Toxic Comments Detection in A Semi-supervised Way

1 Introduction

In this report, we focus on the problem of toxic comments detection. The wide spread use of social media allowed for a more connected and informed world. However, it also give rise to the use of toxic comment, including racist, hatred comment, and so on. This phenomenon makes discussing things more difficult, many people stop exchanging different opinions due to these harmful expression.

To maintain a better online environment, platforms struggle to detect these toxic comments and filter them, so that online conversation can become more productive and respectful. However, due to the ambiguity and flexibility of language, current approach rely heavily on manually labeling, which is not only time-consuming but also cause a great mental burden on inspector stuff. An effective algorithm is in desperate need. Our work is then motivated by such necessity.

In this project, we look deep into past algorithm on toxic comment detection, giving comparison of different methods including LSTM, CNN, GRU and BERT. Afterwards, we propose an efficient semi-supervised tri-train classifier, which is the improvement of the co-train methods, combining LSTM, CNN, GRU together to get a better result, and utilizing unlabeled data in an iterative way. Such a method can greatly reduce labeled data amount needed, and improve accuracy.

We perform adequate experiment on a competition dataset proposed by kaggle[1], results proved the effectiveness of our proposed methods in comparison to traditional supervised methods.

2 Related Work

Toxicity detection in natural languages is essentially a problem of sentiment classification.

[13] firstly researched on comment abuse classification, with application of combining TF-IDF with sentiment contextual features, reporting a 6% increase in Fl score of the classifier on chat style datasets (Kongregate, MySpace). Since then, this area has been intensively researched in the past few years.

[1] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

most of these research base their work on social media data. Approaches for sentiment analysis can be roughly classified into three categories: lexicon-based methods, machine learning approaches, deep-learning methods.

2.1 Lexicon-based Methods

Lexicon-based approaches focus on proper feature engineering, they use precompiled sentiment lexicons containing different words and their polarity to classify a given word into positive or negative sentiment class labels. [11] started the task of sentiment analysis using the lexicon method in 1966. Later, different lexicons were proposed such as WordNet, WordNet-Affect, SenticNet, MPQA, and SentiWordNet. However, the construction of the sentiment lexicon construction for today's user-generated unstructured data is a challenging task.

2.2 Machine Learning Methods

Machine learning approaches help to alleviate the problem of lexicon-based. [12] pioneered the use of these techniques for sentiment analysis.Machine learning approaches are preferred for sentiment analysis due to their capacity for dealing with large amounts of data compared with lexicon based approaches[1]. However, in case there are no human annotated datasets, machine learning methods can be hard to apply.

2.3 Deep-learning Methods

Deep-learning approaches in sentiment analysis has been driven by their ability of automatic feature learning, where they can learn automatically and discover discriminative input representations from data themselves. Their adoption has been motivated by the increase of the training data with multi-class classification and the success of word embeddings[5].

Kim [4] suggested a simple, yet efficient CNN model with two channels where each channel consists of a single convolution layer followed by max-pooling over time for sentence level sentiment analysis. Since then, various kinds of model have been put into use of toxic comments detection, such as Long-short term memory model(LSTM) and Gated Recurrent Neural Networks(GRU).

[2] compared the performance of various deep learning approaches to this problem specifically using both word and character embeddings. They assessed the performance of recurrent neural networks with LSTM and word embeddings, a CNN with word embeddings, and a CNN witl character embeddings. The best performance they achieved was a 93% accuracy using the character level CNN model.

[7] proposed a model for sentiment label distribution that involved a combination of using a DeepCNN for character-level embeddings (in order to increase information for word-level embeddings) with a Bidirectional LSTM which produced sentence-wide feature representations from word-level embeddings. This model attained a best prediction ac-

curacy of 86.63% on the Stanford Twitter Sentiment Corpus. Their findings indicate the prospective advantages of utilizing LSTM and DeepCNN models on the task of toxicity classification.

3 Method

3.1 Semi-Supervised Tri-Train Algorithm

Our goal is to utilize massive unlabeled data in real world to help detect toxic comments, so we proposed a semi-supervised algorithm which is based on a tri-train algorithm which is the improvement of the co-train. In some past studies, the co-train has been commonly used for text categorization[9]. While the co-train always have rigid requirements on features, they must be extracted from the multi-views such as video or voice.

