SMU Data Science Review

Volume 3 | Number 2 Article 12

2020

Toxic Language Detection using Robust Filters

Deepti Kunupudi
Southern Methodist University(SMU), nkunupudi@mail.smu.edu

Shantanu Godbole
Southern Medhodist University, sgodbole@mail.smu.edu

Pankaj Kumar
Southern Methodist University, kumarp@mail.smu.edu

Suhas Pai
NLP Stream Owner,Aggregate Intellect, info@piesauce.edu

Toxic Language Detection using Robust Filters

Deepti Kunupudi[1], Shantanu Godbole[1], Pankaj Kumar[1], and Suhas Pai[2]

[1] Master of Science in Data Science, Southern Methodist University, Dallas TX 75275 USA {nkunupudi, sgodbole, kumarp}@smu.edu
[2] NLP Stream Owner,Aggregate Intellect {info}@piesauce.edu

Abstract. Social networks sometimes become a medium for threats, insults, and other types of cyberbullying. A large number of people are involved in online social networks. Hence, the protection of network users from anti-social behavior is a critical activity [19]. One of the significant tasks of such activity is the detection of toxic language. Abusive/Toxic language in user-generated online content has become an issue of increasing importance in recent years. Most current commercial methods use blacklists and regular expressions; however, these measures fall short when contending with more subtle, lesser-known examples of hate speech, profanity, or swearing[6]. Abusive language classification has become an active research field with many recently planned approaches. However, while these approaches address some of the tasks' challenges, others remain unsolved, and directions for further research are needed. In this work, we intend to create a robust text classification to detect hate speech online. Our analysis explores a two-step approach in performing the classification of abusive language, first detecting the abusive language, and secondly, using the method of the model. We will explore various aspects of abusive language classification and their correlation to the toxicity and leverage the results and verify the toxicity in the language. Additionally, we will simulate the data-set to test to see if we can circumvent the system. Apart from toxic detection, the utmost importance is equally essential is model efficiency. We will be reviewing the models and test a cascade/singular model that achieves high throughput in the average case while maintaining a high accuracy by combining multiple approaches.

Keywords: Natural Language Processing · Text Mining · Recurrent Neural Networks · Social Media · Abusive Language · Neural Networks · User-generated content · Hate Speech · Toxic Language

1 Introduction

The internet has widely impacted the way we communicate. Online communities have grown to become essential places for interpersonal communications [11]. Over the years, social media and social networking usage have been increasing exponentially due to the internet. Flood of information arises from online conversation daily as people can discuss, express themselves, and share their opinion via these platforms. While this situation is highly productive and could contribute significantly to the quality of human life, it could also be destructive and

enormously dangerous [12]. Anytime a user is engaged in an online platform like forums, comments, or social media, there is always a serious risk that he or she may be the target of ridicule or receptor of abusive language. As a result, different platforms and communities find it very difficult to facilitate fair conversation. They are often forced to either limit user comments or get dissolved by shutting down user comments completely [12]. In general, Abusive language refers to any insult, vulgarity, or profanity that debases the target [13]. To combat abusive language, many internet companies have standards and guidelines that users must adhere to and employ human editors, in conjunction with systems that see regular expressions and blacklist, to catch in-appropriate language and thus remove the post. In the current scenario of social media, where it is more prominent, an automated abusive language classifier/filter is in place; however, it seems nearly impossible to resolve the issue successfully. The various techniques are used for human-free detecting the toxic comments[18]. Bag of words statistics and bag of symbols statistics are one typical source of information for the toxic comments detection[18]. Usually, the statistics-based features which are used are the length of the comment, number of capital letters, number of exclamation marks, number of question marks, number of spelling errors, number of tokens with non-alphabet symbols, number of abusive, aggressive, and threatening words in the comment, etc.[16]. A high count of bad words in the comment increases a chance to classify it as toxic. However, there are some difficulties with the usage of the bad words statistics [19]. There are other types of toxic language where euphemisms are involved. These act as an instrument to mask intent, have long been used throughout history. The euphemistic offensive language also qualifies as hate speech because it targets communities based on race, religion, and sexual orientation[15]. As a result, hate speech that relies on intentional word substitutions to evade detection can be considered euphemistic hate speech[15]. Notably, Waseem et al. (2017) provide a typology of abusive language, categorizing abusive language across two dimensions: 1) target of abuse and 2) degree of abuse being explicit[15]. In terms of target, they distinguish directed hate (hatred towards an individual. Example: 'Go kill yourself') and generalized hate(hatred towards a recognized group. Example: 'So an 11-year-old n***er girl killed herself over my tweets? that's another n***er off the streets!!'). For the second dimension, the authors differentiate between explicit (containing 'language that which is unambiguous in its potential to be abusive.' Examples could be language containing racial slurs) and implicit (containing 'language that does not immediately imply or denote abuse.' Example: 'Gas the skypes' (Magu et al., 2017)) hate. The authors discuss the role of context, which is needed to identify hate correctly. There are various proposed solutions include bidirectional recurrent neural networks with attention (Pavlopoulos et al., 2017) and the use of pre-trained word embeddings (Badjatiya et al.,2017). However, many classifiers suffer from insufficient variance in methods and training data and, therefore, often tend to fail on the long tail of real-world data (Zhang and Luo, 2018), and challenges to the current solutions are still error-prone [9].

