Jigsaw Multilingual Toxic Comments Classification on Kaggle

Abstract

This paper explores different deep learning approaches to classifying toxic comments found online. We implement M-BERT and XLM-Roberta with different architectures and examine the effects of downsampling, translating and fine-tuning on datasets. The models were evaluated based on their AUC score on the validation set. Our best score of 0.9239 on the test set was achieved with an XLM-Roberta model fine-tuned on the validation set with a Binary Cross-Entropy Loss function. Overall, XLM-Roberta performed better than M-BERT for this task. Moving forward, we hope to improve performance by experimenting with other pre-trained embeddings, and further preprocessing our data before training.

Introduction

Toxic comments are defined by the host of the competition as comments that are rude, disrespectful or otherwise likely to make someone leave a discussion. They have always been an issue online and given the prominence of social media nowadays, their impacts are more worrying than ever. The goal of this project will be to use deep learning models to identify cross-lingual toxic texts and develop a system that is more robust than existing toxic-comment detecting models.

This project aims to implement two models - XLM-Roberta and M-BERT to solve the issue at hand. The applications of these models with different architectures, such as concatenated output layers and various strategies like adding translations or using few-shot learning will also be examined.

Related Work

a. *Challenges for Toxic Comment Classification: An In-Depth Error Analysis* [1]

   This paper performed an error analysis on the English toxic comments presented in the Kaggle Toxic Comment Classification Challenge 2018 [2]. The authors concluded that the best individual classifier was a Bidirectional GRU Network with Attention embedded with FastText, and the best overall classifier was an ensemble of gradient boosting trees. This paper also states that combining shallow learning approaches with Neural Networks would be highly effective.

b. *Detecting and Classifying Toxic Comments* [3]

This paper used data from [2] to study the effects of SVM, LSTM, CNN and MLP on classification with word and character-level embeddings. The best model that they proposed was a Unidirectional-LSTM model with word-level embeddings.

c. *Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond* [4]

This paper introduced a multilingual sentence representation trained with Bidirectional-LSTM encoder with shared BPE vocabulary for 93 languages. Their proposed embeddings were able to outperform BERT-uncased embeddings in a Zero-Shot Transfer system.

d. *MultiFiT: Efficient Multi-lingual Language Model Fine-tuning* [5]

This paper proposed an LSTM Language model with tuned dropout hyperparameters (AWD-LSTM) and cross-lingual bootstrapping using its zero-shot predictions as pseudo-labels. The proposed model, Multi-lingual Fine-tuning (MultiFiT) outperforms M-BERT in its zero-shot learning task.

e. *Unsupervised Cross-lingual Representation Learning at Scale* [6]

In this paper, the authors trained a Transformer-based masked language model on more than two terabytes of filtered Common Crawl data that consists of one hundred languages. They conclude that pretraining multilingual language models at scale leads to significant performance gains for a wide range of cross lingual transfer tasks.

Datasets

The datasets that were used for this task came from Kaggle. The training dataset is made up of 223,549 unique English comments extracted from Wikipedia's talk page edits and comes in the format of [id, comment_text, toxic, severe_toxic, obscene, threat, insult, identity_hate].

The validation set contains 8,000 comments from Wikipedia talk pages in Turkish, Spanish and Italian and comes in the format of [id, comment_text, lang, toxic].

Another version of the training dataset preprocessed for BERT models where sequences are limited to 128 tokens was also provided by Kaggle and used during the training of M-BERT.

The datasets were imbalanced with a majority of comments being classified as non-toxic (~90%) and further preprocessing steps were taken which are detailed in the methods section below.

Methods

Our whole task is completed using GPU and TPU resources from Google Colab and Kaggle. To achieve higher efficiency, most of our training is processed by the TPU v3-8 board on Kaggle.

During data pre-processing, in order to address the class imbalance problem, the training data is down-sampled by selecting all toxic comments and randomly selecting a small set of non-toxic comments. Also, we pad all sentences with a max sentence length and pack them into batches.