In this paper, we try to use tri-train to avoid that limits and it has also been shown to have the siginificant improvments on many models. The basic classifiers are three deep learning algorithms(LSTM, GRU, CNN), as well as boosting ideas. These algorithms are combined to achieve semi-supervised classification. The specific process is shown in 1, which can be explained as follows:

Step 1 The last 10,000 samples of the data set are taken as test set T, and the rest as training set D. So we have more than 140,000 samples in the training set.

Step 2 The labeled training set D is resampled k times to get k sub-training sets $D_i, i = 1, 2, ..., k$. The remaining data of resampling is U_i, and we delete its labels as unlabeled data set. Among them, k represents the number of classifier algorithms used. In this project, we selected three algorithms (LSTM, GRU, CNN), so $k = 3$.

Step 3 After training D_i respectively, three classifiers are obtained, which predict the unlabeled data sets U_i and get the prediction results $f_i(U_i), i = 1, 2, 3$.

Step 4 In the prediction results of U_i, if the prediction results of samples are the same as the labels of other classification algorithms, these samples are added to D_i, and this part of samples is deleted in U_i.

Step 5 Repeat **step 3** and **step 4** until the samples in D_i do not change and stop the iteration.

Step 6 The three classifiers f_i are tested on the testing set T, and the final result is in accordance with the principle that the minority is subordinate to the majority.

3.2 Baseline—Supervised Algorithm

Here we give introduction of four baseline methods(LSTM, Bert, GRU, CNN) used in our proposed model, as a part of complete survey, we also train them on the same trainning set$D_i, i = 1, 2, ..., k$, test the classifiers on the testing set T and compare the results with the semi-supervised algorithm.

3.2.1 LSTM & GRU

LSTM model is composed of input word X_t, cell state C_t, temporary cell state C'_t, hidden layer state h_t, forgetting gate f_t, memory gate

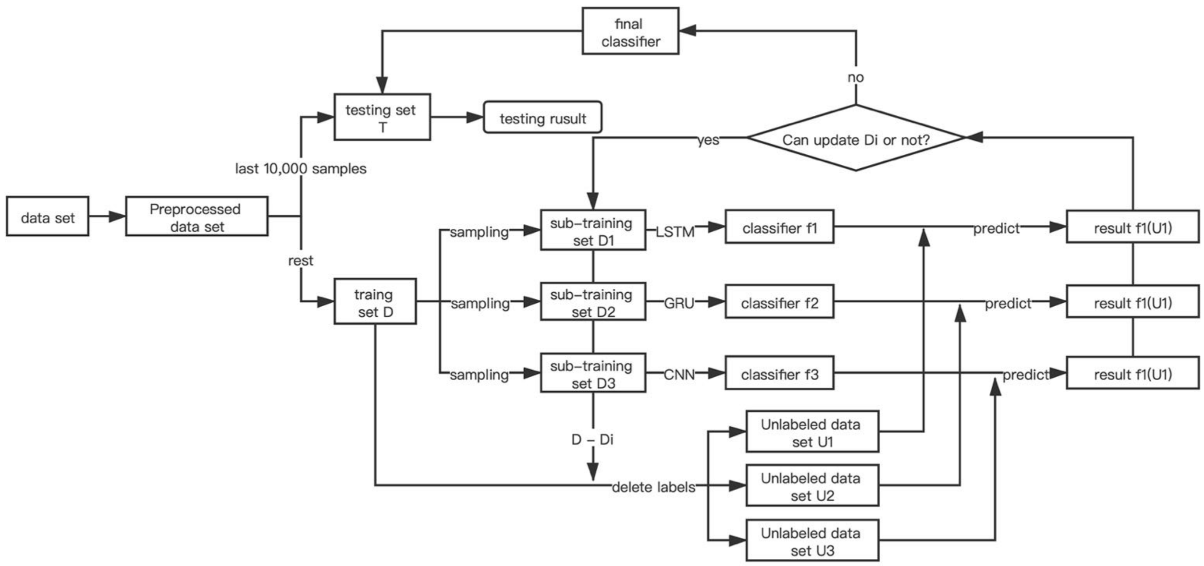

Figure 1: Algorithm flowchart

i_t and output gate o_t at time t. The calculation process of LSTM can be summarized as follows: by forgetting and memorizing the information in the cell state, the useful information for the subsequent calculation can be transmitted, while the useless information is discarded, and the hidden layer state h_t will be output at each time step, in which forgetting, memorizing and outputting the forgetting gate f_t and memory gate $i_t - 1$ calculated by the hidden layer state h_{t-1} of the previous time and the current input X_t It is controlled by the output gate o_t.

