Apply Deep Learning to Identify Toxicity in Online Interactions

Narayan Acharya
112734365
Department of Computer Science
Stony Brook University
nacharya@cs.stonybrook.edu

Amanpreet Singh
112044129
Department of Computer Science
Stony Brook University
amanpsingh@cs.stonybrook.edu

Abstract

The Oxford word of the year in 2018 was *toxic* (Oxf). Although, one can argue that the reason for this was primarily in the context of the environment, it is important to note that one of the top 10 collocates was 'culture'. With an exponential rise in the number of communication channels available to one through the medium of internet, culture is steadily being derived by interactions of the masses on social media platforms and collaborations in online working environments. As part of this project we wish to understand various ways of identifying toxic behavior in discussions on platforms like Wikipedia, and by extension Twitter. We look to leverage recent advances in the field of Natural Language Processing (NLP) using Neural Networks to come up with fast and efficient solutions. Our aim is to suggest improvements over existing methods as well as apply state-of-the-art techniques previously not explored for the problem at hand.

1 Introduction

1.1 Objectives and Importance

Cases of online harassment and cyber-bullying have been on the rise (Lenhart et al.) (Duggan). Such toxic behavior takes various forms like name-calling, insults, hate speech, obscenity, etc. It affects mental health of the victims and impacts productivity in workplace environments. Curbing such behavior is an operational challenge given the scale at which these platforms operate as well as difficulty in identifying the person on the other side of the keyboard in many cases. From a purely engineering standpoint (or linguistics and NLP), it is difficult because language on these platforms is quite unregulated and unstructured. Behavior construed as toxic on one platform can be considered banter on another. Context and perspective of the one judging too affect whether we classify something as toxic or not.

1.2 Existing Approaches

With the advent of Deep Leaning, sizable data to feed to these networks and significant processing prowess, we have seen positive results in identifying and classifying such toxic behavior using Deep Neural Network models. The most prominent example of classifying toxicity is on the Wikipedia Comments dataset (Wulczyn et al., 2017) using Logistic Regression and Multi-Layer Perceptrons with an n-gram based approach over word or paragraph embeddings. On the back of the Wikipedia Comments dataset, many individuals (Maghales and Small, 2018)(Dhakal and Kedia, 2019) contributed to solving this problem by the means of a Kaggle challenge hosted by Jigsaw (Jigsaw et al., 2017). Submissions focused primarily on number of variants of RNN and CNN based models and were able to churn out decent AUROC (Area Under the Receiver Operating Characteristics) scores for the challenge.

Apart from solving for the problem at hand, there have been considerable attempts focused just on the creation of useful datasets.(Davidson et al., 2017),(Golbeck et al., 2017), (Kolhatkar and Taboada, 2017). One of the more recent attempts was made by (Founta et al., 2018) to create a large scale dataset with over 80000 labelled tweets. This highlights the importance of the task.

1.3 Gaps And How To Address Them

The most critical issue about comments on social media are that they might not reflect correct grammar, have typos and contain common internet slang. Using pre-trained GloVe embeddings (Pennington et al., 2014), used by many-a participants of the Kaggle Challenge, might not be an ideal choice for this case. We look into fastText (Joulin et al., 2017) which is more flexible and

uses n-gram characters to represent each word and also GloVe embeddings trained on Tweets which should help us handle any out of vocabulary (OOV) tokens as well. For instance, A common typo is "Mother" being typed as "Motjer" given the proximity of 'h' and 'j' on the keyboard. While GloVe would give us the embedding for an '{UNK}' token if did not see the word with the typo during training, fastText/Twitter embeddings should return some representation based on the sub-word information. More on this in section 3.5.

Other issues worth considering are the class imbalance of non-toxic comments in a given dataset which skew the prediction models and also the presence of foreign language words which are just communicated using the English alphabet. To address these, we experiment with data augmenting techniques for the minority classes. Also, language identification and eventual translation to English should help us provide the model a more consistent and homogeneous corpus to train on. We also try to use information from emoticons within comments towards our task. We discuss these steps in detail in section 3.4, 3.3, 3.2.

