Reddit Auto-Moderation by Evaluating Community Opinion

Ayush Baid, Ankur Bhardwaj, Tarun Pasumarthi

1 ABSTRACT

Platforms like Reddit have become a host of contentious political arguments and sometimes go extreme with radical opinions/threats. It is a big challenge for companies to moderate discussions and keep forums free from hate-mongering and divisive language. Our objective is to create a system which auto-completes sentences related to any subject (like "Trump...", "Jews...", etc.) based on the overall subreddit's opinion such that it becomes easier for moderators to identify hate speech and toxic content across vast platforms.

We start with 100k comments each from the 14 subreddits and trained language models for them independently. We then fed 350 manually curated sample phrases, related to political subjects and issues. Initially, we used a simple fixed-window and word embedding based model, which had limitations with respect to the context window and model complexity. After observing poor results, we tried using RNN based architectures. This baseline generated syntactically correct sentences for most of the phrases, and in some cases generated the correct positive/negative opinion for the subreddit. However, we observed lack of contextual understanding in our model due to data-size and architecture limitations, which we aim to solve in our final approach.

2 DATASET DESCRIPTION

To create our dataset, we wanted to aggregate the comments of multiple popular and controversial political subreddits. We tried to mantain the balance in the dataset and the 14 subreddits we chose for this project comes from a spectrum of topics and ideologies, as described in table 1.

Left	Right	Mixed	Apolitical
r/democrat	r/Conservative	r/news	r/aww
r/ElizabethWarren	r/libertarian	r/politics	r/soccer
r/Impeach_Trump	r/The_Donald		r/YangforPresidentHQ
r/OurPresident			
r/SandersForPresident			
r/The_Mueller			

Table 1: Subreddits used: categorized by their political leaning. We use the name and description of the subreddit to categorize them

2.1 Source

We got our data from PushShift, which is a Reddit crawler that saves copies of objects (threads and comments) and their meta information. Our primary date range was to fetch 20,000 comments each month from September 2019 to December 2019 to obtain a total of 100,000 comments. This was done to ensure that we have comments from a variety of threads from different time periods. We created numpy arrays for each subreddit and the query month.

2.2 Data Pre-processing Steps and Explanation

We followed the following pre-processing steps:

- Break comments into sentences using the Punkt tokenizer
- Remove the user and subreddit internal links, common on Reddit
- Remove characters which are not English alphabets
- Convert to lower case
- (Optional) Remove stop words
- (Optional) Add <eos> token (end of sentence token)

We are operating on a limited resource budget and hence we decided to limit the number of comments we use in this project. We started with 50,000 comments and then increased it to 100,000 as we needed more data for our models.

Subreddit	#Sentences	Vocab-Size	Avg tokens/sent.
r/aww	182,296	43,068	8.83
r/Conservative	305,443	54,651	13.92
r/democrat	1,857	4,246	14.33
r/ElizabethWarren	255,458	42,049	16.15
r/Impeach_Trump	95,563	31,274	12.97
r/OurPresident	189,939	36,070	13.08
r/libertarian	359,008	55,875	14.62
r/news	268,829	52,733	14.07
r/politics	283,121	51,272	13.84
r/SandersForPresident	315,289	50,832	13.95
r/The_Donald	269,942	50,958	12.89
r/The_Mueller	320,194	50,337	12.39
r/soccer	207,777	47,368	12.07
r/YangForPresidentHQ	360,995	50,137	14.99

Table 2: Count-based stats on raw data computed in Pandas

2.3 Raw Data Statistics

We used pandas to run some basic count statistics to generate 3 stats on the dataset. The results are in table 2. Also, we analysed most frequent words as well as most important tf-idf tokens for each subreddit. Result for one of the topics are in Figure 2.