BI-LSTM model is the combination of forward LSTM and backward LSTM, which combines sequence order and flashback into a new sequence, so as to fully consider the influence of sentence context. For example, if the sequence a, b, c, d, after reassembling, the input is $[a, d], [b, c], [c, b], [d, a]$. The bidirectional structure of our loop network is shown in next page.

GRU (Gated Recurrent Unit) algorithm is a variant of LSTM, which is also proposed to solve the problems of long-term memory and gradient in back propagation. Its I/O structure is similar to general RNN and its idea is similar to LSTM. Compared with LSTM, GRU algorithm reduces one gating, and combines forge gate and input gate into a single update gate. The cell state and hidden state are also mixed, and some other changes are added. The final model is simpler than the standard LSTM model and is a very popular variant.

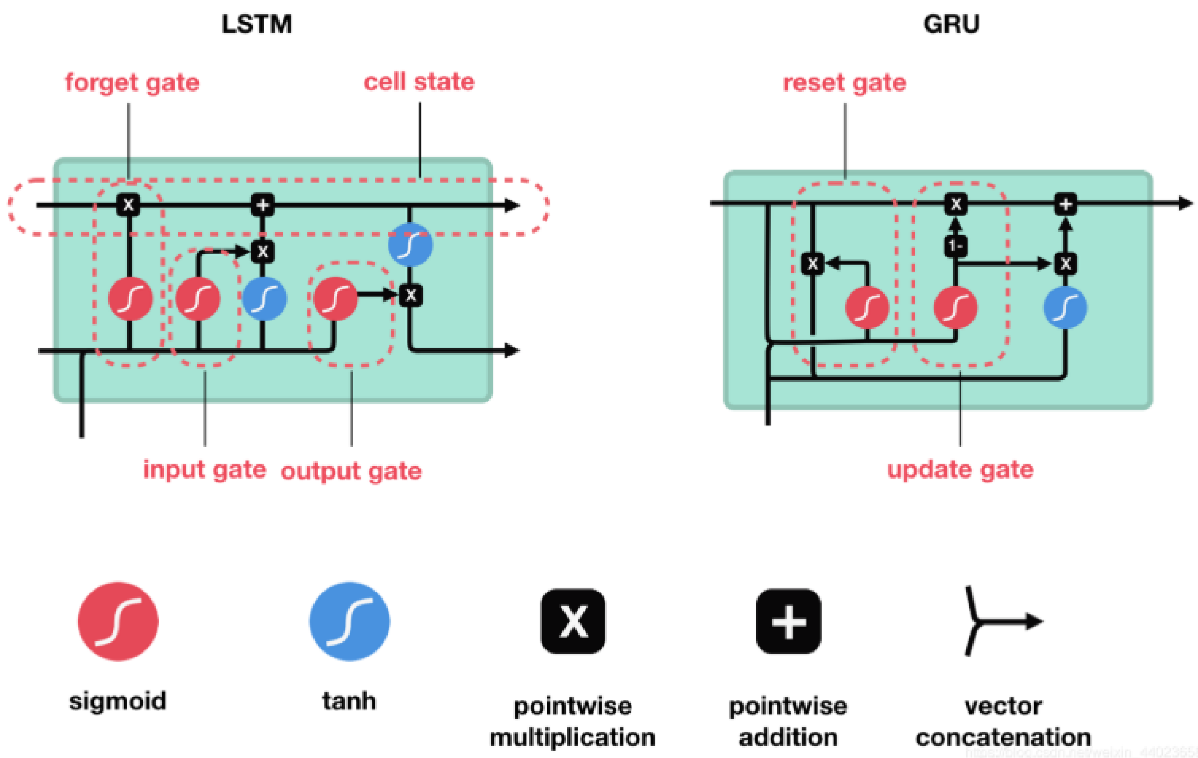

Figure 2: Bi-directional Long-Short Term Memory Model

$$z_t = \sigma(W_x \times [h_{t-1}, x_t])$$
$$r_t = \sigma(W_r \times [h_{t-1}, x_t])$$
$$h'_t = \tanh(W_r \times [h_{t-1}, x_t])$$
$$h_t = (1 - z_t) \times h_{t-1} + z_t \times h'_t$$

Where z_t is the update gate, r_t is the reset gate, which is used as the weight of the new memory, h'_t is used to save the new memory content, and the final output h_t synthesizes the influence of the update memory and the past memory.

As shown in 3, our model first goes through **GloVe** embedding processing, then goes through the coding layer of bidirectional cyclic network (**LSTM** and **GRU**), and the decoding layer selects the superposition of multiple residual layers, and finally outputs the binary classification prediction results.