1.4 Initial Analyses and Evaluation

We begin by obtaining the baseline model that performs the best on the given training data with minimal pre-processing. The baselines we considered were multi-layered DAN (Iyyer et al., 2015) and GRU (Cho et al., 2014) architectures and finally a Bi-GRU with Pooling mechanism. We applied the incremental approach of increasing the complexity of the baseline at every step until we had a model that performed the best given the same conditions of operation. Then, we perform our experiments using this model. First, to alleviate the issue of the dominating class of data we applied random sampling and augmented the dataset for the model to train on. Then, its performance is evaluated on four different word embeddings, viz. GloVe trained on Wikipedia, GloVe trained on Twitter data, fastText common crawl and fastText sub-word embeddings so that the OOV tokens and typos would be as few as possible. Finally, before moving on to the advanced models we tried to use a language identification and translation API to recognize non English text and transform it in the dataset.After each iteration, the best performing model is taken for the next set of experiments so as to ensure only one variable in the experimental conditions

For the purpose of this project, we used the Wikipedia comments toxicity datasets provided by Kaggle as the which has around 150k training and test each. The dataset was a partially cleaned version of the original dataset and the comments were labeled from one of 6 *toxicity classes, toxic, severe toxic, obscene, threat, insult* and *identity hate*. Since the task essentially becomes a multi-label classification, each of our baselines are first compared with each other with respect to accuracy,precision, recall and F1 measures. We were also aided by the AUC calculation from Kaggle submissions. As far as testing transfer learning is concerned, we converted the multi-label classification to a binary classification task (Toxic/Non-Toxic) and trained the model on our dataset. The model was then used to predict toxicity on the Twitter dataset (Founta et al., 2018) and results evaluated.
We also look to apply recent Transformer based models to the task which has largely seen RNN/CNN based solutions. This helps us evaluate the current State of Art performance on the task at hand as well,eg. pre-trained BERT models. Transfer Learning by applying our trained model to other datasets for similar identification is something we tried to attempt as part of this project. Details in section 3.6 and 3.7.

1.5 Outcomes

The main outcomes of the project can be listed as follows:

1. We implemented a high performing baseline and a state of the art model for the task of identifying toxicity and compared the results.

2. To ensure the best possible results, we experimented with extensive pre-processing, different word embeddings, data augmenting techniques, language translation correction.

3. Our evaluations showed the BERT models consistently outperforming even stacked sequence based models when the experimental conditions are similar.

4. We identified that BERT was better able to capture contextual information compared to

other models we tried. But no model was able to capture all information exclusively. Comments that are subjective or those including words that are perceived as toxic but the overall comment was not were instances where models struggled to come up with a unanimous toxicity classification.

5. Additionally, we found that applying a model trained on one toxicity dataset showed promising prediction results on another as well if the dataset is made consistent with the input the models expects.

6. Based on our observations, we can conclude that latest misspelling based embeddings and translation based data augmenting techniques showed deliver even better results as these would lead to an even better dataset for the model to train on. Also, applying the BERT models to the transfer learning task, which we weren't able to accomplish unfortunately should produce some exciting results.

2 Identifying Toxicity

As mentioned earlier we are using a collection of Wikipedia comments as our corpus and after preprocessing try to assign them to the six toxicity classes. Since it is a multi-label classification, the output of the model is the probability of the input belonging to each class calculated based on sigmoid activation. Higher the output value, more chances are that the comment was toxic. Some of the inherent challenges of the task are the subjective nature of toxicity and annotater agreement between the crowdsourced workers which is addressed by (Wulczyn et al., 2017). Apart from that are the obstacles in the actual textual data like typos, use of emoticons and internet slang, the overwhelming presence of the non-negative comments in the dataset and sparsed out non-English text presented in the Latin script which becomes difficult to understand for the model. Pre-trained models like BERT (Devlin et al., 2018) and its variations with their contextual embeddings are considered the State-of-the-art for classification tasks.

2.1 Baseline Model(s)

For our baselines we considered three neural network architectures of increasing complexity:

1. Deep Averaging Network (DAN): A basic architecture of 4 Dense layers with initial dropout where the initial input representation of the text is a mean of the word embedding vectors. This is achieved by creating a Lambda layer which calculates the mean of the embeddings and feeds them to the first Dense layer.

2. Multi-layered Gated Recurrent Units (GRU): To better represent the text we then moved on to sequence-based models. We used a four-layered GRU architecture with a Dense layer at the end.

3. Bi-directional GRU with Pooling (Bi-GRU): Finally, we used a single Bi-GRU based model with dropout and concatenation of average and max pools from the intermediate representations. Also, a 1D Spatial dropout layer is used to get the maximum possible chance to the model to avoid over-fitting and identify new examples. The architecture can be seen in 1.

