CS230

Detecting the Emotion Dynamics of Profanity Language In Social Media Through Time

Haozheng Du, WenXin Dong, Kezia Lopez
Department of Computer Science
Stanford University
bf3magic@stanford.edu, wxd@stanford.edu, keziakl@stanford.edu

Abstract

How has profanity's emotional context changed through time? In our study, we perform emotional analysis to tweets containing curse words and map out the changes in emotional use of curse words through time. We build a multi-class emotion classifier using XLNET to classify curse-word emotional context from archived tweets and present our findings of the trends from 2011 through 2019.

1 Introduction

In English, curse word context has changed drastically; for example, words like 'b*tch', 'n*gga', and 'f*g' have been reclaimed by certain groups to empower themselves while words like 'h*ll' and 'd*amn' deviate from their original religious context. With this in mind, we want to explore different emotional contexts of curse words in colloquial speech and how they differ through time?

The problem with sentiment and emotion classification using existing NLP models for curse-word embedded sentences is that these models tend to give a negative bias towards examples containing curse words. Also, existing emotion classification models do not contain the labels of our interest.

Therefore, we first build our own NLP model based on XLNET and then use it to predict the emotional context of archived tweets from 2011 to 2019. We input a preprocessed tweet containing one or more curse words; then we use our modified XLNET model to output an emotion class: Empowering, Upset, or Sarcastic.

2 Related work

Although there is a significant amount of work done in Twitter sentiment analysis and vulgar language detection models for automatic abuse flagging, the automated study of curse word use through time (compared to non-machine learning linguistic methods) is a relatively unexplored phenomenon, [1, 5]. Researchers have created several datasets of emotion and valence using Twitter [2, 6], but these data sets don't contain many curse words [3]. Some studies have been done on pure vulgar language tweets [4]; however, these studies focus on a binary approach: neg or pos [4], abusive or not [5], and depend on small data sets (8000 tweets). Additionally, these studies do not focus on emotion change over time[2,4].

Based on the gaps of research, we decided to to study the change of word context and usage through time. Combining ideas such as a mining of Archive.org's 9.4 billion Tweets Stream [7] and Tweet emotion analysis by fine-tuning BERT [8], we compared the best model performances from past similar studies using LSTMs and FastText [3] to pre-trained language models like XLNet [8], which substantially performed better with a similar data size [8].

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

3 Dataset and Features

3.1 Training Data Collection

One technique in NLP to collect emotion-labeled tweets is to search for tweets containing certain emotion hashtags and emoticons[9,11]. For example, in a study done by Alfina et al., hashtag-labeled datasets achieved 95% accuracy for sentiment analysis of political tweets[10].

To construct our dataset, we crowdsource tweets that contain the emotion tags and emoticons shown in Table 1 (see Appendix). Our primary querying sources are Twitter API and the Sentiment 140 Dataset[1], which contains 1.6 billion tweets. In addition to querying from Twitter, we also extracted examples containing curse words from the following datasets: Tweet Emotion Intensity Dataset[2]; a Crowdflower dataset[3]; and Sarcasm on Reddit Dataset[4]

3.2 Data Preprocessing

Inspired by techniques used in Twitter Emotion Detection done by Hasan et al.[11] and in the tweet text preprocessing experiment by Singh T. and Kumari M.[12], we cleaned (and re-cleaned after model experimentation) our raw tweets dataset the following way:

1. Lowercase every word
2. Replace links with the string "URL", htmls with the word "HTML", and user mentions like "@username" with the word "USERID".
3. Remove occurrence of three or more characters: replace "happyyyyy" with happy, ":))))" with ":)"
4. Remove ambiguous tweets that belong to more than one sentiment class that might confuse the model. For example, remove "This is sad :(# sarcasm" or ":o This is so sad >:("
5. Remove consecutive hashtags and strip off hashtags from the end only. For example, "I # feel very # happy today # motivated # energy # fun" becomes "I # feel very # happy today".
6. Remove tweets that are less than 6 words long as they do not provide noise in emotional ambiguity.
7. Removed emojis, emoticons, punctuation, and special characters including '# '.