Recent trends in machine learning and deep learning in the context of natural language processing have yielded significant improvements to machines' ability to detect subtleties in natural language, including toxic language or hate speech [1]. Currently, we have a range of publicly available models served through the Perspective API, including toxicity. But the current models still make errors, and they don't allow users to select which types of toxicity they're interested in finding[1].

Our project is motivated by these challenges, and it is mainly focused on detecting hate speech online even in the face of lexical alterations or euphemisms which are made to the text to evade abusive language filters. We compared the accuracy using different classifiers with neural network models and benchmarked the results and provide the possibility for future improvements. For this project, we took the deep learning approaches to solve the toxic speech classification problem, which would detect toxic language even in the face of lexical alterations or euphemisms made to the text. Character-based models are one of the methodologies in identifying the toxic content. Character-based models became more popular for different natural language processing tasks, primarily due to the success of neural networks. They provide the possibility of directly model text sequences without the need of tokenization and, therefore, enhance the traditional preprocessing pipeline [17]. This is somewhat an innovative way to detect the toxic material, unlike the other methods. A multi-headed model that's capable of detecting different types of toxicity was developed[14]. As an alternative to character-based embedding, we explored hyperbolic embedding. The work developed addressed the effect of incorporating various text transformations on model accuracy for toxic detection. The characteristics of the data used for this project is mainly focused on toxic data on social media. For model development, we leveraged the publicly available social media dataset for toxic language detection.

To summarize, the contributions to this project are to analyze the data and focus on different types of abusive content. We leveraged character-based models for detecting abusive language. We compared the effectiveness of end-to-end character-based models, which shows that the techniques perform better than word-based models. We demonstrated how fine-tuning large pre-trained language models is enhancing the state of the art on a few of the abusive language datasets and show that the domain shift isn't considered when applied to abusive language datasets [2].

2 A Related Work

2.1 Early Toxic Language Detection Models

Several researchers have been researching various topics that will make the system more tolerant of different methodologies used to evade the filters. In some earlier works, n-gram with parts of speech was used in an SVM model to classify the language sentiment (Warner and Hirschberg, 2012). In 2016 (Nobata et al., 2016) followed a similar methodology to use linguistic features to train a Machine

Learning algorithm that will detect 'Hate Speech.' [5]. Authors at (Alorainy et al., 2018)'s researched hate speech by identifying words that contribute to hate speech. They designed a custom pronoun lexicon and semantic relationships to capture the linguistic differences when describing the messages in the in-and out-, and trained word embedding model on that data.

Results published by various other papers also indicate there is a consensus that character-based models are more accurate than their counterparts that are based on n-gram and word embedding. The character-based models are more efficient in recognizing the basic obfuscations of the abusive words used to bypass the filters (example: sh!t). The obfuscations are usually complicated enough that they can not be identified by algorithms like spelling correction, checking Levenshtein distance. The character sequence, in this case, can retain some of the character sequences encoded information form the original words and therefore proves to make an educated guess than an Out of Vocabulary word as compared to word embedding models. [1] We explored the possibility of similar efficiency by using hyperbolic embedding in which related words exist in the same hierarchy.