2.4 Data Analysis

We used multiple techniques to perform Exploratory Data Analysis on our dataset in order to drive intution and formulate testable hypothesis. As we used sentences from 100k comments for training our baseline models it was important to check if we have similar number of sentences for each corpus. Also, for training the LSTM baseline model, we have taken back propagation window size of 10 i.e. the model learns the weight parameter using hidden state context from last 10 words and hence we made sure that the average number of tokens in each training imput in greater than 10 as can

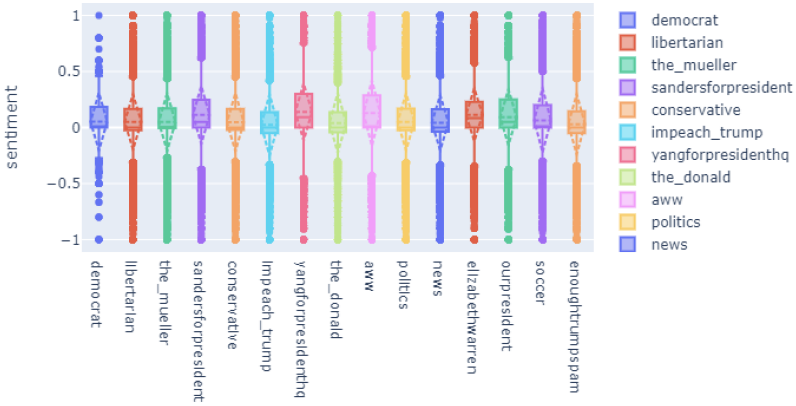

Figure 1: Sentiment Analysis of Subreddits using Textblob

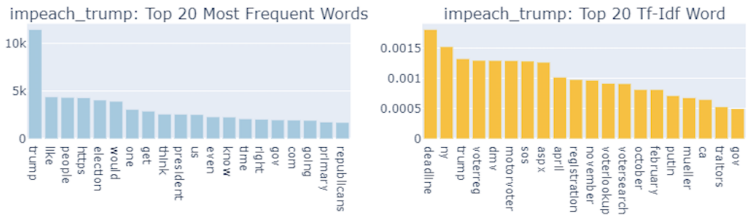

Figure 2: Most common words vs top tf-idf terms for r/Impeach_Trump Subreddit

be seen in table 2. Other important feature was the vocabulary size which showed normal distribution centered around mean of 43000 with a lot of overlap among political and non-political topics which helped in generating contrasting outputs using similar tokens.

Using Textblob sentiment analysis, we observed that except for few subreddit datasets all have neutral sentiment with average polarity close to 0. While other subreddit topics like *aww*, *YoungforPresidentHQ* showed positive average sentiment score and *Impeach_Trump*, *The_Donald* showed slight negative sentiment. In order to understand the the distribution of common words, we found most frequent words used in all subreddits - for instance *Impeach_Trump* has *trump*, *elections* etc. as common words. Moreover, the tf-idf analysis of the corpus helped us find words responsible for the negative sentiment in few subreddit- for ex. *deadline*, *traitors* were among the most relevant tokens in *Impeach_Trump* subreddit.

3 EXPERIMENTAL SETUP

Data split: We split the comments obtained from each month and each subreddit into a 70-30 split to create train and validation sets.

We designed 350 phrases manually (link) which will be fed to the language models to produce completed outputs. These outputs will help a human/machine moderator to take decision and provide the insight behind why a subreddit is being classified as toxic/harmful.

Training problem and metrics: For the training, we posed the problem as next word classification and hence used cross-entropy loss across training and validation set for this milestone.

As our problem to feed manual phrases to the language model and obtain a word/sentence output is a new one, there is no inherent ground truth. Conceptually, we are training our language models to complete the sentences for a subreddit and repurpose the model to complete brand new phrases. We hope that the completed sentences will be consistent with the opinion of the subreddit. This is a novel problem formulation and hence we cannot use the common metrics of classification tasks like AUC-curves and accuracy.

To tackle this difficulty, we came up with a new metric where we manually review the completed phrase from the models and tagged it as *useful* or not *useful*. We then get the ratio of useful completions from a model trained for a particular subreddit and that serves as the metric for our project. We use personal experience from using Reddit and visiting the 14 subreddits to judge if a particular phrase completion is consistent with the general discussions and ideology of the sub.

Here are a couple of examples to illustrate our metric:

- For subreddit the_donald, a phrase "Trump administration has something" is categorized as *not useful*
- For subreddit politics, a phrase "Trump should be impeached" is categorized as *useful*
- For subreddit SandersForPresident, a phrase "Sanders should be impeached" is categorized as *not useful* as it is not a real possibility and is also not something which is discussed in a pro-Sanders subreddit.