After preprocessing, our current dataset consists of a total of 7400 crowd-sourced tweets across three different sentiment classes: Empowering, Upset, Sarcastic. Distribution among the three sentiment classes are shown in Figure 6 (see Appendix). Table 3 shows a few examples from our dataset after pre-processing. Figure 7 shows frequency of each curse word in our current dataset.

3.3 Historical Data Collection

We collected tweets from 2011 to 2019 from Archive Team: Twitter Stream Grab [5] . For years 2011 - 2016 we sampled tweets from a single month at random and for 2017 - 2019 from a single day at random because of download sizing limitations. Then, for each year's data, we randomly sampled around 20k tweets containing curse words. We then preprocessed the sampled historical tweets using the same procedure presented in section 2.2.

As shown in Figure 11, our sampled data has a stable categorical distribution of curse words through time, evidencing fair sampling. We categorized curse words into 8 categories, as lined out in Table 2, like, ableist = { "retarded", "spaz"}. We use this categorization in our later analysis of the evolution of emotional context of both all profanity classes but particular curse words categories, ex. Race, too.

4 Model

4.1 Pre-Analysis with the Amazon Comprehend Sentiment Analyzer Model

To gauge how well industry NLP models perform on this emotion classification task, we used the Amazon Comprehend model to predict the sentiment of our training and test sets. Although the model

[1] https://www.kaggle.com/kazanova/sentiment140

[2] http://saifmohammad.com/WebPages/EmotionIntensity-SharedTask.html, 3000 tweets with anger, fear, joy, and sad labels

[3] https://data.world/crowdflower/sentiment-analysis-in-text, 40000 examples across 13 different emotion labels

[4] https://www.kaggle.com/danofer/sarcasm?select=train-balanced-sarcasm.csv, 1.3 million labelled comments from Reddit labeled as sarcastic or non-sarcastic.

[5] https://archive.org/details/twitterstream

is officially binary (POSITIVE, NEGATIVE) with an additional category MIXED, we analyzed the results using a equation of Encouraging to Positive, Upset to Negative, analyzing Sarcastic tweets separately. We found that Comprehend performed very poorly, especially in words from the general and gender-sexuality categories ("shit", "fuck", gay", "bitch", "queer", etc.). Comprehend also labeled the vast majority of sarcastic tweets (85%) as NEGATIVE. The following is the breakdown of results:

Overall tweets classified incorrectly: 0.4449
Empowering Tweets Labeled Incorrectly as Negative: 0.857
Upset Tweets Labeled Incorrectly as Positive: 0.2458

4.2 Initial Exploration: BERT

To perform emotion analysis on the historical data, we first built and trained a BERT[13] based model on the training set. BERT is a state-of-the-art Transformer that learns contextual relations between words in text via an attention mechanism. The Transformer includes two main components, an encoder that reads the input test, and a decoder that makes prediction for the tasks. Unlike LSTM models which read the text input sequentially (or bi-directionally for bi-directional LSTM), we can consider BERT as non-directional, since BERT was trained by putting [MASK] on words on random locations in texts to learn their contextual relations.

Vanilla BERT inputs two sentences, with a [CLS] token placed at the beginning of the first sentence, and an [SEP] token between the two sentences. However for text sequence classification tasks, the input should consist only of [CLS] token, tokenized sequence data, and an [SEP] token. For our emotion classification task, we introduced a compact pre-trained BERT model: BERT-Base, Uncased[6] with 110M parameters. We tokenized input texts by mapping words to a word IDs using the vocabulary file from BERT. Then we padded the tokenized data to a maximum sequence length of 40 with [CLS] and [SEP] tokens inserted. To output the predicted emotion, we used the output from [CLS] token and added two dense layers with Dropout on top. Detailed model structure is shown in Figure 8. Note: the model outputs a vector of shape (None, 5) because we initially experimented on 5 classes.

