Toxic Comments Analysis Report

Yongchao Qiao, Zhe Fan, Junchi Zhang, Guangji Bai

1. Introduction

1.1 Project Background

Flood of information is produced in a daily basis through the global internet usage arising from the online interactive communications among users. While this situation contributes significantly to the quality of human life, unfortunately it involves enormous toxic comments, which may trigger following dangers:

1. Text arising from online interactive communication hides many hazards such as fake news, online harassment and toxicity.
2. Toxic comment is not only the verbal violence but a comment that is rude, disrespectful or otherwise likely to make someone leave a discussion.
3. Toxic comment can be considered also the personal attack, the online harassment and bullying behaviors.

1.2 Previous Work

The problem of online toxic comments has attracted the attention from academia many years ago. For a text classification problem, many models like Naïve Bayes and Support Vector Machine (SVM) were applied to build the classifier and achieved different level of success (according to Kevin Khieu's paper, Detecting and Classifying Toxic Comment). In recent years, with the development of parallel computing, deep learning becomes more and more popular and provides more possibilities for traditional NLP problems. Researchers have built Recurrent Neural Network or Convolutional Neural Network models (according to Spiros V. Georgakopoulos's paper, Convolutional Neural Networks for Toxic Comment Classification) for toxic comments classification and improved the prediction accuracy to a new level.

1.3 Dataset

Our dataset comes from the Kaggle competition ***Jigsaw Unintended Bias in Toxicity Classification***. At the end of 2017, the Civil Comments platform shut down and chose to make their ~2m public comments from their platform available in a lasting open archive so that researchers could understand and improve civility in online conversations for years to come.

Each data point contains a piece of comment and a label. Each label is a fractional value and represents the toxic level of this comment. The competition organizer asked annotators to rate each comment based on many different rules and aggregated them to get the value for the labels. In this project we will only use the train dataset with label value >=7 and =0.

1.4 Our Goal

The majority of previous work on toxic comments classifier are deep learning-based models like RNN, LSTM, and CNN. Deep learning-based models are hard to interpret, i.e. like a black box, and require huge dataset to get fair results. We wish to not only build a toxic comments classifier, but also be able to identify some **high-level language features** for toxic comments. Also, toxic comments are not necessarily totally toxic and may contain non-toxic information.

2. Toxic words

In the previous parts, the current situation has been uncovered. That is, generally, toxic comments classification gets more attention than toxic words classification, since it is always thought that once the comment contains toxic content, the whole comment will be useless. However, in the reality, many comments can contain useful information like suggestions which lead to the natural process which is finding and deleting these toxic words and keeping the useful suggestions in the comments.

Based on previous toxic comments detecting methods, like CNN, LSTM, Naïve Bayes and so on. The modified Naïve Bayes has been finally selected as the method for detecting the toxic words of the comments in this project, since it is easy to understand and suitable for this problem.

2.1 Naïve Bayes

Naive Bayes is a probabilistic classifier, meaning that for a document d (feature f), out of all classes c∈C the classifier returns the class $\hat{c}$ which has the maximum posterior probability given the document (feature) as shown below:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c)P(c)}{P(d)} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{f \in F} P(f|c)$$

To apply the naive Bayes classifier to text, word positions need to be considered, by simply walking an index through every word position in the document:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{i \in position} P(wi|c)$$

However, the theory of Naïve Bayes is actually the foundation of the method for detecting toxic words and two different approaches have been generated since different modification of the Naïve Bayes.

2.1.1 First approach

The first approach is generally a comments-level modified Naïve Bayes classifier. Since it basically reuse the theory and codes of Naïve Bayes. That is the form of the data is still one label belonging to one whole comment. The difference is that the model here will not perform the forecast step but extract the informative tokens from the data.

The specific steps are shown below:

Step 1: The corpus consists of Reuters' tokens from NLTK will be regarded as baseline of clean tokens. Then take the difference between tokens of train set and clean corpus to get potential toxic tokens.