To coincide with the cross-lingual setting in our task, we import two pre-trained multi-lingual transformer models from the Transformers package: Multi-lingual Bert Cased Base (referred as M-BERT below) and XLM-Roberta Base (referred as XLM-R below). On the top of pre-trained models, the outputs are sent through a dropout layer and passed to a linear layer for classification. In the fine-tuning step, we use the AdamW optimizer with a linear scheduler for warming up. The loss function and training epochs are varied in our experiments (see Experiments for more details). Typically, we use 2-4 epochs.

The performance of classification is evaluated by the Area Under the Receiver Operating Characteristics Curve (AUC score). It varies from 0.5 to 1.0, where 0.5 represents that the model is not capable of distinguishing the two classes, while 1.0 represents that the model perfectly separates the two classes, so we aim to achieve as close to 1.0 as possible.

Below is a summary of hyper-parameters for our used models:

Summary of Model Hyper-parameters

	M-BERT	XLM-R
Model Configuration	● 12 layers ● 768 hidden states ● 12 heads ● 110 Million parameters ● Train source: Wikipedia	● 12 layers ● 768 hidden states ● 8 heads ● 125 Million parameters ● Train source: CommonCrawl
Hyper-parameters	● Batch size = 32 ● Max length = 192 ● Learning rate = 1e-5 ● Dropout = 0.0 ● Warm-up proportion = 0.2	● Batch size = 32 ● Max length = 192 ● Learning rate = 2e-5 ● Dropout = 0.3 ● Warm-up proportion = 0.2

Experiments

a. *Baseline*

We start by comparing the performance of M-BERT and XLM-R models with the basic architecture, and we choose the one with higher performance as our on-going model for more experiments, and the other one as our baseline model.

b. *Loss Function*

Considering our task is binary classification, the loss function used is the Binary Cross-Entropy Loss, which utilizes the log loss function. To be specific, we use the BCEWithLogitsLoss which combines with a sigmoid activation function.

c. *Model Architectures*

i. *Basic Architecture*

Since in the transformer models, the first token `[CLS]` contains all self-attended information, so for the basic architecture, the `[CLS]` representation from the last layer is taken and passed into the dropout and classification layers.

ii. *Alternative Architecture*

As different layers may capture different attention information, we try a different architecture by concatenating all layer's `[CLS]` representations for classification.

d. *Translation*

To approach the zero-shot learning problem presented in the setting of data, we propose the experiment that we first train the model on English source set, and perform the classification on the translated validation and test datasets. Due to limited time constraints, we use the prepared translated datasets from [7].

e. *Fine-tuning on Validation*

Given the multilingual validation data, we can convert the zero-shot learning problem to few-shots transfer learning by fine-tuning the model on the validation data after training on the English dataset. Since the validation set is relatively small, we do not perform sub-sampling and use the complete validation set for fine-tuning. Specifically, we train the model on the English set for 2 epochs, and then train the best model on the validation set for 1 epoch to complete the transfer learning.

Results

In this section, we perform a comprehensive analysis of two multilingual transformer models, M-BERT and XLM-Roberta. Then, we focus on different architectures of the XLM-Roberta models and discuss two strategies we applied in this project in detail. Finally, we would show our best fine-tuned results and our performance on Kaggle competition.

a. M-BERT vs. XLM-Roberta

Table1. M-BERT report

	precision	recall	f1-score	support
0.0 (non-toxic)	0.88	0.95	0.92	6770
1.0 (toxic)	0.55	0.31	0.39	1230
accuracy			0.85	8000
macro avg	0.72	0.63	0.66	8000
weighted avg	0.83	0.85	0.84	8000

Table2. XLM-Roberta report

	precision	recall	f1-score	support
0.0 (non-toxic)	**0.91** ↑	**0.96** ↑	**0.93** ↑	6770
1.0 (toxic)	**0.66** ↑	**0.45** ↑	**0.53** ↑	1230
accuracy			0.88	8000
macro avg	0.78	0.70	0.73	8000
weighted avg	0.87	0.88	0.87	8000

* ↑**or**↓ in Table2 means increment or decrement compared with Table1.