Each module A is the cycle unit, which is replaced by the cycle unit of **LSTM** and **GRU** respectively. [10]

3.2.2 CNN

CNN model[3] also uses **GloVe** embedding processing. The embedded results go through convolution layer. Convolution layer uses

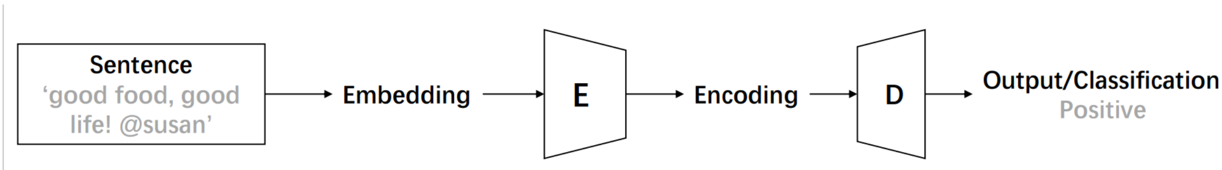

Figure 3: Baseline structure

convolution kernel (2,3,4) of different sizes to extract sentence features. The extracted tensor goes through pooling layer, and then goes through full connection layer to output prediction results. The network structure is as 4.

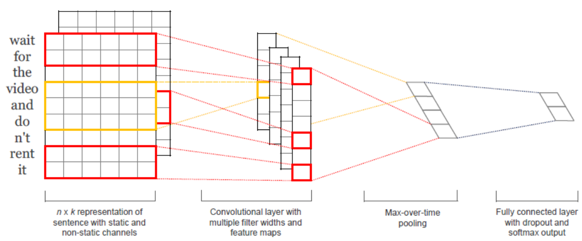

Figure 4: Long-Short Term Memory Model

3.2.3 BERT

The full name of BERT[6] is Bidirectional Encoder Representation from Transformers, that is, the Encoder of bidirectional Transformer, because the decoder cannot obtain the information to be predicted. The main innovation of the model is the pre-train method, which uses *Masked LM* and *Next Sense Prediction* to capture the representation of words and sentences respectively.The model structure is in 5

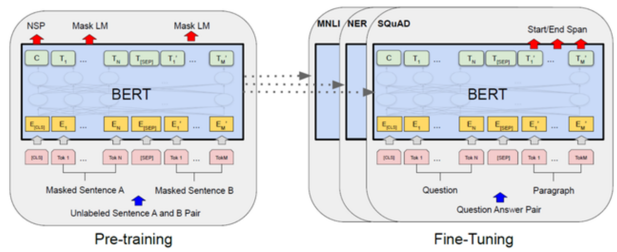

Figure 5: Structure diagram of BERT. The figure on the left shows the pre-training process, and the figure on the right shows the fine-tuning process for specific tasks

After getting the sentence to be input, the word of the sentence is transformed into Embedding, which is represented by E. Different from Transformer, the input embedding of BERT consists of three parts: Token Embedding, Segment Embedding and Position Embedding. The details are shown in 6