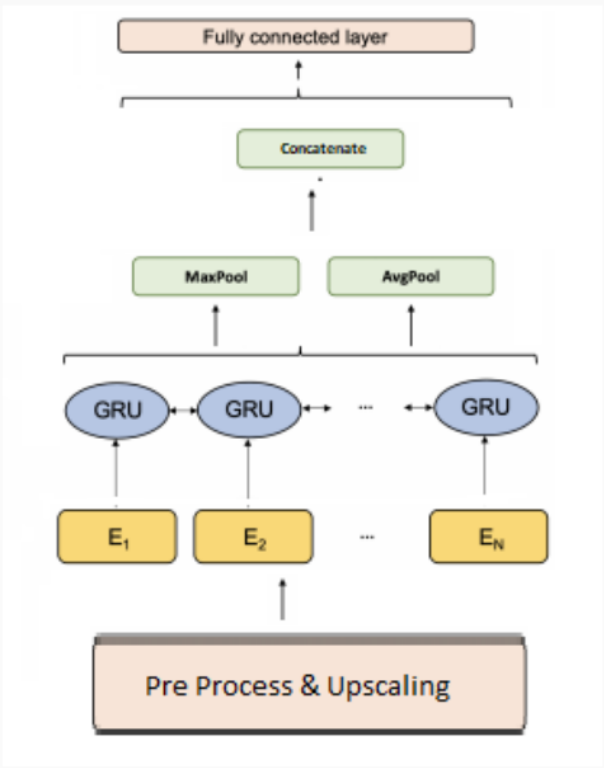

Figure 1: Bi-GRU with Pooling

All of the above were fed by an Embedding layer which converted the tokenized inputs to their word vector representations. The embedding matrix was generated using the publicly available GloVe vectors with 300 dimensions which are

trained on Wikipedia data itself. Additionally, the output of the hidden layers is finally connected to a Dense Classification layer at the end with 6 output units and sigmoid activation to determine the probability that an input text belongs to one or more classes.

The loss function chosen for the task and applied to each model was binary cross-entropy loss which calculates individual 'Yes/No' probability for each of the classes instead of softmax which converts all the logits to a single probability distirbution.

2.2 The Issues

The DAN model just mushes up the token representations using the mean aggregate function, the sequence and contextual information provided by the text is lost. The multi-layered GRU model easily outperforms the DAN model but with four layers it is extremely slow to train. This model actually gave the best results but with almost the same classification performance and half the training time with only a single layer. We therefore chose the Bi-GRU model as the baseline for our further experiments.

The main issues with the initial baseline models were as follows:

1. Class Imbalance: Most of the people who edit Wikipedia entries, we believe would be good Samaritans. The dataset reflects that with only a fraction of the comments being non-toxic. Due to this the model was classifying almost all the comments into the 3 majority classes only. *toxic, obscene, insult* with poor F1 score on the other classes. The number of samples for each class can be seen in figure 2.

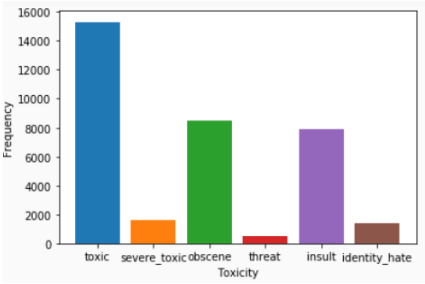

Figure 2: Toxicity Classes Imbalance

2. Typos and Online Communication practices: The example of 'motjer' and 'mother' given in the previous section was just one of the examples where misspellings enter and become a part of the corpus. These then get represented using the *UNK* token even though its obvious that 'mother' would have a specific representation. Even with the GRU based models, the F1 score for classification of 'toxic' was around 80 and we suspected it could be improved if embeddings which are sub-word based to make sense of 'mot' and 'er' in 'motjer' and those trained specifically on such corpus like Twitter are used.

3. Ambiguous Text: There were some examples where the model classified them a toxic just because there was a profanity in it or as non-toxic because it could not understand a word which could be from a non-English language or the toxicity could be veiled in sarcasm.

3 Approach

The manner in which we approached the task was simple. We picked the best performing model in terms of evaluation and training time from our initial baselines as described in the previous section. Thereafter, we perform our experiments initially to overcome the roadblocks in the baseline. Once we have a high performing model, we then train BERT variants on the same dataset and compare the performances. Finally, we apply the high performing model on a large scale Twitter dataset to validate our claim of transfer learning.

3.1 Dataset Details and Pre-Processing

We used the Wikipedia Comments Dataset available through the Kaggle Competition hosted by Jigsaw. The dataset contains almost 160k comments for training. There are around 150k comments for testing our model. Each training sample is annotated with 6 levels/types of toxicity - toxic, severe toxic, obscene, threat, insult and identity hate. The annotation of the comments was done manually using crowd-workers. More details of the dataset can be found (Wulczyn et al., 2017) and (Jigsaw et al., 2017).

