A white hat approach to fighting online trolls: Experiments with BERT and GAN

Jiaying Huang
Management Science & Engineering
Stanford University
hjy1227@stanford.edu

Zhihao Lin
Stanford University
zhl@stanford.edu

Abstract

We investigate multi-class text classification using a two-stage model architecture inspired by Google's Pre-training of Deep Bidirectional Transformers (BERT). Our goal is to build a powerful model to classify toxic comments and machine-generated statements online. Apart from the original data set collected from Wikipedia, we also generate 'fake' toxic comments by using Textgenrnn and MaliGAN model. In the first stage of our classification task, we experiment with different word embeddings such as BERT, Glove, word2vec to retrieve the feature representations of texts. In the second stage, we try different models including BERT, CNN, and RNN-LSTM. Pre-trained models are fine-tuned on our specific task. Based on the generated texts and the two-state model, we are able to achieve a high AUC of over 99% and successfully classify the generated 'fake' comments among other classes.

1 Introduction

In cybersecurity, the term "white hat" refers to an ethical computer security expert who uses his / her skills to attempt to circumvent the security defenses of an organization's information systems. In contrast to a black hat hacker who has malicious intentions, a white hat hacker hacks under good intentions and with permission. The goal of a white hat hacking is to assess the robustness of cybersecurity systems so that the overall level of security could be raised. Taking a leaf from cybersecurity, we first build classifiers to detect abusive online language and then use these "real" toxic comments to generate new ones and test the robustness of the classifiers against machine generated negative language. While there are no known attacks that utilize machine generated abusive language, it is not hard to imagine such a scenario happening given the openness of the deep learning community and the presence of advanced persistent threats (APT). Indeed, in March 2019, OpenAI refused to release their full GPT-2 model to prevent people from using the tool to "generate deceptive, biased, or abusive language at scale." [1]

There are two goals of this study. First, we attempt to improve the state-of-the-art in toxicity detection with Google's BERT model, using both its feature-based approach (which uses the pre-trained representations as additional features to the downstream task) and its fine-tuning based approach (which trains the downstream tasks by fine-tuning pre-trained parameters). For comparison, we used other word embeddings such as GloVe and word2vec as well as models such as CNN, LSTM and Bidirectional LSTM. Second, we attempt to generate toxic comments with a multi-layer RNN (textgenrnn) and a Maximum-Likelihood Augmented Discrete Generative Adversarial Networks (MaliGAN) [2]. We then run the classifiers from the first part of the study to test their robustness against machine generated negative language.

CS230: Deep Learning,Winter 2019,Stanford University,CA. (LateX templte borrowed from NIPS 2018.)

2 Related Work

Word embedding techniques were proposed to generate vector representations of texts. Tang et al.[2] proposed a neural network to learn document representation, with the consideration of sentence relationships. It first learns the sentence representation with CNN or LSTM from word embeddings. Then a GRU is utilized to adaptively encode semantics of sentences and their inherent relations in document representations for sentiment classification. Xu et al.[3] proposed a cached LSTM model to capture the overall semantic information in a long text. Yang et al.[4] proposed a hierarchical attention network for document level sentiment rating prediction of reviews. Li et al.[5] proposed an adversarial memory network for cross-domain sentiment classification in a transfer learning setting, where the data from the source and the target domain are modelled together. Google's BERT[6] model use a multi-layer bidirectional Transformer encoder. In the pre-training procedure the model is trained with two unsupervised prediction tasks on a large corpus. In the fine-tuning procedure, the final hidden state can be adjusted for different tasks including text classificaiton.

3 Dataset

Our dataset consists of comments from Wikipedia's talk page edits. The data contains the id of example, the comment text and its different labels including toxic, severe toxic, obscene, threat, insult and identity hate. The data set has 160k examples in total. As we can see in the chart below, the data is not very balanced because the majority of the comments are non-toxic. Only about 10 percent of the comments are classified into the 6 toxic categories. Among the 6 toxic categories, the class 'toxic' has the most comments while 'threat' has the smallest number of comments. We also use word cloud to visualize our dataset. In the picture below we can obviously see that the toxic category contains many offending words and expressions while the non-toxic comments are all about the article or the descriptions of the web page.