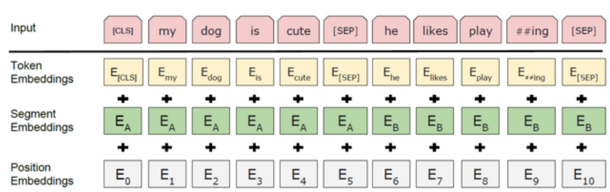

Figure 6: Structure diagram of BERT. The figure on the left shows the pre-training process, and the figure on the right shows the fine-tuning process for specific tasks

3.3 Visualization

To better compare the effectiveness of different methods, we adopt a python package named eli5. This package provides an implementation of LIMEalgorithm [8] which allows to explain predictions of text classifiers by checking what was important in the document to make this decision.

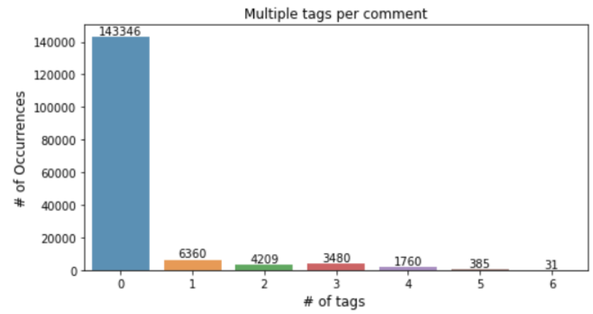

Figure 7: Histogram drawn according to the number of tags contained in it

4 Experiment

4.1 Dataset

We used a Wikipedia comments dataset proposed by Kaggle , including 159,571 training data and 153,164 test data. There are six categories of tags, which are toxic, severe toxic, obscene, insult, threat and identity hate. Labels range between 0 and 1, and a tag value of 1 indicates that the comment contains semantic information of the tag type. In the training set, the data distribution of values corresponding to each tag is as follows.

It can be seen that most of the results are 0, which means that most of the comments in the training set are not toxic comments.

For the data in the training set, the histogram drawn according to the number of tags contained in it (i.e. the number of tags corresponding to the comment is 1) is as 7 (the horizontal axis represents the number of tags and the vertical axis represents the corresponding number of comments).

It can be seen that the tag number of most comments is $<=1$, and only 31 comments contain all 6 tags. Additionally, regarding the length of comments in the training set and the test set, the distribution is drawn as 8

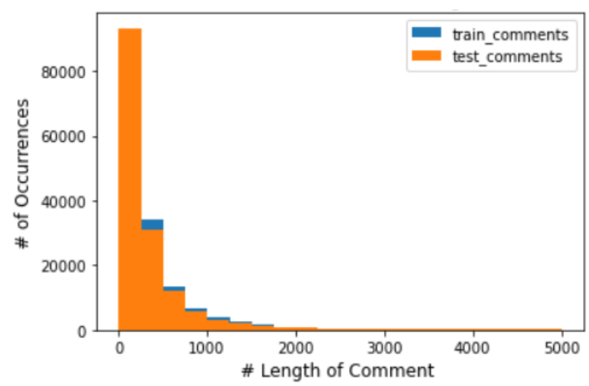

Figure 8: Distribution of comment length

It can be seen that the comment length distribution in the training set and the test set is basically the same.

4.2 Data Preprocessing

In the data preprocessing part, the part of @user is deleted first, which has no effect on the meaning of sentences. Then, some common emoticons:-) and: (are replaced with

toxic	severe toxic	obscene	threat	insult	threat	identity hate
0	144277	157976	151122	159093	151694	158116
1	15294	1595	8449	478	7887	1405

Table 1: Data distribution

smile and sad (because these expressions can reflect the emotions contained in the sentence to a great extent, and deleting them directly will affect the final result), and conjunctions such as didn't are split into did not, and simple reduplicated words such as hahaha are replaced with ha. Turn all uppercase characters into lowercase, delete some insignificant punctuation marks, and keep only? ! (common punctuation marks such as,. Etc. has no obvious effect on sentence meaning, but? ! can effectively reflect the context of the sentence). Finally, short words (word length $<=2$) are deleted, which contain many common words such as I, am, it, is, etc., and have no obvious influence on the sentence meaning.