2.2 Bias in the Models

Researchers have also been paying attention to the bias in such systems. Often in computer vision and facial recognition, models are found to be less accurate in detecting the people with minorities in the dataset. Natural Language Processing models are also prone to the same bias when the model is trained in a supervised fashion. The number of gender bias errors are introduced as a result of the bias in sentiment classification. While hate speech and abusive language detection have become an essential area for natural language processing research (Schmidt and Wiegand, 2017), very little research has been done to understand the systematic bias. For example, if there is a mention of a minority race in different hateful contexts, the model can be biased towards classifying the actual name of the race as a hate word. There can be false positives in such cases that can affect the performance of the models negatively (Kwok and Wang, 2013; Burnap and Williams, 2015; Davidson et al., 2017). Studies show that there is a gender, and race bias can be observed in such models. There can also be a positive bias introduced in such systems. If the words that are overly non-toxic in their meaning, such as 'love,' are used, they can outweigh the other toxic words and give a false negative. For example: "He is such a (toxic word), love!".

2.3 Interpreting the Model

Interpreting the model is also an important part of the toxic language detection process in order. Whenever a statement or text is classified as toxic by the model, companies or the language moderators are expected to explain why the particular sentence was classified as profane/hateful or toxic. The research published by Svej [3] proposes the explanation of the model in 2 steps:

[1] piesauce.com

1. Detection of inappropriate comments, which is done as a binary classification problem. Recurrent Neural Network hereafter referred to as 'RNN' was used with that interpreted the text to have a positive or negative classification. The version of RNN used was the RCNN recurrent cells (Barzilay et al., 2016) instead of LSTM that is more commonly used.
2. Highlighting the inappropriate parts - which is done by selecting the text that contributes most towards the classification.

The authors propose the method of joint learning for the model to be explainable. The output that is generated provides feedback on the quality of the rationale. The rationale predominantly determines the classification. If the valid rationale is used, the classification is going to be correct with better accuracy. Furthermore, the rationale is restricted to have few words and that too the words that are close to each other in the embedding space.

The function that expresses this condition is:

$$loss(x, z, y') = \| \ clas(z, x) - y' \ \|_2^2 + \lambda_1 \|z\| + \lambda_2 \sum_{n=1}^{K-1} |z_t - z_{t+1}| \qquad (1)$$

where x is original comment text, z contains binary flags representing non/selection of each word in x, (z, x) contains actual words selected to rationale, y is correct output and K is the length of x and also z respectively. [3]

With the methodology we plan to implement - this will be one of the most potential methods we plan to use to evaluate and explain the model.

3 Data

The dataset includes over approximately 115k labeled discussion comments from English Wikipedia. Multiple annotators labeled each comment on whether it contains a personal attack. It also includes some crowd-worker demographic data, but it is not in scope for this project. Each comment has a label as 'Toxic' or 'Non-Toxic.' The comments are mainly in the English language for the simplicity of the model.

As part of data pre-processing and augmentation, the data is cleaned, and related information is extracted using Python libraries. For our binary classification task, the label is given as either toxic or non-toxic.

In general, the complete dataset is divided into three parts to avoid overfitting and model selection bias - 1. Training set 2. Cross-Validation 3. Testing Set. To separate the data, we used the 60-20-20 split to fit the three parts. The model is trained on the training set, optimizes the hyperparameters on the validation set, and evaluates the testing set's performance.

Although the dataset is a binary classification with toxic or non-toxic content, it is highly imbalanced where seventy percent of the data is non-toxic, and thirty percent of the data is toxic. In the same solution, we will detect the toxicity and improve the performance of the models. The toxicity classes have a very

skewed distribution. This could create a problem and limit the model the ability to learn how to distinguish and predict classes and overall affect the predictions or evaluation results. To overcome the class imbalance problem, we applied stratification. Stratification is the technique to allocate the samples evenly based on sample classes so that the training set and validation set have a similar ratio of classes. It is essential to ensure training and validation sets share approximately the same ratio of each class so that we can achieve consistent predictive performance scores in both sets.

4 Methodology

The initial challenge is data representation for modeling. In current neural networks, we feed our data using word embedding. Word embedding tries to retain proximity of related words in its representation. This technique is much different from the traditional approach of using one-hot encoding in which each word is independent is from others, and there is no relative ordering.