The comments in the dataset we started off with with were highly unstructured as one would expect from comments in online forums. The critical task was to correct this using appropriate cleaning techniques. In this step we tried to employ

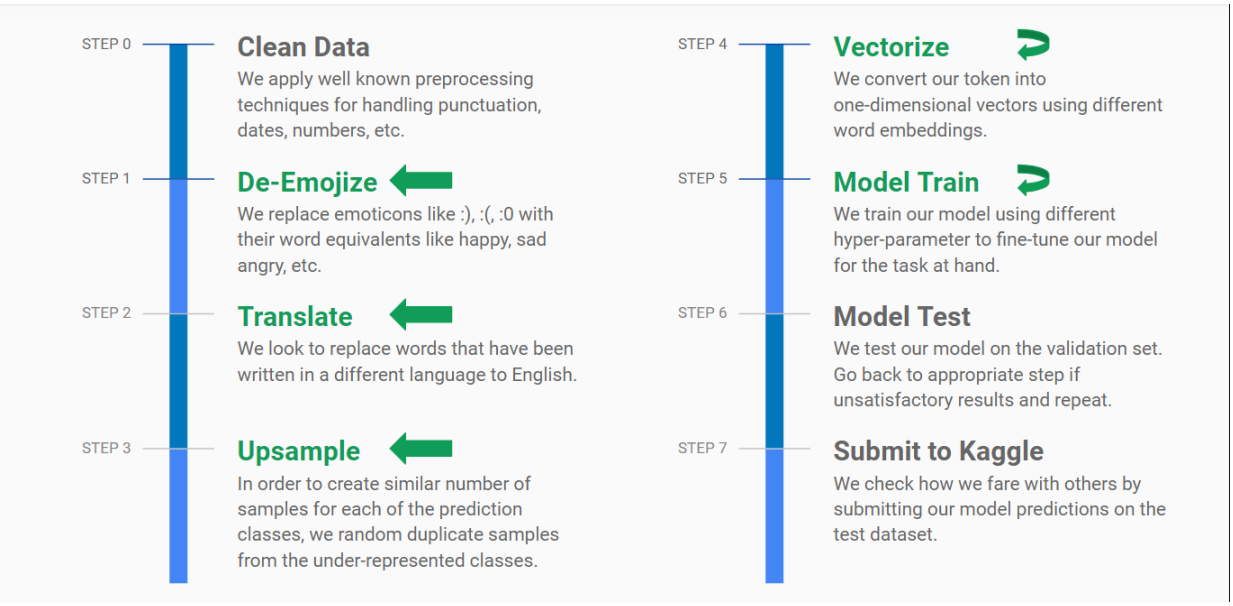

Figure 3: Process Flow

well-known measures to clean our data like adjusting for punctuation like double-quotes, cleaning up date and timestamps, removing IP addresses and HTTP URLs from the text and replacing numbers with the pound sign # (”3” replaced with ”#”), characters like ”&” and ”@”, etc. Apart from these we also introduce three novel techniques listed below, we believed would help with resolving the issues stated previously.

3.2 De-Emojize

An avenue we explored was to make use of emojis/emoticons in text to our cause. We believe that emojis are a common way of expressing one’s interest towards the conversation and is a meaningful source of information for our task. During our Exploratory Data Analysis (EDA) we noticed that quite a few comments had emoticons in them. These were text-based emoticons like :), :(:O, etc. and *not* Unicode-based emojis as we would like to assume that most of the people who work on Wikipedia entries would work from their Personal Computers and emojis are a medium more popular in smart phone communication. We looked to employ a simple search and replace of these emoticons with a word corresponding to its meanings. For eg. *”Thank you for that link :)”* was transformed to *”Thank you for that link happy”*. We chose a rule base approach here by keeping a map of emoticons with their respective meanings as a dictionary while doing these steps.

3.3 Translate

There were a few examples, where users chose to use words from a different language in their conversation. In order to work around this, we introduced another pre-processing step - to translate words that are not of the English language into English. Consider the following example: *”Sokken voor wasmiddel.”* was translated to *”Socks for detergent.”*. We used polyglot, a python library, for this task but had unsatisfactory results. We also tried using an unofficial wrapper around Google Translate, translate, but soon ran into API hit restrictions as we went through our train-test-tweak cycle. Our final models, unfortunately, do *not* use this step during the pre-processing phase.

3.4 Up-sample

The toxic classes we were classifying were close to 10% of the entire dataset, see 4. This would lead to issues generally associated with classification with an unbalanced number of classes. To bring balance we up-sample comments for the classes where number of examples are very low *severe_toxic, threat, identity_hate*. We randomly select 24000 samples from theses classes and duplicate them in our dataset. We avoid bringing all classes to equal number of samples because that could possibly lead to over-fitting. This increased our dataset from 150k to 180k comments.