4.3 Switching to XLNet

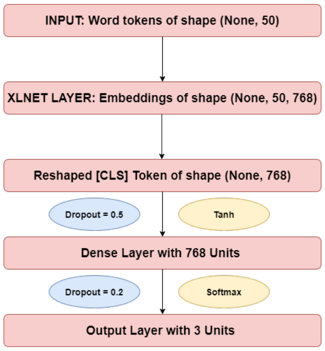

INPUT: Word tokens of shape (None, 50)
XLNET LAYER: Embeddings of shape (None, 50, 768)
Reshaped [CLS] Token of shape (None, 768)
Dropout = 0.5
Tanh
Dense Layer with 768 Units
Dropout = 0.2
Softmax
Output Layer with 3 Units

Figure 1: Model Structure

After experimenting with the BERT model, the best result we could get was a 0.71 prediction accuracy on the test data, not convincing enough to make predictions on the historical data. Therefore, we started exploring other options. XLNet is another state-of-the-art transformer model for NLP studies. The XLNet paper[14] addresses one substantial weakness of BERT: By putting [MASK] tokens on random word locations, BERT makes an assumption that the predicted tokens are independent of each other given the unmasked tokens, not accounting for high order and long range dependency in natural languages. XLNet incorporates the benefits of both autoregressive models (e.g Transformer-XL) and autoencoding models (e.g BERT) while ditching their weaknesses, by adding a Two-Stream Self-Attention mechanism to maximize the expected log likelihood of a text sequence. To explain the difference brought by Two-Stream Self-Attention, here is an example comparison between XLNet and BERT. Given the sentence "Stanford University is competitive", both XLNet and BERT select [Stanford, University] as prediction target. Now their cost function can be reduced to:

$$J_{\text{BERT}} = \log P(\text{Stanford}|\text{is competitive}) + \log P(\text{University}|\text{is competitive})$$
$$J_{\text{XLNet}} = \log P(\text{Stanford}|\text{is competitive}) + \log P(\text{University}|\text{Stanford, is competitive})$$

Structure of the input of XLNet is the same as that of BERT. We tokenized our tweet texts with a [CLS] and [SEP] token added and a sequence length of 50 and introduced the pre-trained XLNet model, XLNet-Base, Cased[7] as our first layer. Then we extracted the output from [CLS] token and

[6] https://github.com/google-research/bert/blob/master/README.md
[7] https://github.com/zihangdai/xlnet/blob/master/README.md

added two dense layers with Dropout like we did with the BERT model, to output the predicted sentiment. A simplified model diagram is shown in Figure 1, and a detailed model summary is shown in Figure 9.

We used tanh and softmax as our activation functions in the two dense layers, respectively. Our model outputs a vector with shape (None, 3) to match the number of unique labels. We then used Adam optimizer and selected Sparse Categorical Crossentropy as the loss function. Sparse Categorical Crossentropy Loss is a modified version of Categorical Crossentropy Loss (shown below) that does not require one-hot encoded prediction output to calculate the loss:

$$Loss = -\sum_{i=1}^{n} y_i * \log \hat{y_i}$$

4.4 Training and Fine-Tuning

Since our labeled tweets data was imbalanced (Figure 6), we implemented both over-sampling for tweets labeled "Upset" and down-sampling for tweets labeled "Empowering" and "Sarcastic", to balance over-fitting "Upset" tweets and having less data. To speed up the training process, we incorporated EarlyStopping monitored on validation accuracy as well as ReduceLROnPlateau. EarlyStopping removed excessive training epochs to prevent over-fitting and picks the best performing weights across the weights obtained from each epoch, while ReduceLROnPlateau helped us get out of saddle points.

For hyper-parameter tuning, we prioritized the number of hidden units in our dense layer, as well as the Dropout percentage. Because our relatively small training set could easily result in over-fitting, we found a smaller model performed better with a relatively high percentage of Dropout (0.5 for the first Dropout layer, and 0.2 for the second Dropout layer). For the same reason, we only implemented one dense layer of 768 units before the model output. We then spent a considerable amount of time on manually tuning the learning rate as well as parameters in EarlyStopping and ReduceLROnPlateau. Monitoring the learning curve (Figure 2), we settled down on an initial learning rate of $4 * 10^{-6}$, and set the number of epochs to 30. Full set of hyper-parameters can be found in Table 4.