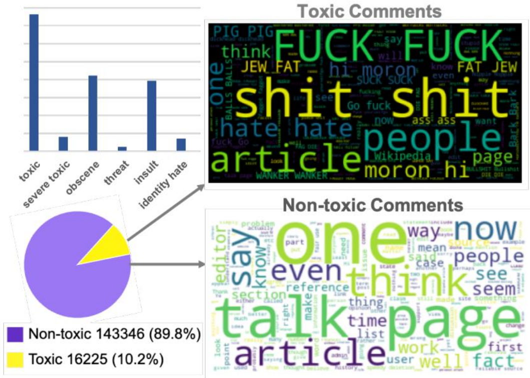


Figure 1: Distribution of various classes within the original dataset and word clouds

4 Methods

Our methods include both generating the fake toxic comments using our original dataset and building classification models to identify toxic and fake comments. First we will introduce the two models we use to generate the 7th class(which is "generated data") in our dataset. Then we will describe our models in detail, including the word embeddings and the classification models.

4.1 Text Generation

We first separated our original dataset into toxic and non-toxic parts. Then we experimented with two models to generate the toxic comments using the toxic comments in our origninal dataset as the input. The models we used include Textgenrnn and MaliGAN.

The first generating model we use is MaliGAN. Like most text GANs, Maligan trains a discriminator (D) to minimize binary loss between real and generated text. What is unique about Maligan is a novel objective for the generator (G) to optimize, using importance sampling, which makes the training procedure closer to maximum likelihood (MLE) training of auto-regressive models, and thus more stable and with less variance in the gradients.

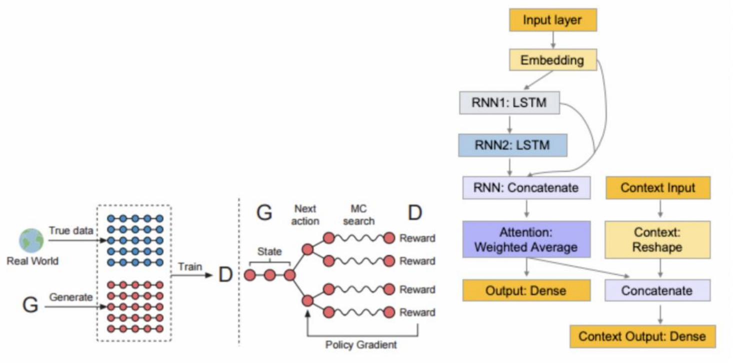


Figure 2: Model Architecture of MaliGAN and Textgenrnn

The second model we use is Textgenrnn. We fine-tuned the model by retraining all the layers in the model. For the default model, textgenrnn takes in an input of up to 40 characters, converts each character to a 100-D character embedding vector, and feeds those into a 128-cell long-short-term-memory (LSTM) recurrent layer. Those outputs are then fed into another 128-cell LSTM. All three layers are then fed into an Attention layer to weight the most important temporal features and average them together (and since the embeddings + 1st LSTM are skip-connected into the attention layer, the model updates can backpropagate to them more easily and prevent vanishing gradients). That output is mapped to probabilities for up to 394 different characters that they are the next character in the sequence, including uppercase characters, lowercase, punctuation, and emoji.

4.2 Classification Model

Our goal is to achieve high accuracy on the classification of different comments. We experimented with three word emebeddings, including Word2Vec, GloVe and BERT emeddings and three model architectures including CNN, LSTM, and LSTM with attention layer. In order to explore the effect of different embeddings and models, we create an evaluation matrix to show the performance of different combinations. Finally, we trained and fine-tuned the BERT model on our dataset and compare the performance of BERT model with other 9 combinations of embeddings and models.

Inspired by BERT, our model consists of two stages. In the first stage we try different word embeddings including BERT embedding, Glove and word2vec. In the second stage, we make use of BERT's pretrained model and apply fine-tuning method to fit our classification task. The models we try include BERT, CNN and RNN-LSTM. BERT's model architecure is a pre-trained multi-layer bidirectional Transformer encoder.

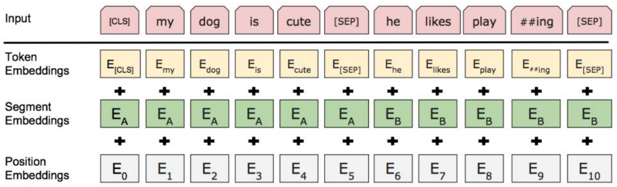


Figure 3: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

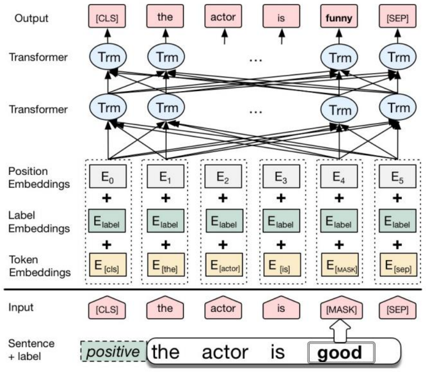


Figure 4: BERT pre-train model architecture

5 Results and Discussion

5.1 Text Generation

We use Textgenrnn and MaliGAN models to generate data based on our original dataset. Both models are trained with 20 epochs. The MaliGAN model achieved minimal loss after 8 epochs and the Textgenrnn after 13 epochs. Since the two models use different loss functions, we do not compare their performance based on the value of their validation loss.