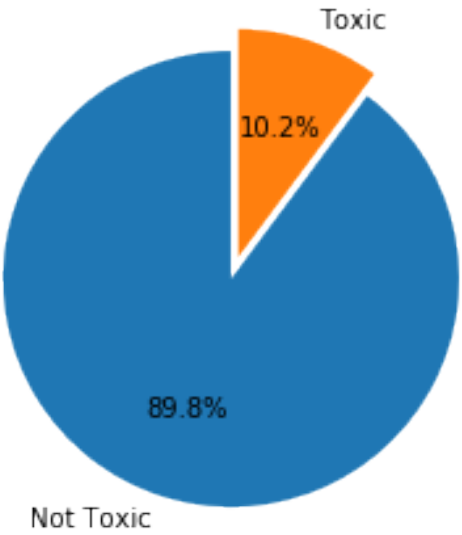

Figure 4: Percentage of Toxic Comments

3.5 Choice of Embeddings

The other steps in our flow our well understood pieces and we avoid explaining them in details. Once we have our clean data, we look to vectorize them using various embeddings. Our choice of embeddings to use was based on multiple options given we were looking to get around the issue of misspellings and OOV tokens. We experiment with GloVE, and fastText pretrained on different corpora like Wikipedia,Twitter and Common Crawl. 3. We even experimented with the embedding dimensions starting from 50,100 and finally 300. No surprises there, the model operating with embeddings of dimensions 300 edged out the others and was used as the common parameter for training the initial baselines.

3.6 BERT Implementation

We mentioned earlier that we would be comparing our best baseline with the BERT variants which are considered State-of-the-Art today. Here, we use Simple Transformer (Sim) based wrappers which are written on top of the Hugging Face Pytorch distributions of the BERT models. We used the Vanilla and distilled versions for our investigation. Since BERT has its own contexualized embeddings, we pass on the augmented and preprocessed data set as input to the model for training. Just to be sure about the impact of upsampling here as well, we trained two similar BERT models and the one trained on upsampled dataset showed a marked improvement over the other. Amongst the various available variants, we used the 12-layered 'bert-base-uncased' and 6-layered 'distilbert-base-uncased'. The high-level architecture can be seen in 5.

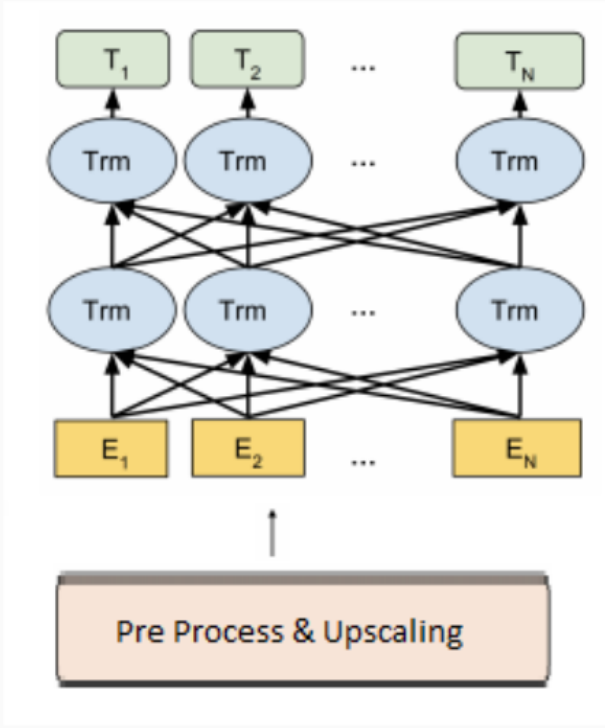

Figure 5: BERT Architecture

3.7 Transfer Learning

With a high performing model in our hands, we wanted to evaluate how well it would do if applied to another dataset created for identifying toxicty online. We use the labelled 80000 Tweets dataset create by [Founta et al.] for this purpose. Now, the labelling for both the datasets is different. The Wikipedia dataset has 6 classes while the Twitter dataset has only 4. To get around this issue we converted the Multi-label classification task to a Binary Classification one, just predict whether the given input is toxic or not. Everything in our high performing model architecture was kept the same except for the output units at the end are changed from 6 to 1 for a simple Yes/No probability for toxicity. Then the model was trained again and evaluated on its performance on the Twitter dataset.