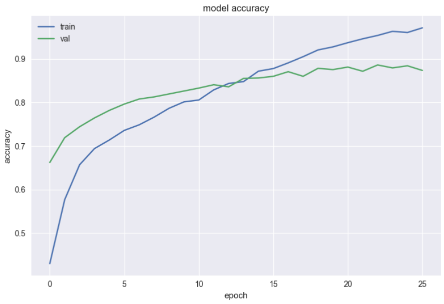
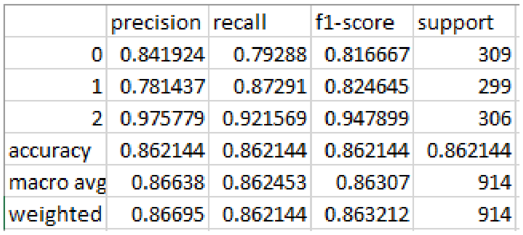


Figure 2: Learning Curve on Accuracy



	precision	recall	f1-score	support
0	0.841924	0.79288	0.816667	309
1	0.781437	0.87291	0.824645	299
2	0.975779	0.921569	0.947899	306
accuracy	0.862144	0.862144	0.862144	0.862144
macro avg	0.86638	0.862453	0.86307	914
weighted	0.86695	0.862144	0.863212	914

Figure 3: Classification Report

4.5 Performance Evaluation

After evaluting the model with our test set, we obtained the confusion matrix (Figure 10) and an overall test accuracy of 0.862. The confusion matrix shows our model performs very well in "Sarcastic" instances but drops accuracy in "Upset" and "Empowered", probably because we collected the majority of "Sarcastic" data from existing datasets of "Sarcastic" or "Not", yet individually collected a large portion of "Upset" and "Empowered" data, deeming it noisier. The F1-scores (full classification report in Figure 3) further proved our observation: we obtained F1-scores of 0.817, 0.825, and 0.948 for "Empowered", "Upset", and "Sarcastic", respectively. This was a big leap from the accuracy our BERT model achieved and Amazon Comprehend, so we hold our model is robust enough to make convincing predictions on the historical datasets.

5 Results, Discussion, and Future Work

We carried out prediction on the historical datasets using our XLNET model and we present our findings below. As shown in Figure 4, we see that from 2011 to 2019, empowering and upset emotional contexts of curse words strongly increase. We also see a mirroring downward trend in the use of sarcastic emotional context. This result suggests that curse words context have become more bi-polarized and less sarcastic.

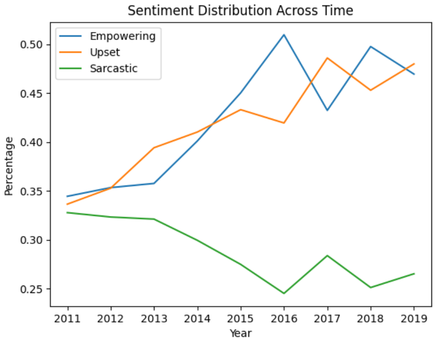


Figure 4: Profanity Language in Twitter: Emotional Context Trend Over Time

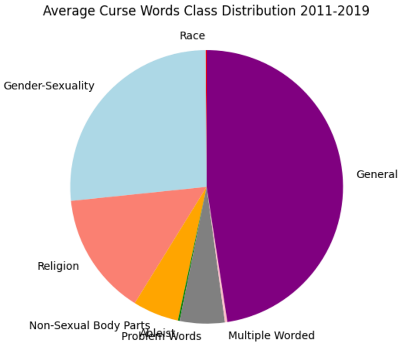


Figure 5: Curse Words Category Distribution Averaged across 2011-2019

We proceeded to identify the cause behind the trend. Was the trend induced by increasing use of certain curse words that are often used in empowering/upset context? Was it a general trend in all curse words categories or just some?

To answer the questions, we plotted the trends in emotional context for individual curse words categories, as shown in Figure 11 - Figure 19. We see a similar positive trend for Empowering and Upset in the 4 most dominating categories (dominating as in more commonly-used): General, Gender-Sexuality, Religion and Non-sexual body parts. For the 4 less dominating categories: Ablest, Problem Words, Multiple Worded, and Race, the trends are noisier, probably due to limited sample size.

Therefore, we speculate that the bi-polarizing trend has not much to do with particular curse words categories, rather a general trend independent of any particular curse words category.