Step 2: Consider each comment as one unit, apply Naïve Bayes to potential toxic tokens in the train and output the informative features(tokens) with the following rules:

$P(1)P(wi=True|1) > P(0)P(wi=True|0)$ or $P(0)P(wi=False|0) > P(1)P(wi=False|1)$

Step 3: Finally, regard these informative tokens as toxic words detected from this approach.

2.1.2 Second approach

The second approach is generally a words-level modified Naïve Bayes classifier. Here the words from comments with label 1 will be grouped in the toxic group and the words from comments with label 0 will be grouped in the clean group. Then the probabilities in toxic and clean group of every word from toxic group will be compared to find the toxic words.

The specific steps are shown below:

Step 1: Split the train set into two groups: one contains comments with label as 0 and another one contains comments with label 1;

Step 2: Tokenize these two groups' comments to obtain clean tokens and possible toxic tokens;

Step 3: Calculate the ratio of the probability of each token from possible toxic tokens showing in the toxic group and clean group:

$$Ratio = [P(1)P(wi|1)]/[P(0)P(wi|0)]$$

Step 4: Set the threshold based on the ratios, e.g. if Ratio > 1, that is the threshold being 1 and then words with ratio values greater than 1 will be regarded as toxic words.

2.2 Dataset

The dataset for detecting toxic words is the first 500,000 comments of the whole dataset. The train set consists of the comments with the value of label equaling to 0 and the value of label not less than 0.7 in the first 500,000 comments. The in the train set the comments with label value not less than 0.7 will be recoded as 1. The validation set is 500 comments from 1,000,000th to 1,006,030th whole dataset with 250 comments with label 0 and 250 comments with label value not less than 0.7 and being recoded as 1. The test set is 500 comments from 1,500,000th to 1,506,840th whole dataset with 250 comments with label 0 and 250 comments with label value not less than 0.7 and being recoded as 1. Then toxic words for validation set and test set, will be annotated on the dataturks website.

2.3 Application

2.3.1 The first approach

Since it will take about 3 days to run Naïve Bayes on the first 500,000 comments, the method will be adjusted, that is running the model on every 50,000 comments instead. Also, in each 50,000 comments, the whole comments with label 1 and the limited number of comments with label 0 will be used, with the limited number equaling to the number of whole comments. Then 10 groups of possible toxic words based on the first approach for the first 500,000 comments and the detail is shown as below

Table1 Toxic words for 10 groups

Groups	*0-50,000*	*50,001-100,000*	*100,001-150,000*	*150,001-200,000*	*200,001-250,000*
Possible T-words	**671**	**683**	**598**	**667**	**700**
Groups	*250,001-300,000*	*300,001-350,000*	*350,001-400,000*	*400,001-450,000*	*450,001-500,000*
Possible T-words	**671**	**683**	**598**	**667**	**700**

The table 1 indicates that there are 10 groups of possible toxic words with each group about 650 words. Then the real toxic words of the whole train set will be selected by the frequency the word occurs in these all ten possible toxic-word groups. That is, each word from these 10 groups will have its occurring frequency in these all 10 groups with the smallest frequency as 1 and the largest frequency as 10. Then set boundaries, following the rule that words' frequency greater than n (n = 0, 1, 2, …, 9), to compare the precision and recall performance for different boundaries based on the validation set. The comparison results are shown as below:

Table2 Comparison of different boundaries

Frequency Boundary	*>0*	*>1*	*>2*	*>3*	*>4*
Precision	0.3673	0.4273	0.4805	0.5131	0.5493
Recall	0.7210	0.7111	0.7012	0.6765	0.6741
Boundary	*>5*	*>6*	*>7*	*>8*	*>9*
Precision	0.5906	0.6329	0.7251	0.8500	**0.9071**
Recall	0.6519	0.6469	0.6123	0.5876	0.5062

The table 2 shows that when boundary equals to 9, that is, when the tokens showing in all 10 groups, the highest precision value will be obtained with a reasonable recall value as 0.5062. Therefore, the tokens showing in all 10 groups will be regarded as the real toxic words detected by the first approach.