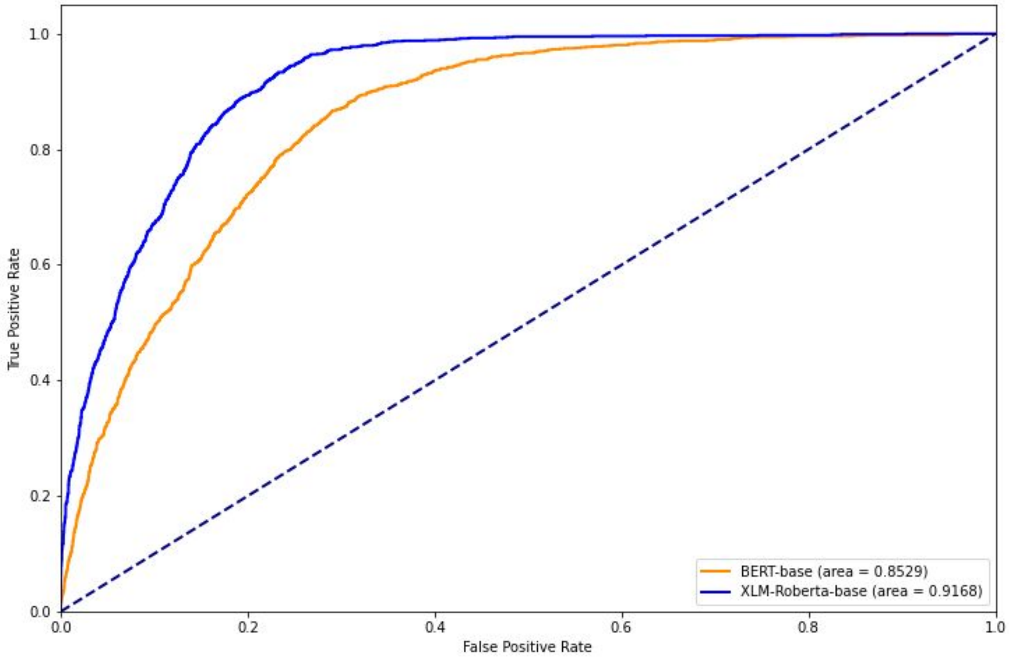

Fig1. AUC curves of M-BERT and XLM-Roberta models

Table1 and Table2 are classification reports for the M-BERT model and XLM-Roberta model, respectively. Regarding those two reports, XLM-Roberta performs much better than BERT in terms of both two categories, toxic and non-toxic. Especially, for toxic comments, XLM-Roberta model gets the f-score of 0.53 which increases by 14%. Fig1 shows that XLM-Roberta overcomes BERT overwhelmingly.

b. Basic Architecture vs. Alternative Architecture

Table3. XLM-Roberta with Concatenated Features

	precision	recall	f1-score	support
0.0 (non-toxic)	**0.92** ↑	0.93 ↓	0.93	6770
1.0 (toxic)	0.59 ↓	**0.57** ↑	**0.58** ↑	1230
accuracy			0.87	8000
macro avg	0.76	0.75	0.75	8000
weighted avg	0.87	0.87	0.87	8000

* ↑**or**↓ in Table3 means increment or decrement compared with Table2.