However, our finding alone is not strong enough to conclude that there is a significant evolution in the emotional context of curse words. We will need to rule out other possibilities, such as:

1. The trend we identify applies to the bigger language context, not just profanity language. I.e. the curse-word emotional contexts did not evolve in any special way.

2. Our model predicts Empowering and Upset with less likelihood for more historical data for reasons unrelated to the actual sentiment context.

We will need to carry out future works to rule out the above possibilities. To rule out (1), we could carry out prediction on general historical tweets in additional to curse-word-specific historical tweets. To rule out (2) we would need to carry out error analysis.

6 Conclusion

To conclude, across 2011-2019, we see a strong positive trend for empowering and upset emotional context use of curse words and a negative trend in the use of the sarcastic emotional context for profanity language over time.

Further, we conclude that this trend is an overarching trend across most curse words. That is, this trend is not induced by any particular curse word category. However, we would need to carry out error analysis to rule out potentially irrelevant contributing factors, to further support our argument.

7 Contributions

As a team of three, we equally contributed to the brainstorming, planning, coding, and writing aspects of the project.

We collaboratively executed parallel tasks, and the following is a rough breakdown of what each person did.

Haozheng Du: Experimented data cleaning and balancing strategies. Implemented BERT and XLNet in our model. Trained, fine-tuned, and evaluated the model. Created visualizations of model performance.

WenXin Dong: Wrote Python scripts for crowdsourcing training data from Twitter API and other Tweets datasets. Experimented different preprocessing strategies. Worked on identifying trends in archived data and plotted graphs.

Kezia Lopez: implemented and analyzed the performance of the Amazon Comprehend Sentiment Classifier for other industry-model performance comparison and collected, processed, and cleaned archived data

References

[1] Razavi, Amir H., et al. "Offensive language detection using multi-level classification." Canadian Conference on Artificial Intelligence. Springer, Berlin, Heidelberg, 2010.

[2] Mohammad, Saif, and Svetlana Kiritchenko. "Understanding emotions: A dataset of tweets to study interactions between affect categories." Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). 2018.

[3] Founta, Antigoni Maria, et al. "A unified deep learning architecture for abuse detection." Proceedings of the 10th ACM Conference on Web Science. 2019.

[4] Cachola, Isabel, et al. "Expressively vulgar: The socio-dynamics of vulgarity and its effects on sentiment analysis in social media." Proceedings of the 27th International Conference on Computational Linguistics. 2018.

[5] Pamungkas, Endang Wahyu, Valerio Basile, and Viviana Patti. "Do you really want to hurt me? Predicting abusive swearing in social media." The 12th Language Resources and Evaluation Conference. European Language Resources Association, 2020.

[6] Mohammad, Saif M., and Felipe Bravo-Marquez. "WASSA-2017 shared task on emotion intensity." arXiv preprint arXiv:1708.03700 (2017).

[7] Tekumalla, Ramya, Javad Rafiei Asl, and Juan M. Banda. "Mining Archive. org's Twitter Stream Grab for Pharmacovigilance Research Gold." Proceedings of the International AAAI Conference on Web and Social Media. Vol. 14. 2020.

[8] González, J. Ángel, et al. "ELiRF-UPV at TASS 2020: TWilBERT for Sentiment Analysis and Emotion Detection in Spanish Tweets." Proceedings of TASS (2020).

[9] Koto, F., Adriani, M. (2015, December). HBE: Hashtag-based emotion lexicons for twitter sentiment analysis. In Proceedings of the 7th Forum for Information Retrieval Evaluation (pp. 31-34).

[10] Alfina, I., Sigmawaty, D., Nurhidayati, F., Hidayanto, A. N. (2017, February). Utilizing hashtags for sentiment analysis of tweets in the political domain. In Proceedings of the 9th International Conference on Machine Learning and Computing (pp. 43-47).

[11] Hasan, M., Rundensteiner, E., Agu, E. (2014). Emotex: Detecting emotions in twitter messages.

[12] Singh, T., Kumari, M. (2016). Role of text pre-processing in twitter sentiment analysis. Procedia Computer Science, 89(Supplement C), 549-554.

[13] Devlin, J., Chang, M., Lee, K., Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL-HLT.