3.8 Implementation Details

Having outlined our baselines and experiment details we, train our models with the goal of reduc-

No.	Parameter	Values Considered	Final Value
1	Embedding Size	50/200/300 (200 for Twitter)	(i)300 for Baselines (ii)200 for Experiments as the training time required was lesser with comparable performance to GloVe vectors of dimension 300.
2	Epochs	4/3 - Baselines 3/2 - BERT	(i)3 for Baselines incl. Bi-GRU, seemed to overfit even with dropout for epochs >3 (ii)2 Primarily due to the training time required for each epoch is around 30 mins. distillBERT does a little better but still takes around 27 minutes per epoch.
3	Dropout	0.1/0.2/0.5 - Both element wise and 1D Spatial	(i) 0.2 - DAN as mentioned in the paper (ii)0.2 - 0.1 seemed to have negligible effect and 0.5 leads to reduced performance on both GRU and Bi-GRU with and without recurrent dropout
4	Vocab Size	20k/30k	30k - Seemed to give max performance at the same time not including a lot of rare words
5	Max Sequence Length	100/150/200	100-The mean sequence length was around 67, so 100 captures all the information without much padding for those while not losing a lot of information for the others, which on analyses would be similar to what was already captured after pre-processing for most cases.

Table 1: Hyper-parameter choices for the models

ing the validation loss and then run our model on the hidden test dataset provided by the aforementioned Kaggle challenge. We submit our predictions and see how our models fare and re-iterate from there. The process flow can be seen in 3. For model training we split the data 90:10 into dev and validation sets as we also had a huge test set separately provided by Kaggle for evaluating new data. Both the dev and validation sets were tokenized using the Keras API provided by TensorFlow. Also, as mentioned earlier we considered the maximum sequences length as 100 and the sequences were either padded as prefixes or truncated based on their length. This was done for each of our baseline experiments and the BERT models.

4 Evaluation

As mentioned in the previous section, we have separate training and test datasets of considerable volume to train and evaluate our models. We need to evaluate the models with appropriate measures. Considering only accuracy for our task would be misleading as 80% of the comments in the dataset are not toxic. But we still need to determing how well the model is doing in placing the comments in their appropriate buckets.

4.1 Model Parameters

The hyper various parameters used for the models are outlined in 1

4.2 Evaluation Measures

Classification Report (bert):

	precision	recall	f1-score
toxic	0.91	0.94	0.92
severe_toxic	0.83	0.95	0.88
obscene	0.92	0.97	0.94
threat	0.94	0.99	0.97
insult	0.89	0.95	0.92
identity_hate	0.91	0.98	0.94
micro avg	0.90	0.95	0.92
macro avg	0.90	0.96	0.93

Figure 6: BERT Classification Results

Classification Report (Bi-GRU):

	precision	recall	f1-score
toxic	0.94	0.90	0.92
severe_toxic	0.79	0.85	0.82
obscene	0.94	0.92	0.93
threat	0.86	0.77	0.82
insult	0.89	0.90	0.90
identity_hate	0.90	0.89	0.90
micro avg	0.90	0.90	0.90
macro avg	0.89	0.87	0.88

Figure 7: Bi-GRU Classification Results

We have made use of the following two methods for evaluating our models:

1. Minimize the validation loss on each epoch of training and finally compare the precision, recall and F1 scores across all the toxicity classes.

2. Once satisfied with the validation set evaluation metrics score, we further use the mean AUROC over each of the 6 labels that we need to classify for as a metric for evaluating our models. We look to submit our predictions on the test data to see where we fare on the Kaggle Competition Leaderboard for this challenge and evaluate how close or how far we were from the 'best' or 'state-of-the-art' for this challenge.

   Other than the numeric metrics, we have done extensive analysis of the various samples every model classifies as toxic. non-toxic and we present those findings in the next section.

4.3 Results

The results obtained for each of the experiments outlined in the Approaches section follow.

(i) Baselines: Bi-GRU with average and max pooling results in the best possible with least number of layers and training time.

Model	Valid. Loss	Valid. Acc.	Kaggle AUC
DAN-4 layers	0.0625	0.9781	0.9481
GRU-4Layers	0.0436	0.9836	0.97938
Bi-Gru w Pooling	**0.0449**	**0.9834**	**0.97563**

Table 2: Baseline Experiment Result

(ii) Upsampling: Upsampling with random samples leads to a marked increase in the evaluation scores.

Dataset	Valid. Loss	Valid. Acc.	Kaggle AUC
150k	0.0449	0.9834	0.97563
180k	**0.0592**	**0.9775**	**0.98192**

Table 3: Up-sampling on Bi-GRU Model

(iii) Embeddings: The GloVe embeddings trained on Twitter data take the leaset time to train given the dimensions are 200 but provide almost the same performance as GloVe.6B.300.

Embeddings	Valid. Loss	Valid. Acc.	Kaggle AUC
glove.6B	0.0592	0.9775	0.98192
fastText-crawl	0.0601	0.977	0.98117
fastText-subword	0.0616	0.9765	0.97644
glove.twitter.27B	**0.065**	**0.9746**	**0.9811**

Table 4: Embeddings experiment with up-sampled dataset

(iv) BERT Models: BERT models were trained on both the original and our up-sampled version of the dataset.