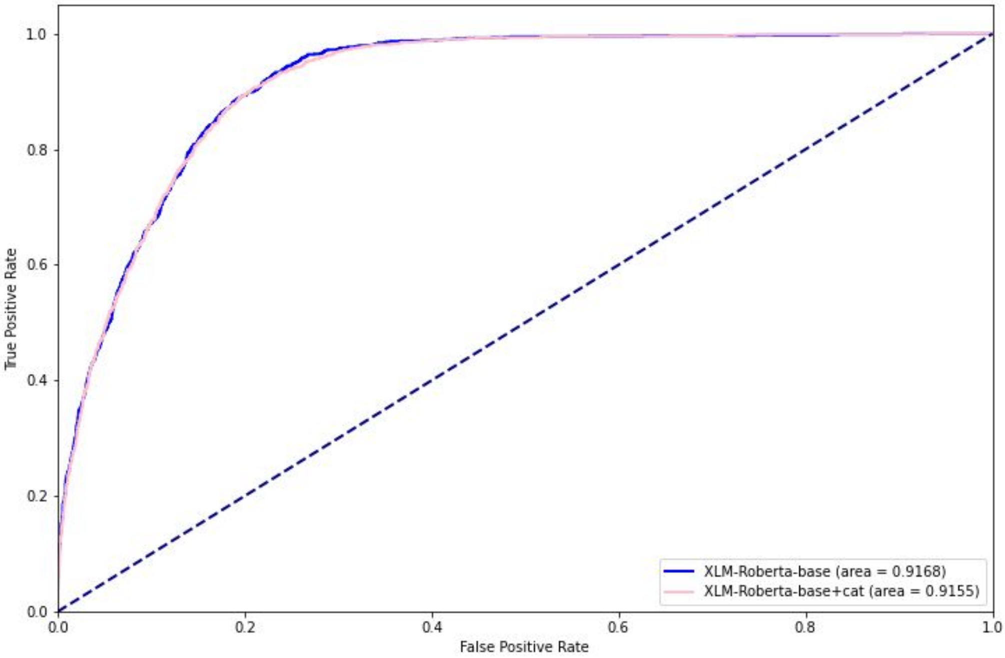

Fig2. AUC curves of basic XLM-Roberta and alternative XLM-Roberta

Compared with the basic XLM-Roberta model report (see Table2), the alternative architecture achieves the same f1-score for non-toxic class and higher f1-score for toxic class, even though overall auc score is not higher than the basic one. Fig2 is showing the AUC curves of these two architectures. There's not a strong evidence to prove alternative architecture performs better.

c. Translation vs. Few-shot Learning

Table4. XLM-Roberta with Translation

	precision	recall	f1-score	support
0.0 (non-toxic)	**0.93** ↑	0.93 ↓	0.93	6770
1.0 (toxic)	0.62 ↓	**0.59** ↑	**0.61** ↑	1230
accuracy			0.88	8000
macro avg	0.77	0.76	0.77	8000
weighted avg	0.88	0.88	0.88	8000

* ↑**or** ↓ in Table4 means increment or decrement compared with Table5.

Table5. XLM-Roberta with Few-shot Learning

	precision	recall	f1-score	support
0.0 (non-toxic)	0.91 ↓	**0.95** ↑	0.93	6770
1.0 (toxic)	**0.65** ↑	0.46 ↓	0.54 ↓	1230
accuracy			0.88	8000
macro avg	0.78	0.71	0.73	8000
weighted avg	0.87	0.88	0.87	8000

* ↑**or** ↓ in Table5 means increment or decrement compared with Table4.