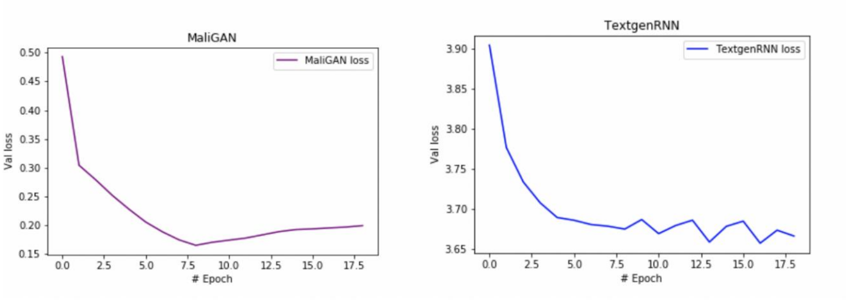


Figure 5: Validation results of MaliGAN and Textgenrnn

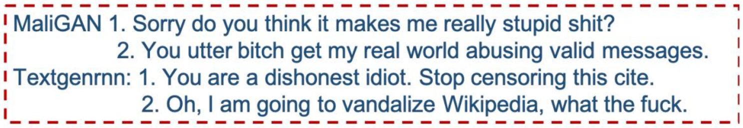

MaliGAN 1. Sorry do you think it makes me really stupid shit?
2. You utter bitch get my real world abusing valid messages.
Textgenrnn: 1. You are a dishonest idiot. Stop censoring this cite.
2. Oh, I am going to vandalize Wikipedia, what the fuck.

Figure 6: Examples generated by textgenrnn and maligan

Below are some examples of the generated texts. It obvious that both models are able to generated texts that are close to human language. They captured some of the characteristics of the toxic comments and were able to mimic some of the offensive statements. Although there might be some grammar mistakes or some incomprehensible sentences, our models in general produced comments of good quality.

5.2 Results on Original Data

The results of training the different word embeddings and models on the original dataset are shown in the Figure 7 and the validation results of the best models from training are in Figure 5. The BERT model achieves near perfect performance with an AUC of 0.994. Compared with other word embeddings and models, the pre-trained BERT model achieved minimum loss after only 2 epochs and the validation loss is much smaller than other models'. Another finding is that adding an attention layer on our model can boost our model performance. From the table below we can see that after adding an attention layer to our three models, the performances are better than the previous models.

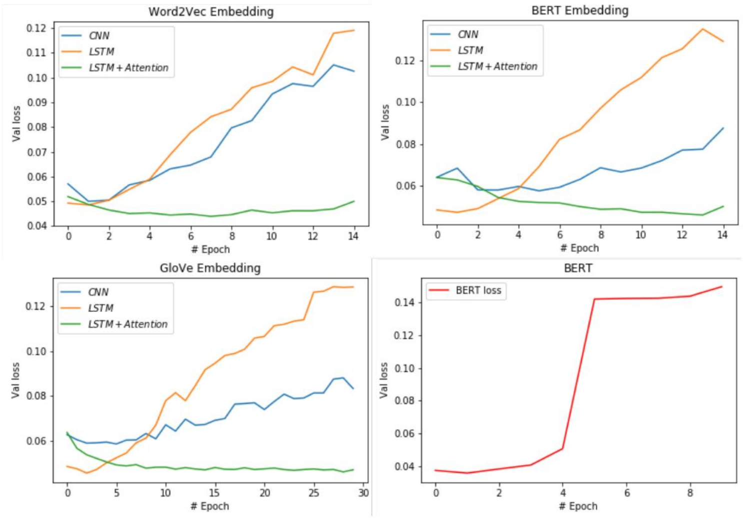
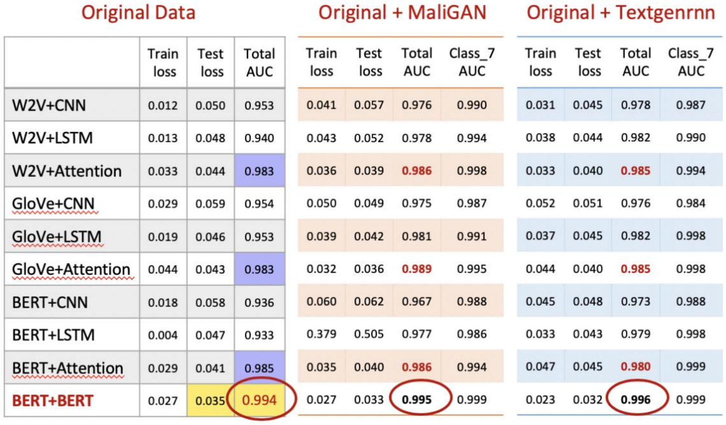