Model	Valid Loss	Valid Acc	Kaggle AUC
BERT upsampled	**0.04997**	**0.99447**	**0.98382**
DistillBERT upsampled	0.0618	0.99258	0.98354
DistillBERT	0.04244	0.99554	0.97571

Table 5: BERT Experiments

(v) Transfer Learning on Twitter dataset: As we can see below applying the model trained on Wikipedia Comments dataset to the Twitter dataset yield almost similar results. In fact, for the toxic the models are more or less equivalent. Hence, we validate the power of transfer learning from this endeavor.

	precision	recall	f1-score
Non-Toxic	0.94	0.95	0.95
Toxic	0.92	0.89	0.91
accuracy			0.93
macro avg	0.93	0.92	0.93
weighted avg	0.93	0.93	0.93

Figure 8: Classification results on Twitter Dataset

	precision	recall	f1-score
Non-Toxic	0.97	0.98	0.98
Toxic	0.93	0.90	0.92
accuracy			0.97
macro avg	0.95	0.94	0.95
weighted avg	0.97	0.97	0.97

Figure 9: Classification results on original Wikipedia Dataset

4.4 Analysis

1. Up-sampling: Up-sampling was a critical piece in our pre-processing steps. Taking a look at 6 one can easily identify that our best

Dataset	Model	Batch Size	Embeddings	Kaggle Score (AUC)
150k	**Bi-GRU w/ Pooling (BASELINE)**	1000	glove.6B.300d.txt	0.97563
180k	Bi-GRU w/ Pooling and/ **Up-sampling**	1000	glove.6B.300d.txt	0.98192
180k	Bi-GRU w/ Pooling and Up-sampling	1000	**glove.twitter.27B.200d**	0.9811
180k	**BERT** w/ Up-sampling **(BEST)**	2	bert-base-uncased	**0.98382**

Table 6: Summary Evaluation Results

models were with up-sampled dataset which was to work around the unbalanced class size issue while classification.

2. Translation: Translation to English during pre-processing did *not* help us much as number of examples where this mattered was way to small compared to the size of our dataset. Another drawback of adding translation was the time it added to our pipeline was significantly much more compared to the benefit it brought to the table.

3. False Positives: Following examples highlight some comments that were classified as toxic even though there was nothing toxic in them if one knew context or the background.
   (a) "(1956 film)—Moby Dick" - This is a movie.
   (b) "Axel from Streets of Rage Blaze from Streets of Rage Skate (unlockable) from Streets of Rage 2" - Axel, Blaze and Skate are characters from the series Streets of Rage.
   (c) "* Crap. Sorry. I'm fixing it now." - This looks like an apology. "Crap" in this context is not toxic.

4. Grey-Area scenarios: There were quite a few samples where different models identified them as toxic or not toxic differently. Table 7 lists some of these examples. If you see, some of the sentences were categorized toxic, maybe, because of the presence of some words, "kickass" and "badass", in examples 1 and 2. Example 3 illustrates a situation where the text contains the name of a book but is possibly categorized toxic due to the negative connotation involved with one of the words in the book's title. Example 4 again, is again seems to be a question in a healthy conversation, but the mention of some of the words largely found in toxic comments puts our models off. Examples 5 and 6 have nothing one can put a finger on for classification as toxic or not. Maybe, these require an expanded context, say the comments before and after and making a decision solely based on this comment might not be correct.

Finally, we compare predicted labels per class for Distill BERT vs Bi-GRU which can be seen in figure 10. Note that the final metrics are actually based on prediction confidence as a floating point number between 0 and 1. The comparisons in the figure are based on rounding off to the nearest integer.

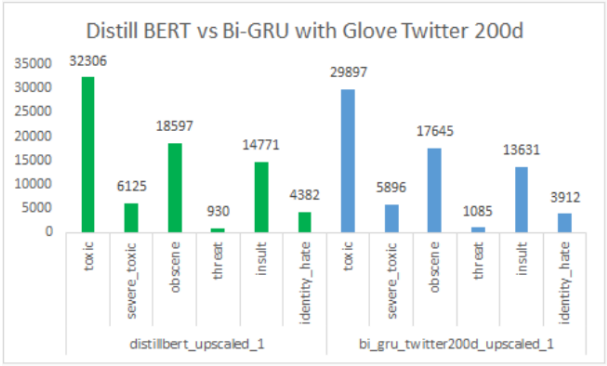

Figure 10: BERT vs Bi-GRU Prediction Results

4.5 Code

Any one of the below should work:
i) Code@Google Drive:Access Code
ii) shorturl.at/ehoqS