[14] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q. (2020).XLNet: Generalized Autoregressive Pretraining for Language Understanding

Appendix

Github Repository: https://github.com/DHZBill/CurseWordsInContext

Sentiment	Hashtags	Emoticons
Empowering	#happy,#funny,#greatmood,#superhappy, #atlast,#ecstatic,#thankful, #feelinggood,#love,#loveyou,#joy, #yay,#blessed,#thrilled,#lol, #motivation,#positive,#fun #positivethinking,#excited,#exciting,	;),:))),=), :],:P,:-P, :D,;D,:>, :3,;-),:-D
Upset	#sad,#heartbroken,#leftout, #sadness,#depressed,#disappointment, #disappointed,#unhappy,#foreveralone, #pissed,#angry,#pissedoff,#furious, #hateyou,#annoying,#ugh,#anger,#fuming, #heated,#angrytweet,#aggressive,#godie, #pieceofshit,#irritated,#afraid,#mad, #petrified,#scared,#anxious,#worried, #frightened,#freakedout,#haunted	:(,:(((,=(((, =(,:-(,:^(, :’(,:-<,>:S, >:{,>:,x-@, :@,:-@,:-/, :/,:-o,:$, :-O,o_O,O_o, :-O,:O,:-o, :o,:-0,8-0, >:O,:-l,:-|
Sarcastic	#sarcasm	

Table 1: Hashtags and emoticons used to auto-label train data

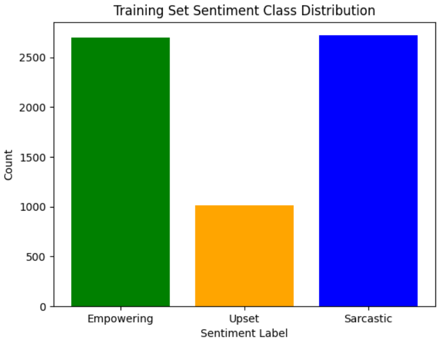


Figure 6: Training Data Class Distribution

Category	Curse Words
General	fuck,fu*k, f*ck,f**k,sh*t,shit,pissed,screw
Gender-Sexuality	dick,di*k, d*ck,cunt,pussy,pu**y,fag,queer,qu**r,boner, dong,slut,sl*t,dyke,pimp,whore,hoe,bitch,b*tch,bi*ch, cock,tramp,cum,schlong,spunk,skank,motherfucker,tit,gay, mothafucker,screw,blowjob
Religion	hell,damn
Non-Sexual Body Parts	ass, queaf,shart,urine,rimming,arse,shat,crap
Ableist	retard,spaz
Problem Words	tit,cum,hoe,chink,gay
Multi-Worded	son of a bitch,doggie style,fucked up
Race	nigger,n*gger, n*gg*r, chink,niglet, wetback

Table 2: Curse Words Categorization

Text	Sentiment
sometimes it's cool to be bitch to those deserving of it	Empowering
still cant log into my fucking snapchat	Upset
stop being pragmatic we are talking about constructing a utopia here man get out of here with that historical evidence and facts shit	Sarcastic
sunday nights are about chilling the fuck out before hell aka monday arrives again mother fucker hope you all have a great great night	Empowering
thanks everyone for allowing me to get a summary of your shitty feeds because mine alone wasnt enough	Sarcastic