In most of the word embedding that is present, like Glove used to find the distance between words using Euclidean space measures. The concept of distance in Euclidean space helps establish proximity of words, but it doesn't measure specificity among words. We need one more level of measure to express that property.

In 2017 Nickle and Keila [10] in their work proposed Poincaré Embeddings; the model uses hyperbolic space instead of Euclidean space to create word embedding. Embedding in hyperbolic space is capable of express not only proximity but also specificity.

4.1 Hyperbolic Embedding

Hyperbolic embeddings have been proposed as a way to capture hierarchy information for use in link prediction and natural language processing tasks[7]. These approaches are an exciting new way to fuse rich structural information (for example, from knowledge graphs or synonym hierarchies) with the continuous representations favored by modern machine learning [7]. It can preserve graph distances and complex relationships in very few dimensions, particularly for hierarchical graphs. These embeddings offer excellent quality with few dimensions when embedding hierarchical data structures like synonyms or type hierarchies [7]. In contrast, Bourgain's theorem shows that Euclidean space is unable to obtain comparably low distortion for trees—even using an unbounded number of dimensions [7]. Moreover, hyperbolic space can preserve specific properties; for example, angles between embedded vectors are the same in both Euclidean space and the Poincare model, which suggests embedded data may be easily able to integrate with downstream tasks [4]. Figure 1 illustrates the relations between words both on hierarchical as well as lexical level. To understand the hierarchical relation, we can look at the more generic word dark at the root. As we move to the periphery, words get more specific, like the relation "dark -

semidarkness- dimness" shows one hierarchy. Similarly, the words "shadow and shade" are lexically closed. Poincare models are best suited for representing hierarchical data. These are the method to learn vector representations of nodes in a graph. The input data is of the form of a list of relations (edges) between nodes. The model tries to learn representations such that the vectors for the nodes accurately represent the distances between them. It is a natural hierarchy in words of natural language. As we can see in figure 1, a word at the center is more general, and as it goes towards the periphery, it becomes more specific. This kind of representation helps us identify the word specificity in our embedding. The proximity of words is represented by there distance across surface and hierarchy across the line.

$$d(u, v|) = arcosh(1 + 2(\| u - v \|^2 /(1- \| u \|^2)(1- \| v \|^2)) \quad (2)$$

As Shown by Nikckel and Keila 2017, the distance formula in the Poincare model is differential; hence it is suitable for gradient-based optimization and other machine learning techniques.

Poincare embeddings capture notions of both hierarchy and similarity, adding it one more degree of freedom.

1. Adding similarity by connecting the nodes close to each other and those unconnected nodes far from each other.
2. Later adding hierarchy by placing nodes higher when closer to the origin and in lower hierarchy nodes when farther from the origin.

In generating the model for hyperbolic embedding, we generate the word and the adjective sync set closure from wordnet [2]. Later, using the Poincare embedding from gensim[3] to create the embedding. Based on the output converting the Poincare embedding into word2vec format. This output is used to train and infer our model. The performance of the model is evaluated by a loss function that penalizes low distances between unconnected nodes and high distances between connected nodes. The main advantage of these embeddings is that the model is learned in hyperbolic space compared to Euclidean space. As mentioned in an earlier note, the hyperbolic spaces are more suitable for capturing hierarchical relationships inherently present in the graph.

Model performance will capture the characteristics of abusive language using standard classification performance metrics. The metrics evaluated based on precision, recall, and f1-score apart from accuracy.

4.2 Creating an encoded hyperspace for toxic phrases and euphemisms

In this attempt, we have added two robust filters to classify toxic phrases and euphemisms further. Both filters are based on the concept of transfer learning.

[2] wordnetweb.princeton.edu/perl/webwn
[3] pypi.org/project/gensim/


Toxic Sample For Word Dark
gloom.n.01
obscurity.n.0
cloudiness.n.02
id1
id2
night.n.05
blackout.n.02
total_darkness.n.01
semidarkness.n.01
limness.n.01
dark.n.01
shade.n.01
umbra.n.01
shadow.n.01
penumbra.n.01
0.2
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.2
−0.1
0
0.1
0.2

Fig. 1.

