CS505: Final Project – Toxic Spans Detections using BERT-based models

Yinpei Su, Kaijun Wang, Shoki Ko
Department of Computer Science
Boston University
Boston, MA
`{syinpei, kaijwang, shokik}@bu.edu`
`Github Link: https://github.com/syp1997/Toxic-Spans-Detection`

April 30, 2021

Abstract

Span detection has been an important task in the email world since the beginning of the in-ternet history. As the platforms on the internet become more widely spread and diverse, so does the content within them. Toxicity and spam have both been rising with the rise of diversified content; therefore, it is important to automatically identify and detect toxic spans within passages, which became the focus of SemEval-2021 Task-5. Most participants in the contest combined both span prediction and sentiment analysis within the passage to identify phrases that are likely to be toxic spans. In this paper, the approach is to simply use various models, mainly transformer-based models (BERT-based) to train using the given train data. Our best approach achieved high F-1 score of **68.48%** using Google's electra-large-discriminator model, making us in **15th** place in the official competition. P.S. The participants of the competition did not have the test dataset which were used to evaluate their models. We were able to access the dataset and used it for testing our models, but not for training.

1 Introduction

Toxic language can include in various contexts. Some can include racial slurs, sex and gender-related slurs, threats, insults, and discrimination against particular groups of individuals. Automatic detections is not only recommended, but by looking the scale of the Internet, but also necessary to process such vast amount of information. According to a paper, there has been 900% increase in hate speech against Chinese people in Twitter, 200% increase in traffic to sites targeting Asians, 70% increase in hate speech among teens and children in online chat, and 40% toxicity on gaming platforms. Since there are some private communities such as Reddit subreddit pages, they rely individual, sometimes untrained moderators to identify hate speech and toxic spans in the communities, which in-creases their work and mental pressure. Hence, it also increases the necessity of automating the process of toxic span detection. In this paper, we discuss our method to automate the process by identifying the toxic span phrases in sentences in character level using Transformed-based models (Vaswani et al., 2017), including SpanBERT (spanbert-base-cased), MobileBERT (mobilebert-uncased), ELECTRA (electra-large-generator, elecctra-large-discriminator, electra-base-discriminator), BERT Base (bert-base-uncased) and BERT Large (bert-large-uncased).

2 Related Works

2.1 Convolution Neural Networks for Toxic Comment Classification [1]:

As opposed to the traditional Bag-of-Word approach, the paper described a method utilizing Convolution Neural Network (CNN), a Deep Learning approach, which gives a better performance with the rise of computational power and vast amount of big data used for analyzing and training. The paper aims to classify with either a phrase of word or a whole sentence. During the experiment, Georgakopoulos et al. train their models using Kaggle Datasets and Wikipedia corpus. They explore the accuracy and False discovery rate of each models, measuring the ratio that the models have mistakenly predicted a “non-toxic comment” as a “toxic comment”. The paper concluded that CNN models have the highest accuracy and lowest False Discovery Rate, out-performing methods such as traditional Bag-of-Words approach and SVM classifier.

2.2 Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation [2]

In this paper, Ling et al. explore the possibility of POS tagging (part-of-speech tagging) with independent vectors per letter using bidirectional LSTMs. The main purpose of the paper is to benefit the training pipeline in morpho-logically languages, types of languages that has “signifi-cant information is expressed morphologically, e.g. via word-level variation…” [3], which refers to languages such as Arabic, Hebrew, or Turkish. The paper explains that bidirectional LSTM has been a useful tool to perform POS tagging, up to 97% accuracy.

3 Models

3.1 ELECTRA [3]

Models such as BERT replaces some tokens with [MASK] token to train a model and reconstructs the original token when predicting. They usually produces state-of-the-art results and have been proven over time; however, they usually require vast amount of training data and computation to become effective. ELECTRA, on the other hand, instead of masking the input tokens and predicting the original token, uses a generator to generate the original token and uses a small discriminative model that predicts whether the token was generated by the generator or not. With the same computation, data, and model size, ELECTRA outperforms GPT. ELECTRA is much more effective when given smaller data size and computational power.

3.2 BERT [4]

BERT provides state-of-Art performs for NLP tasks. Just like ELECTRA, it is also developed by Google Engineers and trained with Wikipedia corpus, which are large, unlabeled corpus. BERT is deeply bi-direction, meaning that it captures both right and left context in a sentence. When predicting a word in a sentence, BERT can view the context before as well as predict the context of the right side. Therefore, it is a Masked Language Model (MLM). We can mask a phrase in a sentence, and BERT can predict the masked word by capturing the context from both right and left of the sentence.

4 Methodology

4.1 Data preparation

Given the text and labels provided is character-level, the first thing we should do is encoding and translating label to token-level in order to fit in to pre-trained model such as BERT and ELECTRA. The basic idea is when a token is overlapping with toxic span(label provided), we label it as toxic with digit ’1’. An example of the process is demonstrated in Figure 1. After training, we also decode the output to the original format for the convenience of evaluation.