Twelve kinds of word clouds are made, which correspond to the cases where the six categories of toxic, severe toxic, obscene, insult, threat and identity hate are 0 and 1 respectively. Under the corresponding circumstances, the words with higher frequency are visually highlighted. The corresponding word cloud with toxic=1 is shown in the following figure.

Figure 9: Word Cloud

4.3 Experimental Setting

In terms of data set division, we choose 1000 samples as training set, 10000 sample as testing set. Then we use an additional unlabeled 49000 samples for semi-supervised learning.

For fair comparison, we have done roughly the same experimental settings for different classifiers. In training phase, we set learning rate to 0.0025, batch size as 32. We train each optimizer for 10 epoch, using Adam as optimizer.

More specifically, in **BERT** model, we use 'bert-base-uncased' model, pad or truncate all sentences into fixed length of 128 tokens. We set bias, gamma, and beta parameters with weight decay rate being 0.0, while other parameters have a weight decay rate of 0.01. We train it for 2 epoch as officially recommended

4.4 Results

We make a visualization of our proposed method, take CNN and our method as examples, as shown in 10, 11

Take CNN and semi-supervisor as examples. When they recognize the example sentence, words such as thank you, very much, help, etc. have made a proof contribution to this; but there are still some differences in details

We also provide quantitative comparison between different classifier, with metrics including accuracy, f1score, recall, precision, AUC. The experimental results of 1000 samples from the training set are shown in the 2.

We can clearly see that semi-supervised algorithm performs better than direct deep learning algorithm in the case of small labeled samples. This phenomenon is worth expanding. The sample size is usually a very big limitation of the experimental effect. In most cases in reality, we often can't get a lot of annotated data, or it is very expensive to annotate the data. Therefore, for the case of small data, if we can take the unsupervised learning algorithm, we can get better results.

Next, when we increase the sampling ratio, that is, the number of training samples increases gradually, while the number of unlabeled samples decreases gradually, we find that the effect of semi-supervised learning is not as good as supervised learning. It is easy to explain the result. Semi-supervised learning is to further try to enhance its effect on the basis of existing information. However, when the training sample size has contained sufficient information, semi-supervised learning will lead to worse results when using unlabeled data sets.

However, such a conclusion does not mean that semi-supervised learning is not good. On the contrary, we often encounter unlabeled data sets in real life, while we may need to get a clear classifier more than using unsupervised learning algorithm. At this time, we can usually choose to label a small number of samples manually. In view of the high cost of labeling, we can usually obtain a small number of labeled data sets and a large number of unlabeled data sets. This reality is very suitable for our semi-supervised algorithm. This is our contribution.

5 Division of labor

- Overall arrangement: Yang Rongjian
- Datasets and Related work: Luo Kun
- Algorithm: Chen Lei, Wang Xiao
- Experiment: Hu Zhiyuan, Qian Yudong, Zhang Zeyang, Han Weidong
- Liaison and Translation: Ma Yidi
- Report: Zhang Yuntao

classifier	accuracy	f1score	recall	precision	AUC
GRU	0.850	0.443	0.576	0.360	0.729
LSTM	0.878	0.475	0.534	0.428	0.726
CNN	0.890	0.492	0.519	0.468	0.725
BERT	0.862	0.458	0.575	0.470	0.737
Semi-supervised Methods(Ours)	0.887	0.515	0.583	0.462	0.752

Table 2: Results

References

[1] Ebru Aydoğan and M Ali Akcayol. "A comprehensive survey for sentiment analysis tasks using machine learning techniques". In: *2016 International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*. IEEE. 2016, pp. 1–7.

[2] Theodora Chu, Kylie Jue, and Max Wang. "Comment abuse classification with deep learning". In: *Von https://web.stanford.edu/class/cs224n/reports/2762092.pdf abgerufen* (2016).

[3] Spiros V Georgakopoulos et al. "Convolutional neural networks for toxic comment classification". In: *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*. 2018, pp. 1–6.

[4] Yoon Kim. "Convolutional neural networks for sentence classification". In: *arXiv preprint arXiv:1408.5882* (2014).