We performed an ablation study and retrieved the tri-grams that influenced the model the most. We narrowed down the scope of these filters to tri-grams as most of the toxic phrases will be captured in tri-grams. After running the ablation study, according to the label, an auto-encoder is trained using contextual vectors. For example, true positive and false negative phrases will train the toxic encoded space, and true negative and false positive phrases will train the toxic encoded space acquiring the toxic and non-toxic context of the phrases. In this study, evaluation is done on four cases: When the sentence is 1. true-negative 2. true-positive 3. false-positive 4. false-negative. In each of the cases, we identify the top tri-gram hoping to capture the words' context. Later, we create two auto-encoders with toxic and non-toxic, and the probability score is generated to identify these two classifiers.

For testing, an incoming sentence is divided into the noun and verb phrases. The phrases are checked against both the encoded spaces and the reconstruction loss is calculated using cosine difference between each generated word. The loss will give us more insight into which phrase was inferred as toxic, hence aiding the explainability, interpretability, and the fairness of the classification.

5 Models

5.1 Understanding the Networks

The use of RNNs in the field of NLP has increased drastically in the past few years, and RNNs (Basic RNNs, Gated Recurrent Units "GRUs", Long Short Term Memory "LSTM") are well established to perform great on the language-related tasks like summarization, sentiment analysis, translation, and others. Neural Networks are known or referred to as a black box. It is not because it is impossible to decode 'why' a neural network classified or generated specific text. Still, it is tough to understand and provide explanations based on the calculations performed by a Neural Network.

We tried to classify whether the particular text used as an input to the Neural Network is toxic or not. With that being said, there is always a chance that the text might be incorrectly classified as toxic or abusive when it is not (false positive). In this world of free speech, we cannot just classify text as abusive because our model says so. In some instances, it is very evident why the network could have classified some text as abusive based on our linguistic knowledge. However, in many cases, it might not be very apparent. Therefore, it is essential to understand what the model behaved in a certain way to explain and improve the model's performance.

5.2 LSTM Model to establish a baseline

The project focuses on studying the effects of different models based on the baseline model. We implemented the baseline model to get some preliminary results on the data. Our most straightforward model is based on LSTM with

regular embedding with features based on n-grams(3) into the model and tunes the hyper-parameters with rmsprop as an optimizer and binary cross-entropy for loss. Accuracy would be the measure of metric based on which the model is evaluated.

For our word analysis, the model is trained in our word-level embedding using Keras embedding layer package. For model evaluation, only using accuracy as the evaluation metric is not preferable due to imbalanced data, and the accuracy can be biased. We compared based on performance metrics across the models using loss, accuracy, precision, recall, and f1-score. Of all, f1-score is the preferred score for measuring along with others. F1-score can be interpreted as the harmonic mean of precision and recall values, and it shows the real model performance with skewed data. This would become an essential lever to gauge the classifier's performance in either case of balanced or imbalanced classes.

5.3 Hyperbolic embeddings generation

To generate hyperbolic embedding, we used the gensim Poincare module. Embedding length is 16 vectors, and input data was sync sets from wordnet. We trained for 50 epochs to generate embeddings used in our model. We converted hierarchical format to word2vec format to read it and feed it to Keras embedding layer. We have fine-tuned it further after initializing the weight with generated embedding.

5.4 Auto-encoder Filters

The two filters are auto-encoder networks - that are similar to sequence to sequence networks are used, as shown in the figure 2. We call them positive Auto-encoder and negative Auto-encoder. Both the networks are 4 Layered Stacked LSTM with the Encoder nodes 1024 - 720 - 128 - 64, and the decoder nodes are 64 - 128 - 720 - 1024. 1024 nodes were chosen to resemble the 1024 dimensional vector that we get from Elmo. A sequence of 3 words, three vectors from ELMO, is passed, and each network tries to re-create the vectors. Figure 2 explains the architecture of the autoencoder.

Both the networks are trained after performing ablation studies, and reconstruction losses are calculated using cosine similarity between the original vector and the regenerated vector, over both the networks. A threshold is used to determine how positive the phrase is or how toxic the phrase is. If we find any phrase that is above our threshold limit, the phrase and the text is flagged as toxic.

5.5 Model Explainability

In this section, we have attempted to explain our model using examples to see how it performs. We executed the two auto-encoders that we have trained to perform ablation studies on trigrams (trigrams are 3 words sequential words from any text) to check the phrase is negative or positive based on the cosine difference between original and the reconstructed vectors.