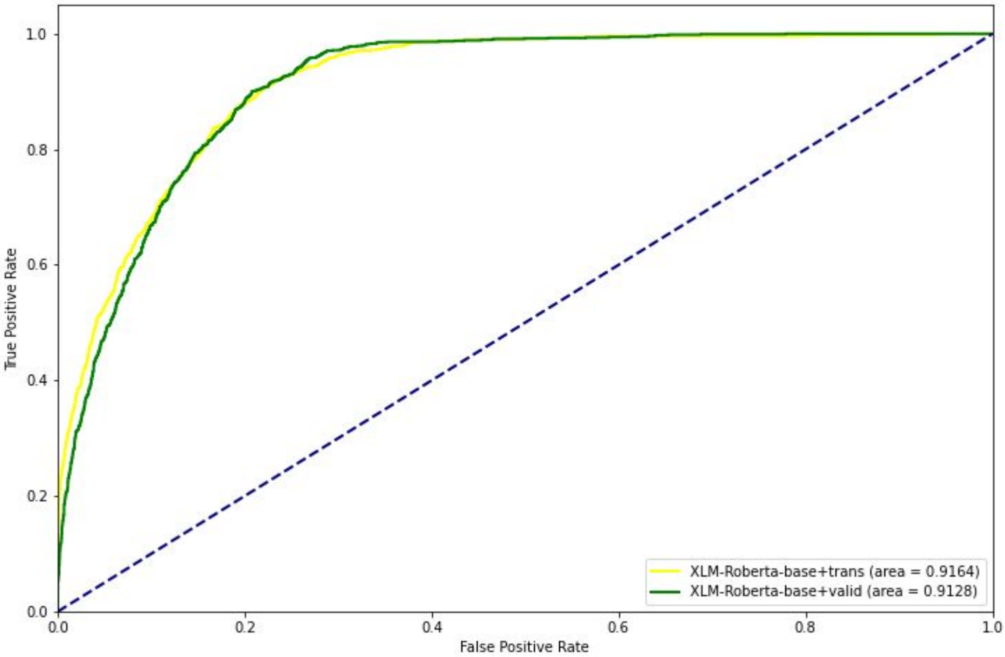

Fig3. AUC curves of two strategies, adding translation vs. using Few-shot learning

In Table4 and Table5, we highlight the spots of two methods. Looking at the f1-scores, the XLM-R model with translation has a better score for classifying toxic comments. Fig3 demonstrates the performances of two strategies. Even though adding translation obtains a little higher auc score, translating multilingual languages costs a lot because of enormous demands of robust translators. In this aspect, we won't sacrifice time and energy for a tiny improvement.

d. Hyper-parameter tuning

Table6. Best XLM-Robert with Few-shot Learning

	precision	recall	f1-score	support
0.0 (non-toxic)	0.94	0.97	0.96	6770
1.0 (toxic)	0.81	0.65	0.72	1230
accuracy			0.92	8000
macro avg	0.87	0.81	0.84	8000
weighted avg	0.92	0.92	0.92	8000

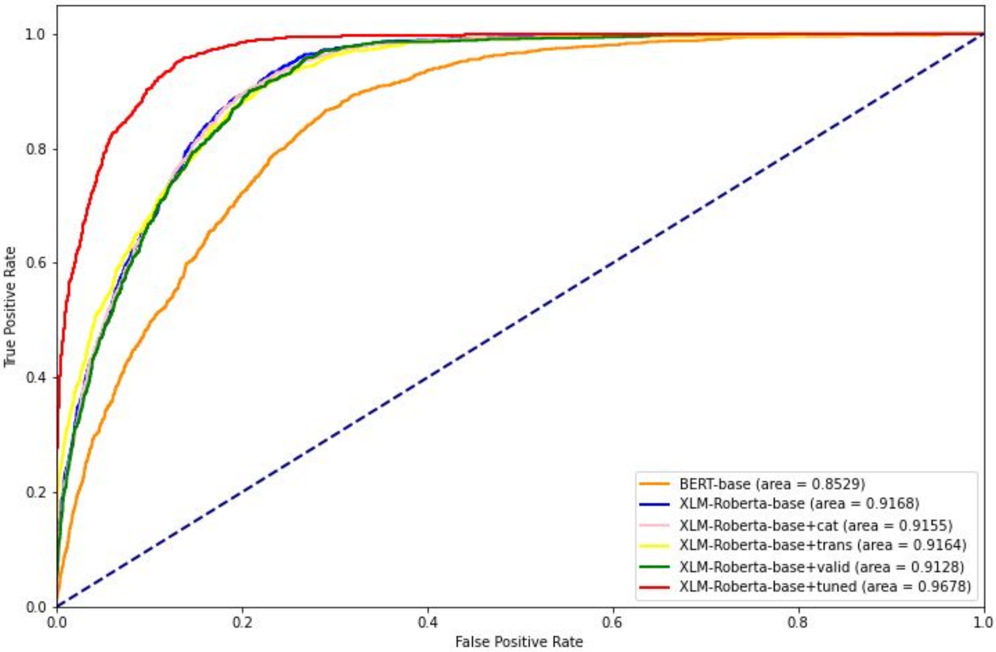

Fig4. AUC curves of all methods

Table6 is the classification report on our best model which is using few-shot learning based on XLM-Roberta model. After hyper-parameter tuning, including adding a dropout layer, increasing downsample rate, adjusting learning rate and so on, we get a significant improvement. Fig4 compares all methods we applied in this project. And the red line represents our best model which outperforms all the others.