Sample	Distill BERT	Bi-GRU [GloVe Twitter 200d]
”make those kickass userboxes”	TOXIC	NOT TOXIC
”this page should be altered to relflect the fact that John Luther Adams is a badass.”	TOXIC	NOT TOXIC
”Commandant of Auschwitz”	NOT TOXIC	TOXIC
”How did he die? I just ask because his death was during black death years.”	TOXIC	NOT TOXIC
”Babies can’t be wealthy.”	NOT TOXIC	TOXIC
”Oh, and stop calling everybody racists too.”	NOT TOXIC	TOXIC

Table 7: Grey Area Examples

5 Future Work

i. Use Translation to other languages and back to English as a way of augmenting dataset rather than randomly up-sampling. This would create new instead of duplicate training samples.

ii. Use Misspelling Oblivious Word Embeddings or MOE by Facebook Research as an alternate over GloVe and fastText.

iii. Apply BERT models to get an even better performance for transfer learning.

iv. Try using Lamb optimizer to quicken up BERT model training.

6 Conclusions

We created a high performing Bi-GRU model for the task of classifying Wikipedia comments into six sub-classes of toxicity. Thereafter, we compared its performance with the State-of-the-Art BERT models and also applied the model to the task of Transfer Learning on a large scale Twitter dataset. Some of the key takeaways are:

i. The effect of dominating classes can be negated to a great extent if resampling done carefully. (Point i. in future work)

ii. There is a tradeoff between the training time and every point of accuracy gained from DANs to BERT.

iii. The power of transfer learning with pre-trained models like variants of BERT.

iv. Pitfalls of a subjective task such as this - comments with profanities are more likely to get marked as toxic.

v. Multi-label classification using a deep learning architecture is implemented as repeated binary classification.

7 References

References

Simple Transformers BERT... https://github.com/ThilinaRajapakse/simpletransformers#minimal-start-for-multilabel-classification. [Online; Accessed 10-Nov-2019].

Word of the Year 2018 is... https://languages.oup.com/word-of-the-year/word-of-the-year-2018. [Online; Accessed 10-Nov-2019].

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM ’17, pages 512–515.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Aaakarshan Dhakal and Dhruv Kedia. 2019. Cs224n final paper. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1184/reports/6838795.pdf. [Online; Accessed 10-Nov-2019].

Maevve Duggan. Online Harassment 2017. https://www.pewresearch.org/internet/2017/07/11/online-harassment-2017/. [Online; Accessed 10-Nov-2019].

Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *11th International Conference on Web and Social Media, ICWSM 2018*. AAAI Press.

Jennifer Golbeck, Zahra Ashktorab, Rashad O. Banjo, Alexandra Berlinger, Siddharth Bhagwan, Cody Buntain, Paul Cheakalos, Alicia A. Geller, Quint Gergory, Rajesh Kumar Gnanasekaran, Raja Rajan Gunasekaran, Kelly M. Hoffman, Jenny Hottle, Vichita Jienjitlert, Shivika Khare, Ryan Lau, Marianna J. Martindale, Shalmali Naik, Heather L. Nixon, Piyush Ramachandran, Kristine M. Rogers, Lisa Rogers, Meghna Sardana Sarin, Gaurav Shahane, Jayanee Thanki, Priyanka Vengataraman, Zijian Wan, and Derek Michael Wu. 2017. A large labeled corpus for online harassment research. In *Proceedings of the 2017 ACM on Web Science Conference*, WebSci '17, pages 229–233, New York, NY, USA. ACM.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daume III. 2015. Deep unordered composition rivals syntactic methods for text classification. https://people.cs.umass.edu/~miyyer/pubs/2015_acl_dan.pdf. [Online; Accessed 10-Nov-2019].

Jigsaw, Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Toxic comment classification challenge. https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge. [Online; Accessed 10-Nov-2019].

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics.

Varada Kolhatkar and Maite Taboada. 2017. Constructive language in news comments. In *Proceedings of the First Workshop on Abusive Language Online*, pages 11–17, Vancouver, BC, Canada. Association for Computational Linguistics.

Amanda Lenhart, Michele Ybarra, Kathryn Zickuhr, and Myeshia Price-Feeney. Online Harassment, Digital Abuse, and Cyberstalking in America . https://www.datasociety.net/pubs/oh/Online_Harassment_2016.pdf. [Online; Accessed 10-Nov-2019].

Ashe Maghales and Howard Small. 2018. Deep learning approaches to classifying types of toxicity in wikipedia comments. https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1184/reports/6838795.pdf. [Online; Accessed 10-Nov-2019].

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. [Online; Accessed 10-Nov-2019].