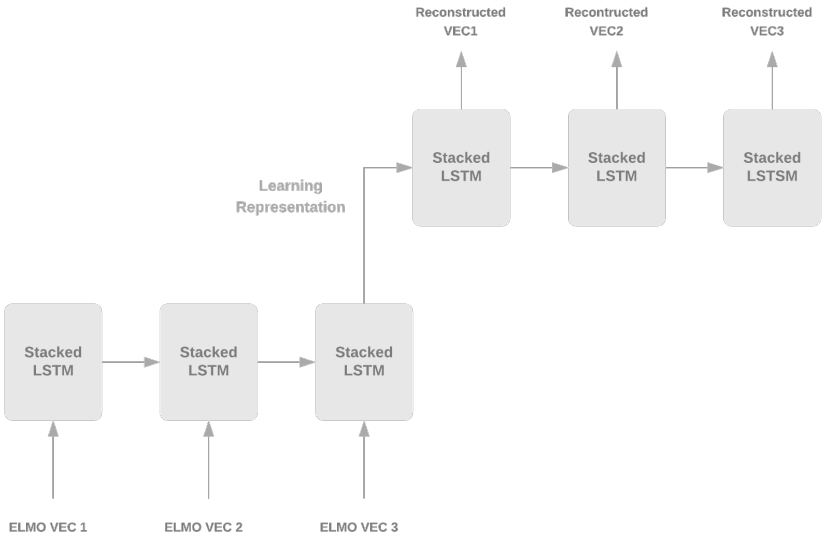

Fig. 2.

1. Adding similarity by connecting the nodes close to each other and those unconnected nodes far from each other.
2. 'May 2014 - You have been blocked forever for vandalism' was labeled as abusive in the dataset. The trigram "forever for vandalism" was an abusive phrase in the model evaluation. Linguistically the word vandalism has a negative connotation. Therefore we can say that the model correctly performed the classification.
3. "You have proved yourself to be the cruel personal attacker to be banned forever, unfortunately." "be the cruel" was pointed as one of the abusive phrases in the model evaluation, therefore this again linguistically can be verified.
4. "I have just returned from a several months block, which I feel I should not have had as my edits are not vandalism and are all correct." Toxic phrase: This is an example of a non-toxic record from the dataset. The phrase "is not vandalism" was picked as the most toxic phrase likely because of the word vandalism. However, several months" was given as the most positive (non-toxic phrase).
5. "That is very kind of you to say. If I missed anything that should be added or deleted, feel free to let me know or even change it yourself." This is labeled as positive, but under the hood, "even change was" the most (slightly) negative phrase and "of you to" is the most positive phrase. In this explanation, both the negative and the positive phrase linguistically do not make much sense

but can give us some insight into how the model worked. This explanation can further help us to tune the model better to avoid false negatives and false positives.

The ablation process that was used to train the auto-encoders is a very computationally heavy process. Our observation was that the explanation was imprecise in some instances and needed human interpretation. However, the auto-encoder was trained on limited records, and there can be a lot of scope for improvement by training it over more records.

6 Discussion

For base model LSTM without autoencoder filters, our optimum model resulted from 128 LSTM nodes. When running on the test set, the model obtained 0.87 test accuracy, 0.87 precision, 0.87 recall, and 0.86 f1-score. For the initial model, the results looked promising and extended work on other models gave us better insights. The following table summarizes this result.

Model Performance Results			
Model Name	Accuracy	Nodes	F1 Score
LSTM Baseline	0.87	128/64	0.86
LSTM + Hyperbolic	0.89	32	0.86

For LSTM with hyperbolic embeddings, our optimum model resulted from a single layer 32 node LSTM. The optimizer and loss are set similar to the baseline model. The test set obtained 0.89 accuracy, 0.87 precision, 0.87 recall, and 0.86 f1-score. The results look similar to the baseline model. Similarly, for LSTM with hyperbolic embeddings with autoencoder filters, our optimum model resulted from 32 LSTM layers with 32 output units in each layer. The optimizer and loss are set similar to the baseline model. The test set obtained 0.867 accuracy, 0.86 precision, 0.87 recall, and 0.86 f1-score.