2.3.2 The second approach

The processing time of this approach is less than the first one. However, the validation set will be needed to find the best threshold of ratios based on the previous method introduction. Intuitively, the threshold should be 1, however, it does not perform well. The adjustment of the threshold will be needed. The adjustment result based on the validation set is shown as below:

Table3 Precision and recall of different thresholds

Ratios	*0.9*	*1*	*1.1*	*1.2*	*1.3*	*1.4*
Precision	0.5472	0.5271	0.5373	0.9522	**0.9557**	0.9552
Recall	*0.7580*	*0.6963*	*0.6913*	*0.6395*	***0.6395***	*0.6320*

The table 3 shows that when the threshold equals to 1.3, that is, when the ratio value is greater than 1.3, the highest precision value will be obtained with a reasonable recall value as 0.6395. Therefore, the tokens with the ratio of the probability of each token from possible toxic tokens showing in the toxic group and clean group greater than 1.3 will be regarded as the real toxic words detected by the second approach.

2.4 Evaluation

These two approaches will be compared on the same test set obtained in the section 4.2 to find the better approach. Typically, there are 359 annotated toxic words in the test set. And the comparison is shown as below:

Table4 Precision and recall of two approaches

Approaches	*First Approach*	*Second Approach*
Precision	0.9095	**0.9149**
Recall	*0.5877*	***0.5989***
Toxic words	*239*	*246*

The table 4 indicates that the second approach obtains the higher precision as 0.9149 and the higher recall as 0.5989. Therefore, the second approach will be the final method of detecting toxic words for the whole data.

2.5 Examples of detected toxic words

After applying the second approach to the whole dataset, there are about 670 toxic words has been detected. Some of the toxic words are shown as below:

Table5 Examples of detected toxic words

as*hole	*hit	a*s	ass	asshole	b*tch	bitches	buffoon	buffoonish
bullshit	buttholes	clown	crap	cunt	damn	dickhead	dumb	f**king
f^ck	f-ck	fck	fools	halfwit	hypocrite	whores	ignorance	imbecility
jerks	loser	stupid	idiot	moron	scumbag	sucker	traitor	…

3. Toxic Phrases

In many cases, the whole part of toxicity in a toxic comment is not just a single word, but a widely used syntactic structure/phrase that may contain toxic words. Thus, we hope to find out two different kinds of structure. One is toxic word related phrases and the other one is nontoxic based phrases.

3.1 Toxic words related phrases

After building up our own toxic word corpus, we would like to find out how these words been commonly used in phrases in the next step.

The main idea comes from the coding work in the last lecture, which includes

- Filter out all sentences that contain toxic words in the corpus we build.
- Pick out all trigrams (two previous tokens + itself / itself + two subsequent tokens) but exclude punctuations and spaces.

- Rank these trigrams with their frequencies.

For all comments with a target value ≥ 0.7, we find out 1139 trigrams with frequency ≥10 in total, and here are some of them on the top of ranking.

Table 6. Toxic words related phrases

Phrase	Frequency	Phrase	Frequency
('be', 'an', 'idiot')	844	('be', 'a', 'jerk')	151
('be', 'a', 'liar')	538	('what', 'a', 'stupid')	148
('-pron-', 'be', 'stupid')	315	('what', 'an', 'idiot')	125
('be', 'a', 'fool')	301	('fool', 'of', '-pron-')	121
('be', 'too', 'stupid')	220	('piece', 'of', 'shit')	117
('a', 'pathological', 'liar')	215	('idiot', 'like', '-pron-')	107
('be', 'a', 'moron')	212	('be', 'a', 'troll')	98
('dumb', 'and', 'dumber')	162	('sexual', 'predator', 'enabler')	95
('be', 'a', 'hypocrite')	155	('be', 'a', 'traitor')	93
('be', 'so', 'stupid')	154	('full', 'of', 'crap')	77