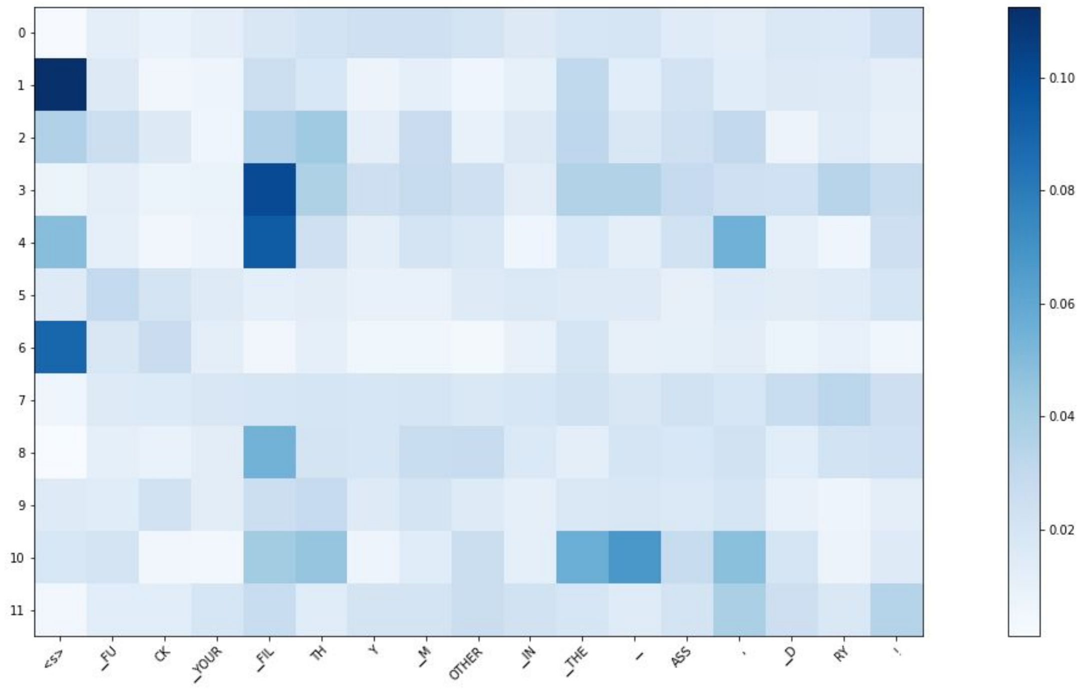

Fig5. “FUCK YOUR FILTHY MOTHER IN THE ASS, DRY!”.

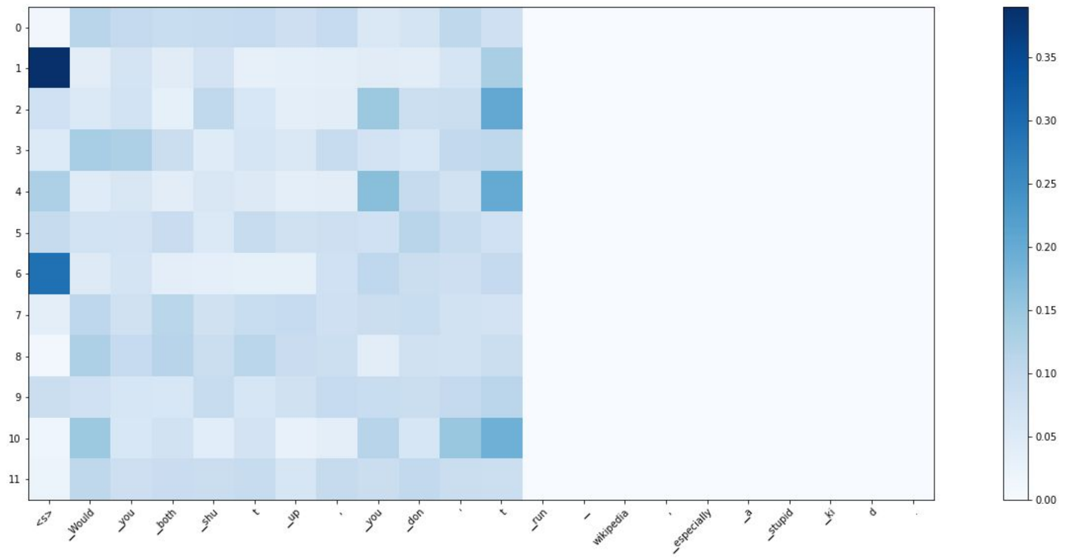

Fig6. “Would you both shut up, you don't run wikipedia, especially a stupid kid.”