7 Conclusion

We reviewed the binary classification for toxic and non-toxic from the dataset used for evaluating different models that can predict the abuse or toxicity in the text. The model with hyperbolic embedding with almost a fourth size of the baseline model performed marginally better than the baseline model. Therefore, the hyperbolic embedding that is hierarchically arranged carries more information; thus, the model can be made simple. The auto-encoder explanation can be more precise with further training.

8 Ethics

The ethics of social scientific and social relevant computational research is under scrutiny in recent years. Academic research can be used to not only monitor

and capture social behaviors but also influence and manipulate them (Ruppert, Law, and Savage, 2013). Given the sensitivity of this area, ethics should be at the forefront of all research [4]. Political science research also suggests that any form of extremist behavior, such as online hate, could fuel social antagonisms and even reprisals Eatwell, 2006) [8]. As such, the ethical case for moderating online content is secure. However, research is unevenly distributed, with far more attention paid to abuse in English as well as abuse directed against defined targets. This is partly due to how research is organized [4]. The listed above also impact the researchers who are researching the abusive content by reading and thinking about the content. This can inflict considerable emotional harm on researchers, mainly through vicarious trauma [4]. Another primary ethical consideration for analysis and model building is to ensure data is not biased. We will ensure that we have enough balanced data between toxic and non-toxic information. One more ethical consideration would be privacy. Similarly, we will ensure that the data is not extracted from any personal social media sites. The data set which will be leveraged in this project would be from publicly available social media data.

9 Future Work

In the field of Natural Language Processing, recently bigger models have gained a lot of prominences. The models are very computationally expensive to train as well as to use for the prediction of the texts. This is because of the use of multi-layer transformer networks and the dimensions of the embedding vectors used. We found hyperbolic embedding slightly gaining performance over the regular 'LSTM.' However, there is still a lot of scope to improve the model adding few more dimensions to the hierarchy, which is not expected to be computationally expensive. The Auto-encoded filters, even though trained on just the training part of the dataset that we had, can also be fine-tuned on other datasets, which will contribute to enriching the vocabulary of the filters the detection of toxic language more robust.

References

1. Betty van Aken, Julian Risch, R.K., Loser, A.: Challenges for toxic comment classification:an in-depth error analysis
2. Al-Onaizan, S.B.B.S.G.Y.: Neural word decomposition models for abusive language detection
3. Andrej, Pikuliak, M.M.: Improving moderation of online discussions via interpretable neural models (Sep 18, 2018)
4. Bertie Vidgen, Dong Nguyen, R.T.A.H.S.H.H.M.: Challenges and frontiers in abusive content detection
5. Brassard-Gourdeau, E., Khoury, R.: Impact of sentiment detection to recognize toxic and subversive online comments (Dec 4, 2018)
6. Chikashi Nobata, Joel Tetreault, A.T.Y.M.: Abusive language detection in online user content

7. Christopher De Sa, Albert Gu, C.R.F.S.: Representation tradeoffs for hyperbolic embeddings
8. EATWELL, R..: Community cohesion and cumulative extremism in contemporary britain.
9. Hu, A.K..K.: Toxic speech detection
10. Maximilian Nickel, D.K.: Poincare embeddings for learning hierarchical representations 2017
11. Noé Cécillon, Vincent Labatut*, R.D., Linarès, G.: Abusive language detection in online conversations by combining content- and graph-based features
12. P. A. Ozoh, A. A. Adigun, M.O.O.: Identification and classification of toxic comments on social media using machine learning
13. Park, J.H., Fung, P.: One-step and two-step classification for abusive language detection on twitter
14. Phd, Patrick Adigun, A..O.O.: Identification and classification of toxic comments on social media using machine learning techniques
15. Rijul Magu, J.L.: Determining codewords in euphemistic hate speech using word embedding networks
16. Salminen, J., e.a.: Anatomy of online hate: developing a taxonomy and machine learning models for identifying and classifying hate in online news media. in: Proceeding of the twelfth international aaai conference on web and social media, 2018, pp. 330-339 (2018)
17. Schütze, H.A.A.: Overview of character-based models for natural language processing
18. Serhiy Shtovba, Olena Shtovba, M.P.: Detection of social network toxic comments with usage of syntactic dependencies in the sentences
19. Serhiy Shtovba, Mykola Petrychko, O.V.S.: Detection of social network toxic comments with usage of syntactic dependencies in the sentences