of the model provides us with a baseline against which we can test how the model performs when we adjust different metrics within the model, as well as how it performs against other models using different embeddings. The skeleton version of the model can be found in the aforementioned GitHub repository.

The baseline experiment provided us with a **59.7%** accuracy on predictions. While a sub-optimal result, it will serve as our reference value to compare other model runs with adjusted model parameters to.

4 Experiments

All the experiments ran for the purpose of the midterm report were ran on a database of approximately 23k observations. Every observation is equal

[5] `https://huggingface.co/transformers/model_doc/distilbert.html`

[6] `https://github.com/ThilinaRajapakse/simpletransformers#multi-modal-classification`

to a unique reddit comment and its label. More details on the data and the experimental procedure are provided below.

4.1 Data

We use an extensive database of reddit comments from May 2015, released by Reddit (in excess of 1.7bn comments) and classified according to 22 labels. Unfortunately, none of the labels specifically judge the comments on 'hate-speech'. Therefore, we decided to resort to sorting the comments according to controversiality as a proxy for hate-speech. While this is by no means a perfect solution, a short review of the database showed that a large portion of the randomly selected comments would at least qualify as offensive. Consequently, we judge controversial comments as 'soft hate-speech comments' and assume that there are patterns setting the aside from the non-controversial ones.

A major benefit of this database is its size. With tens of millions of comments, it is representative of the wide variety of user-generated text and can therefore provide the model with a more complex view of what a 'controversial' comment encompasses.

With respect to the database's weaknesses, we identify two. First, we do not know how many annotators annotated the database and therefore cannot know the rate of cross-annotator agreement against which we could compare our model's accuracy. Second, the comments are from one month only; consequently, we assume that it will be replete with topical references not common during different time points. While the former weakness might be problematic in terms of the labels being a result of individual annotators' biases (as we simply have to assume that no cross-annotation took place), the latter should not present an issue for the task we are using the database for.

Lastly, a commentary on the distribution of classes (labels) is in order. When randomly selecting observations for the purpose of initial model training, we noticed that controversial comments usually make up only 3-4% of the total dataset. To avoid overfitting on non-controversial comments and underfitting on the controversial ones. Consequently, we equalised the number of classes, ultimately opting for a 3:2 ratio between controversial and non-controversial comments. By doing so, we were able to judge whether the model was actually capable of distinguishing between the two different types of reddit comments.

Table 1	
Model run	**sklearn accuracy**
Model run 1 [baseline]	59.7%
Model run 2	66%
Model run 3	61%
Model run 4	67.8%
Model run 5	68.1%

Table 1: Results of the initial model runs.

4.2 Evaluation

We primarily rely on two evaluation metrics. The F-1 accuracy and the sklearn package accuracy metric. In the model runs, we primarily focus on the sklearn accuracy metric to give us a rough indication of how the model's performance changes when we tweak the aforementioned hyperparameters.

For the final report, we will equally heavily rely on the Weighted F1 score, measuring the success of the classifier in identifying abusive language. Consequently, we will try and replicate the same model runs as Bodapati et al., but using different texts. As such, we will consider our project successful, if we can achieve high Weighted F1 scores on our fine-tuned DistilBERT model runs.

4.3 Experimental details

Overall, executed five model runs adjusting the maximum sequence length, learning rate, batch size, and the number of training epochs, attempting to observe differences in performance.

Model run 1 [baseline]

During the baseline run, we achieved below 60% accuracy. This training run was limited to a single epoch. All the subsequent runs were stipulated to 10 epochs. None of the model runs was fitted with a loss evaluation auto-stop argument. The run time was very quick, a model run on a GPU-supported backend lasting below 10 minutes.

Model run 2

During the second model run, we achieved a 66% accuracy. Here, the model ran for 10 epochs under the default assumptions of 4e-5 learning rate, maximum sequence length of 128, and a batch size of 8. The performance of distilBERT model was far below expected performance. According to huggingface engineers (link to article), DB's performance should have been approximately 97% or BERT, with BERT's accuracy being close to 97% in predicting. Consequently, we consider 66% to be clearly subpar.

Model run 3

In this model run, we changed the learning rate from the default 4e-5 learning rate to 0.001 as the recommended learning rate for Adam optimisers.[5] Everything else was held constant. Overall, the model performed better on identifying non-controversial than controversial reddit comments. However, accuracy dropped significantly down to **61%**. We attribute this to the noise created by higher learning rate over a small number of epochs, where stopping the model training is not conditioned by epoch-specific loss evaluation.

Model run 4

In this model run, we achieved a 67.8% accuracy. Following the dismal results of the previous model run, we model, increased maximum sequence size from 128 to 256 to allow for a more complex vector representation of the word embeddings. Running the model with these specifications significantly improved our model's performance. We also saw a significant improvement in the running loss values, with loss evaluation metric dropping significantly after every iteration. Nevertheless, we still remain far from the promised accuracy of the model.

Model run 5

In this model run, we adjusted the maximum sequence size even further to maximum sequence size of 512. We did not observe a sufficient increase in performance, with accuracy with sklearn accuracy metric remaining at 68%. When advancing with our project, we will consider both increasing the sample size and introducing a loss evaluation-based condition for stopping the training procedure, while increasing the ceiling for maximum epoch runs.

5 Future work

There are two main avenues we propose to go to improve our project.

First, now that we have established a working model and have a very clear overview of the runtime/accuracy trade-offs, we will further increase the size of the dataset that we will train the DistilBERT model on.

Second, we will introduce a wider array of metrics to get a more accurate overview of how our model performs under different hyperparameters. A key aspect of increasing the usefulness of our model will be the introduction of an auto-stop criterion and a significant increase in the number of epochs for which the model will be run. This step will, of course, take into account the computational power limitations bestowed upon us.

References

[1] Sravan Babu Bodapati, Spandana Gella, Kasturi Bhattacharjee, and Yaser Al-Onaizan. Neural word decomposition models for abusive language detection. *arXiv preprint arXiv:1910.01043*, 2019.

[2] Spandana Singh. Everything in moderation: An analysis of how internet platforms are using artificial intelligence to moderate user-generated content. *New America*,

https://www.newamerica.org/oti/reports/everything-moderation-analysis-how-internet-platforms-are-using-artificial-intelligence-moderate-user-generated
case-study-reddit/(N/A):1–42, 2019.

[3] Shagun Jhaver, Iris Birman, Eric Gilbert, and Amy Bruckman. Human-machine collaboration for content regulation: The case of reddit automoderator. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 26(5):1–35, 2019.

[4] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[5] Delip Rao and Brian McMahan. *Natural language processing with PyTorch: build intelligent language applications using deep learning.* ” O’Reilly Media, Inc.”, 2019.