In order to prove the validity of our best model, we visualize the attention weights from the last layer. Fig5 and Fig6 are two examples of toxic comments. The darker blue block in figures means more attention on such tokens from a single head. In Fig5, a conspicuous token is “FILTHY”. And in Fig6, the highlight part is “‘t” which is a contraction. Also, due to the restriction of max length of sentence, the latter part of the long sentence is ignored. That might lead to some problems.

e. Competition Results

Our best submission result on Kaggle is achieved by the XLM-R model with fine-tuning on the validation set. The best AUC score evaluated on the Kaggle test set is 0.9239, and our current rank is 340 over 662 participating teams. Below (Table7) is a summary of reference scores from the Kaggle leaderboard:

Table7. Some test scores from Kaggle leaderboard

	Kaggle AUC Score
Current Top 1	0.9508
Our Best Submission	0.9239 (340/662)
Kaggle Benchmark	0.8135

Conclusion

a. Summary

In our experiments, the XLM-R model performs better than the M-BERT model, which could be due to the effectiveness of XLM-R model on cross-lingual understanding and the illustrated improved performance on downstream tasks. Also, the Binary Cross-Entropy loss function suites the best to our task. Comparing the two methods we propose to address the zero-shot learning problem, fine-tuning on validation is more effective than translating the multilingual data for classification, and it is more compatible with the basic architecture. Finally, we achieve over 92% AUC score and beat the benchmark set on Kaggle (81.35%) by over 10%.

b. Limitations

Due to the limited memory problem in GPU/TPU usage on either Google Colab or Kaggle, we only manage to use the down-sampled train data from the first competition and not involve the new dataset present in the current competition. Thus, we fail to observe the effectiveness of the new train dataset. Also, we cannot determine the reliability and quality of the translations as the translation process is not presented in the source, so the potential error propagation could affect our result on the classification of translated datasets.

c. Possible Future Directions

There are a few possible directions that we can explore. First, we can try translating the English training set to multiple targeted languages and build models for each of these languages, and then we can perform classification using the models with corresponding languages in the validation and test set. Second, to boost the training efficiency, we can try allocating multi-cores on TPU by sending parallel tasks, which may allow us to feed in more data and avoid the out-of-memory problem. Third, there are more multilingual pre-trained embeddings that we can experiment on, such as XLM-Roberta Large [6] and MultiFiT [5]. Lastly, we can perform data cleaning during pre-processing, namely filtering out stopwords and non-text comments, which may possibly boost the performance.

References

[1] Challenges for Toxic Comment Classification: An In-Depth Error Analysis. (2018). Retrieved from https://www.aclweb.org/anthology/W18-5105.pdf

[2] Toxic Comment Classification Challenge. (n.d.). Retrieved from https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data

[3] Detecting and Classifying Toxic Comments. (2018). Retrieved from https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1184/reports/6837517.pdf

[4] Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond. (25 Sep 2019). Retrieved from https://arxiv.org/pdf/1812.10464v2.pdf

[5] MultiFiT: Efficient Multi-lingual Language Model Fine-tuning. (10 Sep 2019). Retrieved from https://arxiv.org/pdf/1909.04761v1.pdf

[6] Unsupervised Cross-lingual Representation Learning at Scale. (8 Apr 2020). Retrieved from https://arxiv.org/pdf/1911.02116.pdf

[7] Jigsaw Multilingual Toxic Comment Classification. (n.d.). Retrieved from https://www.kaggle.com/c/jigsaw-multilingual-toxic-comment-classification/discussion/138671

Acknowledgement

We want to give extraordinary thank to our great instructor, Muhammad Abdul-Mageed, who patiently and kindly gives us detailed teaching videos. Also, we appreciate the whole teaching team: Chiyu Zhang, Peter Sullivan and Ganesh Jawahar, who put extra efforts in tutoring and giving us suggestions.