From the table above, we could summarize two major properties:
1. The most common toxic-words-related structures are:
'be a/an/the toxic_word', '-pron- be toxic_word', 'what a/an/the toxic_word', 'toxic_word and toxic_word', 'kind of / bunch of toxic_word', 'like a/an toxic_word'.
2. Most frequently used toxic words in these phrases are:
'idiot', 'liar', 'stupid', 'moron', 'hypocrite', 'ignorant', 'dumb',' pathological', 'loser', 'crap', 'fool', 'jerk', etc.

3.2 Nontoxic words related phrases

All the above phrases make people feel being offended by their toxic words, which may not be strongly convincing results as we hope. In this case, we also would like to look up all the toxic comments for some nontoxic based phrases and we guess they would be slangs,
idiomatic structures or some political allegorical phrases.

I remove all the toxic words, stop words and punctuations away from the raw comments to check all the trigrams on the top 500 frequency list and then pick out my desired results posted in the table below.

Table 7. Nontoxic words related phrases

Phrase	Frequency	Phrase	Frequency
right wing nuts	29	hook line and sinker	12
people voted Trump	25	fear engendered community	11
typical left wing	22	liberalism mental disorder	10
left wing sheep	21	worthless stain America	10
not care less	21	turn a blind eye	10
not handle truth	17	come back to bite	10

brain washed sheep	15	play the race card	9
poorly educated white	14	ad hominem attack	9

As we can see from this table, many of these nontoxic based phrases are political field terms due to the major topic from our data sources.

In summary, with the extraction of these toxic phrases, we can use them to do the comment cleaning by removing these kind of toxicity components from the toxic comments and see what kind of useful information we could obtain after that.

4. Information Refinement

Looking at the toxic comments, we found that many of the comments contain information besides curse. Instead of toxicity classification, we wanted to utilize the toxic word corpus we built using Naïve Bayes method to extract useful information from the toxic comments.

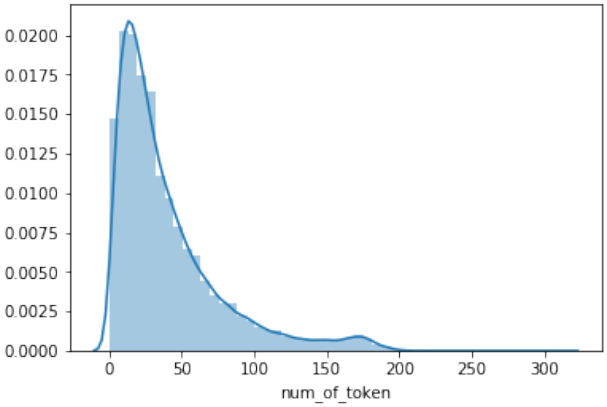
4.1 Filter Out Useless Comments

We wanted to filter out useless comments and sentences by checking the number of tokens and toxic words in each comment and sentence. There are in total 33872 comments used during this process and all of them contains at least one toxic word from the toxic word list.

4.1.1 Short Comments

After tokenization, we removed symbols and obtained the number of tokens in each comment. There were some short comments in the dataset, comments with less than 8 tokens. Looking at these short comments, since all these comments were toxic, and contains at least one toxic word from the toxic word corpus, as a result, most comments with less than 8 tokens contain only curse. Thus, we filtered out these short comments (3734 comments are removed).