Table 3: Training Set Examples After Preprocessing

Layer Name	Value
dropout	0.5
dense	768
dropout_1	0.2

Parameter	Value
learning_rate	4e-6
num_epochs	30
batch_size	32

Table 4: Hyper-parameters

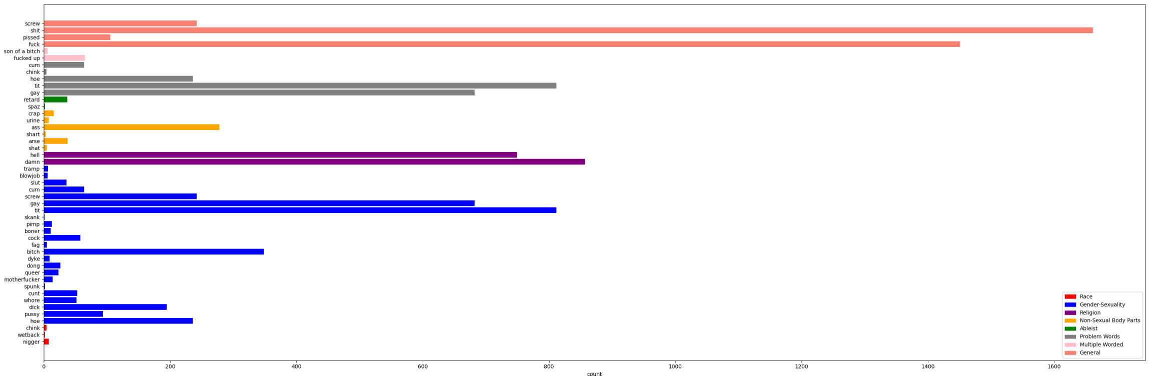
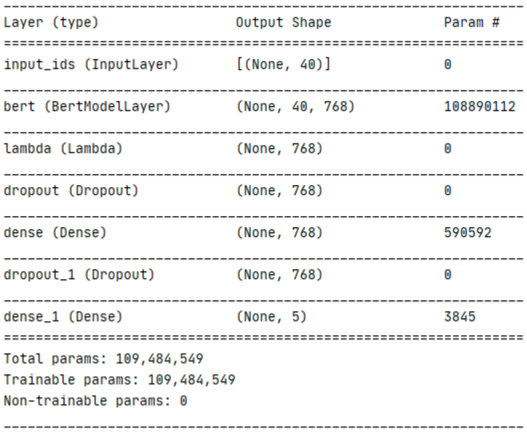
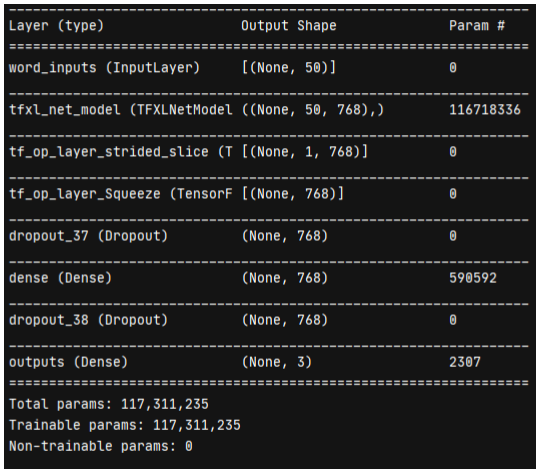


Figure 7: Frequency of Curse Words in Training Dataset



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_ids (InputLayer)       [(None, 40)]              0
_________________________________________________________________
bert (BertModelLayer)        (None, 40, 768)           108890112
_________________________________________________________________
lambda (Lambda)              (None, 768)               0
_________________________________________________________________
dropout (Dropout)            (None, 768)               0
_________________________________________________________________
dense (Dense)                (None, 768)               590592
_________________________________________________________________
dropout_1 (Dropout)          (None, 768)               0
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 3845
=================================================================
Total params: 109,484,549
Trainable params: 109,484,549
Non-trainable params: 0
_________________________________________________________________
```

Figure 8: BERT Summary



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
word_inputs (InputLayer)     [(None, 50)]              0
_________________________________________________________________
tfxl_net_model (TFXLNetModel ((None, 50, 768),)        116718336
_________________________________________________________________
tf_op_layer_strided_slice (T [(None, 1, 768)]          0
_________________________________________________________________
tf_op_layer_Squeeze (TensorF [(None, 768)]             0
_________________________________________________________________
dropout_37 (Dropout)         (None, 768)               0
_________________________________________________________________
dense (Dense)                (None, 768)               590592
_________________________________________________________________
dropout_38 (Dropout)         (None, 768)               0
_________________________________________________________________
outputs (Dense)              (None, 3)                 2307
=================================================================
Total params: 117,311,235
Trainable params: 117,311,235
Non-trainable params: 0
```