[5] Stefan Kombrink et al. "Recurrent neural network based language modeling in meeting recognition". In: *Twelfth annual conference of the international speech communication association*. 2011.

[6] Marzieh Mozafari, Reza Farahbakhsh, and Noël Crespi. "A BERT-based transfer learning approach for hate speech detection in online social media". In: *International Conference on Complex Networks and Their Applications*. Springer. 2019, pp. 928–940.

[7] Huy Nguyen and Minh-Le Nguyen. "A deep neural architecture for sentence-level sentiment classification in twitter social networking". In: *International Conference of the Pacific Association for Computational Linguistics*. Springer. 2017, pp. 15–27.

[8] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "" Why should I trust you?" Explaining the predictions of any classifier". In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, pp. 1135–1144.

[9] Sara Rosenthal et al. "A large-scale semi-supervised dataset for offen-

sive language identification”. In: *arXiv preprint arXiv:2004.14454* (2020).

[10] Saurabh Srivastava, Prerna Khurana, and Vartika Tewari. “Identifying aggression and toxicity in comments using capsule network”. In: *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*. 2018, pp. 98–105.

[11] Philip J Stone et al. “The general inquirer: A computer system for content analysis and retrieval based on the sentence as a unit of information”. In: *Behavioral Science* 7.4 (1962), p. 484.

[12] Suchita V Wawre and Sachin N Deshmukh. “Sentiment classification using machine learning techniques”. In: *International Journal of Science and Research (IJSR)* 5.4 (2016), pp. 819–821.

[13] Dawei Yin et al. “Detection of harassment on web 2.0”. In: *Proceedings of the Content Analysis in the WEB* 2 (2009), pp. 1–7.

y=反例 (probability **0.996**, score **5.540**) top features

Contribution?	Feature
+0.911	for
+0.893	in
+0.722	the
+0.705	very
+0.465	of
+0.441	out in
+0.414	<BIAS>
+0.413	on
+0.407	read
+0.389	ojigbani
+0.382	your
+0.359	very much
+0.291	chris
+0.256	much for
+0.245	t
+0.206	of the
+0.205	first
+0.194	much
+0.191	my article
+0.184	you very
+0.171	in my
+0.165	help
+0.161	the article
+0.157	thank you
+0.145	i
+0.127	message
+0.085	article on
+0.080	and i
+0.078	after i
+0.071	first article
+0.060	it
+0.060	i learnt
+0.049	your message
+0.043	helping me
+0.039	from
+0.030	and
-0.034	for helping
-0.049	your help
-0.050	i read
-0.070	message t
-0.074	article chris
-0.077	update
-0.081	article here
-0.084	chris ojigbani
-0.096	ojigbani it
-0.096	my
-0.128	here and
-0.162	it was
-0.184	after
-0.193	update of
-0.236	out
-0.256	ojigbani thank
-0.270	from your
-0.341	me
-0.366	thank
-0.501	very first
-0.906	you

Figure 10: LAME on CNN

y=反例 (probability **1.000**, score **10.770**) top features

Contribution?	Feature
+1.841	article
+1.483	for
+1.344	in
+1.010	of
+0.852	of the
+0.816	very
+0.769	the
+0.689	the article
+0.672	on
+0.670	much
+0.646	article on
+0.639	in my
+0.554	out in
+0.441	helping me
+0.436	much for
+0.330	very much
+0.303	for helping
+0.274	read your
+0.229	my article
+0.226	thank you
+0.224	t
+0.216	me out
+0.203	first
+0.189	and i
+0.187	i learnt
+0.154	ojigbani
+0.153	you very
+0.138	from your
+0.112	on chris
+0.104	help
+0.092	your
+0.086	ojigbani it
+0.063	here
+0.058	from
+0.058	out
+0.049	very first
+0.026	update of
-0.033	learnt from
-0.038	chris
-0.039	help after
-0.040	helping
-0.076	i read
-0.156	my
-0.187	your message
-0.191	was
-0.225	read
-0.234	learnt
-0.251	first article
-0.262	thank
-0.277	i
-0.280	<BIAS>
-0.365	it
-0.384	ojigbani thank
-0.421	article here
-0.974	me
-1.132	you

Figure 11: LAME on our proposed method