Figuer1-Distrbution of Number of Tokens in Each Comment



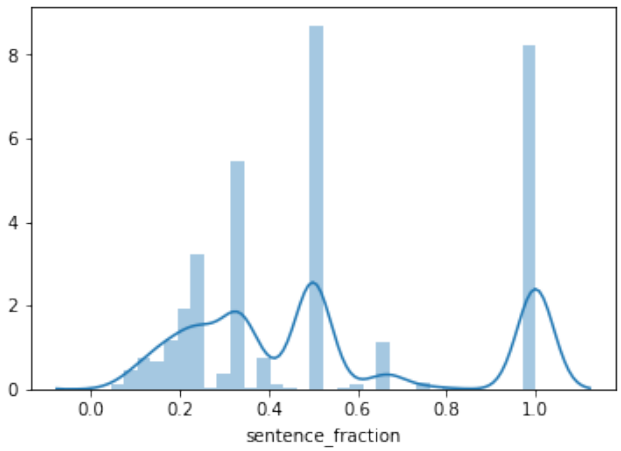
Here are some samples from the shorts comments we filtered out:

- You're an idiot.
- The referee is shit!
- You are a fucking cunt, lady.

4.1.2 Multi-Sentence Comments

There were many multi-sentence comments that had toxic words in each sentence of the comment. In this case, we filtered out some comments according to the fraction of number of sentence that contained curse, i.e. $h = \frac{number\ of\ sentence\ that\ contains\ curse}{total\ number\ of\ sentence\ in\ the\ comment}$. In other words, with higher h, there is less information besides curse and the bar we set here for h is 0.66. (1370 comments are removed).

Figuer2-Distrbution of h



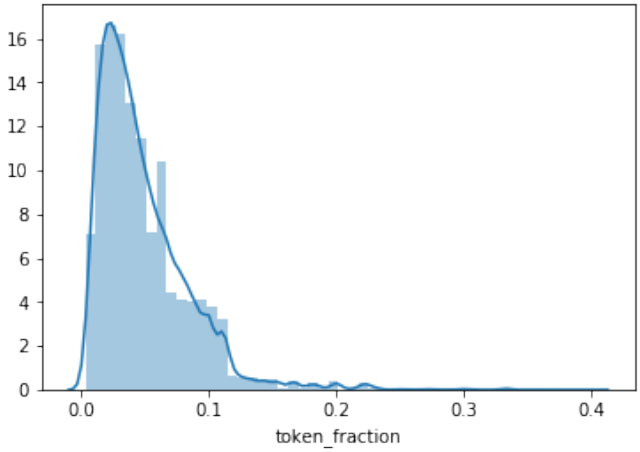
Here are some samples from the multi-sentence comments we filtered out:

- Wondering if you could be a bigger jerk. Bigger idiot? Bigger loser. Naaaahhhh
- You are an absurd and an ignorant moron. Go the F away you old dope. Hope you finally croak soon and your old ignorant af ways. TROLL!
- Moron walks on tracks. Moron killed by train. Moron's family sues railroad. A herd of morons sit on the jury and hand over bushels of dollars.

4.1.3 Comments with Many Toxic Words

After previous steps of filtration, there were still many comments contain mostly toxic contents. Thus, there was need for addition filter. In this step, we looked at the fraction of toxic words in the comments, i.e. $g = \frac{number\ of\ toxic\ words\ in\ the\ comment}{total\ number\ of\ tokens\ in\ the\ comment}$. We wanted to filter out comments with high g value. Therefore, comments with g higher than 0.09 were removed. (3605 comments are removed).

Figuer3-Distrbution of h



Here are some samples from the comments with many toxic words we filtered out:

- I have been saying this since that stupid fucking show came out. Fuck you Portlandia.
- Happy the moron was thrown off. He was probably liquored up as most of these immature morons are.
- Balto, you are an idiot. But that goes without saying. Just look who you support and what you say. Moron.