Figure 9: XLNet Summary

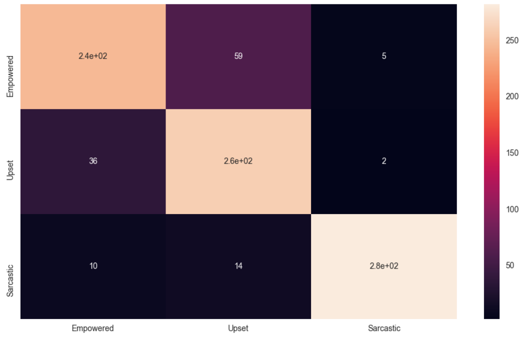


Figure 10: Confusion Matrix

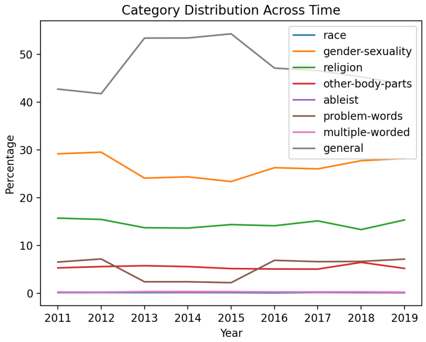


Figure 11: Distribution of Curse Words Category in Historical Data

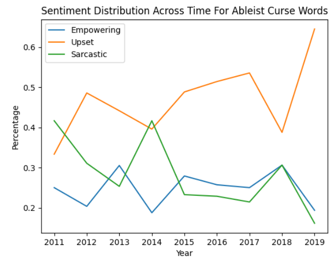


Figure 12: Ableist Curse Words: Emotional Context Trend Over Time

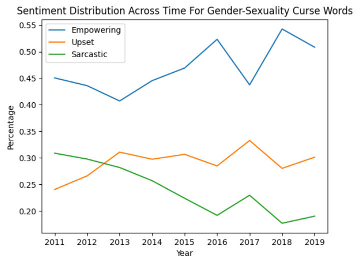


Figure 13: Gender-Sexuality Curse Words: Emotional Context Trend Over Time

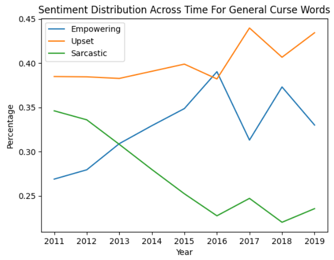


Figure 14: General Curse Words: Emotional Context Trend Over Time

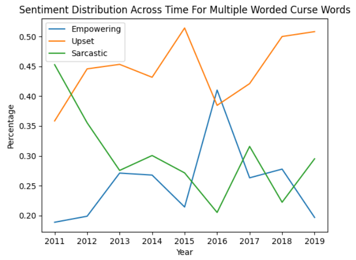


Figure 15: Multi-worded Curse Words: Emotional Context Trend Over Time

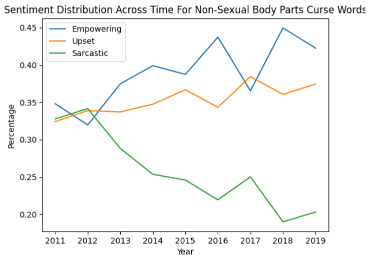


Figure 16: Non-Sexual Body Parts Curse Words: Emotional Context Trend Over Time

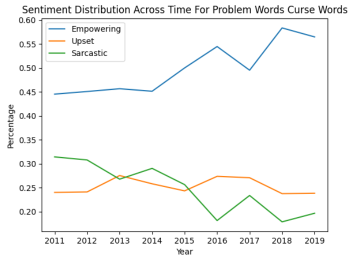


Figure 17: Problem-Words Curse Words: Emotional Context Trend Over Time

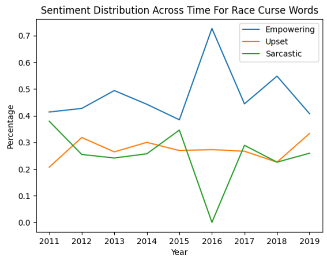


Figure 18: Race Curse Words: Emotional Context Trend Over Time

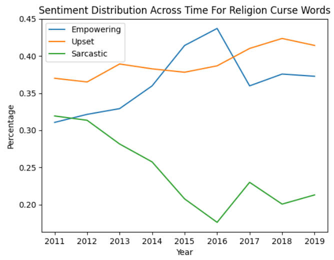


Figure 19: Religion Curse Words: Emotional Context Trend Over Time