Figure 7: Training results on original dataset

5.3 Results on Generated Data

We then added the machine generated toxic comments into the original dataset and then run the classifiers on the combined dataset and the results are shown in Figure 8. Once again, the BERT model achieves near perfect performance with an AUC of 0.995 and has no problem detecting the generated toxic comments (Generate AUC).



	Original Data			Original + MaliGAN				Original + Textgenrnn			
	Train loss	Test loss	Total AUC	Train loss	Test loss	Total AUC	Class_7 AUC	Train loss	Test loss	Total AUC	Class_7 AUC
W2V+CNN	0.012	0.050	0.953	0.041	0.057	0.976	0.990	0.031	0.045	0.978	0.987
W2V+LSTM	0.013	0.048	0.940	0.043	0.052	0.978	0.994	0.038	0.044	0.982	0.990
W2V+Attention	0.033	0.044	0.983	0.036	0.039	**0.986**	0.998	0.033	0.040	**0.985**	0.994
GloVe+CNN	0.029	0.059	0.954	0.050	0.049	0.975	0.987	0.052	0.051	0.976	0.984
GloVe+LSTM	0.019	0.046	0.953	0.039	0.042	0.981	0.991	0.037	0.045	0.982	0.998
GloVe+Attention	0.044	0.043	0.983	0.032	0.036	**0.989**	0.995	0.044	0.040	**0.985**	0.998
BERT+CNN	0.018	0.058	0.936	0.060	0.062	0.967	0.988	0.045	0.048	0.973	0.988
BERT+LSTM	0.004	0.047	0.933	0.379	0.505	0.977	0.986	0.033	0.043	0.979	0.998
BERT+Attention	0.029	0.041	0.985	0.035	0.040	**0.986**	0.994	0.047	0.045	**0.980**	0.999
BERT+BERT	0.027	0.035	0.994	0.027	0.033	**0.995**	0.999	0.023	0.032	**0.996**	0.999

Figure 8: Validation results on original dataset + generated toxic comments using the best model from training

6 Conclusions

The prowess of Attention is obvious - both BERT (Bidirectional Encoder Representations from Transformers) and BiLSTM Attention perform superior to LSTM and CNN across different word embeddings. It is not surprising that the BERT model performed the best – the model is now considered the state-of-the-art in NLP as evident from its results on SQuAD v1.1.Our results imply that even with access to modern text generation models such as MaliGan and Textgenrnn, it will be difficult for motivated malicious actors to trick abusive language classifiers.

Contributions

Both team members contributed to the project. Jiaying built 8 models, fine-tuned the models to obtain better results, ran the 8 models on original and generated texts on AWS, and wrote the poster and paper. Zhihao built the LSTM model, extracted the embeddings from BERT, trained and fine-tuned the BERT model, generated toxic comments with MaliGAN and Textgenrnn, and wrote the poster and paper. Our code is available at: https://github.com/kristen-h/CS230_Project, https://github.com/zhihaolin/bert-toxic-comments-multilabel, https://github.com/zhihaolin/CS230-textgenrnn and https://github.com/zhihaolin/Texygen.

References

[1] https://www.theregister.co.uk/2019/03/20/openai_language_model/

[2] T. Che, Y. Li, R. Zhang, R. D. Hjelm, W. Li, Y. Song, and Y. Bengio. Maximum-likelihood augmented discrete generative adversarial networks. arXiv preprint arXiv:1702.07983, 2017.

[3] Tang D, Qin B, Liu T. Document modelling with gated recurrent neural network for sentiment classification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2015), 2015.

[4]Xu J, Chen D, Qiu X, and Huang X. Cached long short-term memory neural networks for document-level sentiment classification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2016), 2016.

[5] Yang Z, Yang D, Dyer C, He X, Smola AJ, and Hovy EH. Hierarchical attention networks for document classification. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016), 2016.

[6] Li Z, Zhang Y, Wei Y, Wu Y, and Yang Q. End-to-end adversarial memory network for cross-domain sentiment classification. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2017), 2017.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.