4.1.4 Short Sentence in a Comment

Similar to 6.11, in this step we filtered out the short sentences in a comment. Thus, sentences with less than 8 tokens, also containing toxic word in the sentence, were removed.
Here are some samples of short sentences we removed from the comments:

Table 8. Removed Sentence Samples

Comment	Short Sentence Removed
This bitch is nuts. Who would read a book by a woman.	This bitch is nuts.
Well shit, they drafted a guide. We should all be good now, whew aht a relief...	Well shit, they drafted a guide.
You're an idiot! Period! As a 53 year old woman, I love Bruce's music and him and his music is still relevant to me today as it was in 1980. Thanks for putting our guy in the media again. Idiot!!	You're an idiot Idiot!
Wyatt is a real jerk. I guess that's why he doesn't use his real name.	Wyatt is a real jerk.
WTF. Quit right now and focus on the budget. What a freaking circus. Our legislators act like they have time to waste.	WTF.
What crap. I am talking about permanent oil jobs lost, you have NO clue	What crap.

4.2 Extract Useful Information from Comments

4.2.1 Find Verbs for Filtered Comments

We assumed that we had the filtered comments that contain useful information. The most frequent verbs used in these comments might be verbs that are commonly used for suggestions or expressing opinions. Thus, we made a list from the filtered comments containing only the verbs. We also found a list of verbs commonly used when expressing opinions and suggestions from web searching. Then we obtained the intersection of the two lists as the final verb list.
Here is a sample of words from the verb list:

Table 8. Verb List Sampe

consider	accept	point	see	disagree	agree	denote
mean	feel	suggest	propose	recommend	advise	indicate

4.2.2 Select Sentence from Each Comment

Utilizing the verb list, we selected the sentence with verb from the verb list that does not contain toxic word. In addition, the sentence cannot be the shortest one among all the sentences from the comment. There were 1479 results generated.

Here is a list of samples of the selected sentence from the comments:

Table 9. Information Extraction Samples

Original Comment	Toxic Words	Selected Sentence
This is either dumb or stupid. The issue most folks have is not with the size of the PFD, but the fact that we are giving up any of our PFD while we throw money at wealthy oil companies in the form of tax deductions. If the Senate will pass the Wilson-Seaton oil tax bill, am sure the house will get behind the senate PFD bill. It is called compromise.	Dumb, stupid	The issue most folks have is not with the size of the PFD, but the fact that we are giving up any of our PFD while we throw money at wealthy oil companies in the form of tax deductions. If the Senate will pass the Wilson-Seaton oil tax bill, am sure the house will get behind the senate PFD bill.
Just imagine, a lying, crooked, corrupt, beech on a broom as the next President. And to think it's all possible through liberal loons voting for her. Now isn't that special? Stop censoring me you assholes!	assholes	Just imagine, a lying, crooked, corrupt, beech on a broom as the next President. And to think it's all possible through liberal loons voting for her.
So you write an article dismissing a product without (and I say this screaming at you, my computer, etc.) even trying it? Seriously go fuck yourself!	fuck	So you write an article dismissing a product without (and I say this screaming at you, my computer, etc.)
if tony really sent "mean" messages about her body & family, why would he want to continue that friendship? tony sounds like a jerk.	jerk	if tony really sent "mean" messages about her body & family, why would he want to continue that friendship?

5. Conclusion

In this project we made toxic word and phrase dictionaries and try to filter comments and extract useful information from the huge dataset of online comments. Utilizing the toxic phrases, we can potentially remove the toxic part of the comments which can serve as a cleaning tool for online environment without eliminating too much content. Using method from information refinement, we are able to find meaningful results in large dataset more efficiently which can be used to help content creators identify potential improvement faster.

Reference

[1] Spiros V. Georgakopoulos, Sotiris K. Tasoulis, Aristidis G. Vrahatis, and Vassilis P. Plagianakos. 2018. Convolutional Neural Networks for Toxic Comment Classification.
[2] Kevin Khieu and Neha Narwal, Detecting and Classifying Toxic Comment.
[3] Betty van Aken, Julian Risch, Ralf Krestel, and Alexander Lo¨ser. 2018. Challenges for Toxic Comment Classification: An In-Depth Error Analysis.
[4] James H. Martin and James H. Martin. 2018. Speech and Language Processing.