System Configuration: We ran some aspects of the work like dataset scraping and simple data analytics described till this point on Google Colab, which provides 16GB of RAM for a session. The machine learning models were trained on the cluster made available to Ayush by his advisor. The specs of those machines are variable, but they generally have GPUs with 12-15 GB memory, Intel Xeon CPUs with around 50 cores, and upwards of 350 GB of RAM.

4 BASELINES

We tried two different baseline language models based on neural networks and we will describe the details and the results in this section. Before diving into neural models, we tried building a very simple language model based on n-grams where we computed probability of next word given bi-gram words in the whole corpus i.e. n-grams model with n=2. This model had two main problem while choosing larger values of n - firstly sparsity of exact same words as in the query lead to bad results and secondly the frequency of all n-grams in the corpus are needed to be stored which makes it computationally and storage expensive. Thus, to solve these issues we looked into neural based language models.

4.1 Baseline Description

We use two simple and commonly used language models for this project. The first one is the fixed-window based language model and the second is an RNN-based language model.

Fixed-Window Based Language Model. This language model was proposed by Bengio et al. [1] in 2003. It has a very simple architecture composed of just fully connected layers. The model has a fixed context window which is embedded into a feature space using the word embeddings layer. All the embedding vectors resulting from the context are concatenated into a single big vector and the fully connected layers of the model have to process this big vector to predict the next word. An example with single layer model is provided in figure 3.

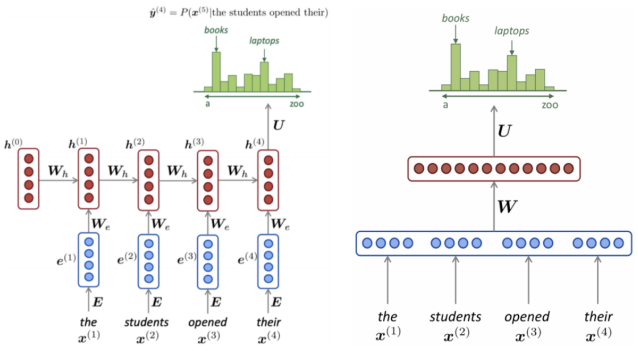

Figure 3: Left: LSTM and Right: Fixed-window based language model

Credits: Stanford 224n class

This is a very limited model, as it does not have sufficient complexity to learn about the meaning of different words, the semantic and contextual understanding of the language. However, it also offers an advantage that it does not need huge volume of training data.

We will use an example to convey our intuition: there are many discussions about President Trump on the political subreddits. There will be many sentences of the form "Trump is a <adjective>". To handle to limited complexity of the model and the hard limit of a context window size, we can remove the stopwords from the sentence and obtained lots of datapoints in the training set which are of the form "Trump <adjective>". This simple model will be able to pick up what words are generally used next to "Trump". During evaluation of phrases for the auto-moderation task, if we feed the word "Trump" to the model, it might give us the commonly used adjective(s) with President Trump.

As we have a limited training data compared to deep-learning standards, we limit the context window size to 2. We also use a pretrained word2vec model [2], a 300 dimension embedding space learnt on google news. The pretrained embeddings will provide us a lot of similarity associations between words instead of training it from scratch.

LSTM based language model. Language has an inherent temporal dimension. As a result, the introduction of RNNs [4] has been highly beneficial to the NLP community. RNNs can be used to complete sentences but they are not restricted by the fixed context size like the previous baseline.

We follow the LSTM based recurrent language model [3] and modify it to better suit our needs. The 'smaller' version in their paper has 2 LSTM layers and a hidden space dimension of size 200. They also propose to tie the weights of the encoder and decoder in the network. We replicate this model architecture.

For training, we use the Stochastic Gradient Descent optimizer with gradient clipping. We train the model for each subreddit for 40 epochs and using the back-prop through time (BPTT) value to 10. This choice was made after considering the average number of tokens presented in the data analysis section.