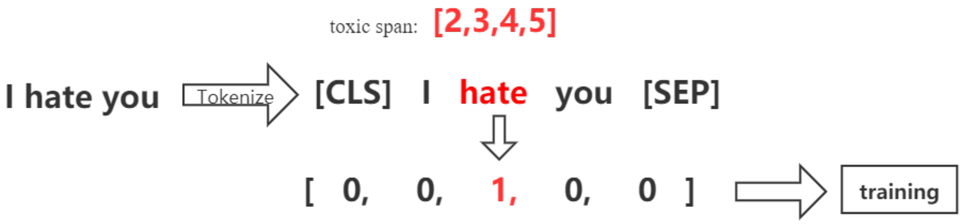

Figure 1: Encode label from character-level to token-level

Table 1: Model Performance v.s. Max Seq Len

Max Seq Len	Valid F-score	Valid Recall	Valid Precision	Test F-score	Test Recall	Test Precision
64	61.67	65.43	67.93	63.22	71.37	61.29
128	66.49	70.33	73.88	63.34	74.53	60.15
192	66.20	70.81	73.75	64.85	**74.62**	62.14
256	67.97	70.45	**76.11**	**65.16**	73.97	62.88
320	**68.22**	**71.71**	75.95	65.10	73.47	**63.17**

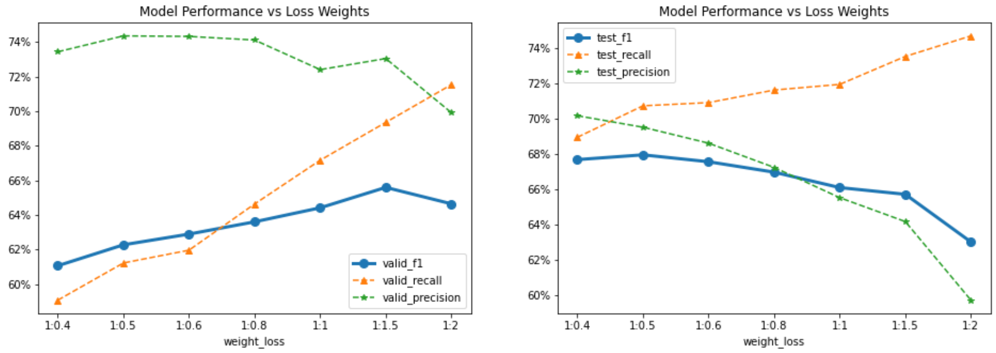

Figure 2: Model Performance for BERT-Base v.s. Loss Weight

4.2 Data Augmentation

Considering the fact that a toxic token in one sentence is also regarded as toxic in another sentence in a large probability, we randomly exchanged toxic spans in different sentences. Since it is also important to ensure the consistence between the original data and augmented data while changing labels, We set 6 characters as the upper limit for the change in each sentence.

4.3 Training Strategy

We use BERT-base as our baseline pre-trained model. Then we explore various hyperparameters, mainly different max length of input sequence and different ratio of loss weight, and used cross-validation to select optimal hyperparameter settings. After finding the best combination, we first pretrained the model on augmented data and fine-tune on the original data.

5 Experiment

5.1 Experimental Settings

Our implementation was based on Transformers. We used BERT-base as our baseline model, and split train set and valid set into 8:2 to find best hyperparameter settings on valid set. The learning rate is 1e-5 and the warm up step is 10% of the total training steps. All the models are trained on Nvidia P100 and V100 GPU.

5.2 Hyperparameter Settings

We mainly adjust two hyperparameters: max sequence length and weight of Cross-Entropy loss. Shown in the Table 1, when the max sequence length more than 128, the improvement is small. Because longer sequence length leads to higher time complexity, we use 128 length in the next experiments. Due to the imbalance between toxic spans and non-toxic spans, we use a weighted Cross-Entropy loss. Shown in Figure 2 We set the weight of loss of toxic spans from 0.4 to 2.0, the larger number means the model prefer to predict tokens as toxic spans, so the recall will be higher. The best weight is 1:1.5 on valid set. However, after several submissions on test set, we found the precision affect the F-score on test set a lot, instead recall valid set. It may because the test set contains more non-toxic spans than valid set. For alleviate the imbalance, we used weight of 1:1 as our default hyper-parameter.

Table 2: Performance of different models

Model	**Valid F-score**	**Test F-score**
BERT-base	64.41	66.10
BERT-large	66.51	67.85
ELECTRA-base	65.41	65.56
ELECTRA-large	68.01	**68.41**
SpanBERT	67.95	65.68
MobileBERT	63.29	61.71
ELECTRA-large + augmentation	-	**68.48**

5.3 Models Comparison

In Table 2 we report the results for the best configuration of each of our models both in the valid and test sets. We found ELECTRA-large has the best performance on both valid and test set. After finding the best model and best configuration, we used our data augmentation strategy. We first train the model on augmented data and then fine-tuned on original data. Since we use the whole training set for augmentation, we would use the trial set for validation to find the model of best performance. Due to the change of valid set, we directly compare the performance on test set, instead of valid set. We got the best F-score of **68.48%** on test set, ranked **15th** in the competition out of 91 teams.

6 Discussion and Future Works

In our experiment, we found out that ELECTRA-large performed the best among the selection of models we have. We believe this is because ELECTRA-large with more parameters can handle the complexity of the task, and the original pre-training objective of ELECTRA is very similar to our task. We also identified that the imbalance of labels in the dataset (with more non-toxic than toxic phrases in a sentence) has plateaued the performance. We explored the possibility of adjusting the loss weight, which did perform better in some dataset, but does not generalize in all situations. As we read other papers of participants, they have not discussed this problem. We believe that this issue is subject to be explored in future studies. Additionally, as multiple parts of the countries are also facing the same problem, Cross-Lingual Language Model (XLM) [5] can become one of the fields to be explored.

References

[1] Spiros V. Georgakopoulos, Sotiris K. Tasoulis, Aristidis G. Vrahatis, and Vassilis P. Plagianakos. Convolutional neural networks for toxic comment classification. *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, 2018.

[2] Wang Ling, Chris Dyer, Alan W Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo, and Tiago Luis. Finding function in form: Compositional character models for open vocabulary word representation. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

[3] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. Electra: Pre-training text encoders as discriminators rather than generators, 2020.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[5] Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining, 2019.