We experimented with layers of LSTM (1 v 2), BPTT values (5 v 10 v 35), and use of pretrained word2vec embeddings. We chose the final values of these paramters after qualitatively analysing the completed phrases. The code for this part was obtained from [1] and we used it with some minor modifictions.

Suitablity of the baselines. Our objective is to create a scalable system to generate necessary information for human moderators and enable deletion/flagging of toxic subreddits. In order to bridge this gap and provide a small amount of data to review instead of millions of comments per day, we auto-complete a fixed set of phrases to find out the general behaviour/opinion about subjects in each subreddit.

Our project is a new application of sentence completion models and hence we start off with simple learning architectures which can be trained on a limited dataset and computational resources. These methods will serve as a good baselines when we move the the cutting-edge in NLP which are harder to train.

[1] https://github.com/pytorch/examples/tree/master/word_language_model

Subreddit	Input	Output
r/politics	trump	supporters
	jews	people
	sanders	warren
	centrists	vote
r/The_Donald	trump	train
	samders	www
	jews	people
	bloomberg	russia
r/Our_President	trump	sanders
	sanders	supporters
	jews	people
	obama	clinton
	border wall	street
	bloomberg	secure

Table 3: Sample phrase completions from fixed-window language model

4.2 Baseline Results

We demonstrate some sample phrase completions from the two baseline models in table 3 and 4. The quantitative metric of usability is presented in 5. We evaluate the usability on just three subreddits for now because of the manual cost of tagging the results.

4.3 Result Discussion

The results from the fixed window model were unsatisfactory in terms of syntactic and semantic rules of the generated english text. For ex. trained on r/politics, it predicts the words 'trump' and 'people' for most of the input prompts and thus is not useful at all for our intended use case.

The RNN based model, these models perform very well in understanding the dependency in the sentence structure. However, it struggles to keep the understanding of the subject till the end and makes conflicting predictions for the same phrase. It also repeats the same predictions for different subjects. The examples show that the model identifies *Trump*, *Bernie* in the input query as Person and then predicts next words as person's quality like *liar*, *honest*, *criminal* with proper prepositions in place; however the capability to predict these qualities in specific person's context is still wanting. The results in table 4 are also not usable: r/The_Donald is a pro-trump subreddit and our outputs are negative for trump. Similarly, r/Our_President will not have positive opinion of Trump.

5 NEXT STEPS

The results we obtained from the RNN Language Model baseline demonstrate that our approach has some potential and might be successful in automoderation applications. Moving from fixed-window to RNN based models provided the much needed increase in model complexity and encourages us to take the next step to convert this project into a usable product.

Going forward, it'll be of interest to look into models like BERT (Bidirectional Encoder Representations from Transformers) which focuses on bidirectional training across each input which helps in better understanding of the context in longer sentences. Also, we'll explore HuggingFace's implementation of GPT-2 Language Model which has been trained with over 100 million parameters and claims to work on the semantics of long generated text summaries quite well.

Subreddit	Input	Output
r/politics	trump is a	traitor
	trump is a	criminal
	biden should	not be the nominee
	sanders is	a socialist
	ubi	is a good thing
r/The_Donald	trump is a	cuck
	trump is a	fucking joke
	biden should	be the same
	sanders is	the best
	ubi	*null*
r/Our_President	trump is a	socialist
	trump is a	republican
	biden should	be the nominee
	sanders is a	socialist
	ubi	is a good thing
r/aww	trump is a	good boy
	trump is a	cat
	jews are	the best
	jews are	the worst
	sanders is	so cute

Table 4: Sample phrase completions from RNN language model

Subreddit	Fixed Window	RNN based
politics	2/149	45/344
The_Donald	1/145	37/342
Our_President	1/137	30/345

Table 5: Usability Metric for baselines

6 CONTRIBUTION

In the first phase of the project we chalked out three main tasks which included data exploration and scraping; building baseline model pipelines and finally training validating models across each subreddit. Each of us contributed in all three aspects as we divided the workflow in parallel to finish each task before moving onto the next one.

REFERENCES

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3, Feb (2003), 1137–1155.

[2] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[3] Ofir Press and Lior Wolf. 2016. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859* (2016).

[4] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature* 323, 6088 (1986), 533–536.