Toxic Comment Classification

Anjali Chadha
Texas A&M University
anjali_chadha@tamu.edu

Puneet Kohli
Texas A&M University
puneetkohli@tamu.edu

Sukhdeep Gill
Texas A&M University
sgill12@tamu.edu

***Abstract*—There is a constant threat of abuse and harassment on the internet, with over 70% adult users having witnessed some form of online harassment. Due to the personal nature of the damage caused, dealing with online harassment has become extremely important. One major form of internet harassment is "toxic" textual comments on forums, websites, and social media platforms. In this paper, we aim to classify comments from Wikipedia talk page edits into various toxic classes. We evaluate models based on both traditional machine learning and Natural Language Processing (NLP) techniques as well as multi-headed neural network based methods. Additionally, we propose solutions to deal with various types of adversarial attacks on our models. Our best model uses a Bi-directional Gated Recurrent Unit (GRU) based Recurrent Neural Network (RNN) with a FastText embedding layer, achieving 0.9878 mean class-wise area under the ROC curve. Although our results are theoretically accurate, we further investigate how this may be in part due to the inherent bias and imbalance in our dataset.**

I. Introduction

In the recent years, technology has boosted the usage of online communities as more people have access to the Internet these days. The idea of these virtual communities is to facilitate discussion environments where people can freely express themselves and engage in conversations of their interest while contributing useful information to the internet. This results in tons of content being uploaded on these platforms every minute which may contribute to several problems like online harassment, bullying, hate, etc. According to a 2014 Pew Research Institute survey, 40 percent of internet users have personally experienced online harassment, and 45 percent of those harassed have experienced severe harassment (i.e. physical threats, sustained harassment, sexual harassment, etc.) [1]. The victims of online harassment get discouraged and tend to decrease their participation in discussions. In extreme cases, cyberbullying can even lead to victims committing suicide [2]. In order to effectively facilitate conversations without any threat, platforms need to moderate user content. This content is filtered manually and the people performing this job tend to suffer from PTSD-like symptoms. Hence, there is a strong need to automate the moderation process to effectively detect toxic language in order to reduce the workload on manpower and to provide a safe discussion platform to the users.

In this project, we built and compared the performance of different models using machine learning techniques. These models are capable of detecting different types of toxicity such as threats, obscenity, insults, and identity-based hate. Additionally, these proposed models have the ability to deal with hostile attacks in order to provide a learning and engaging environment to the system users. We found that neural network based model which uses a Bi-directional Gated Recurrent Unit outperforms the traditional and other deep learning models.

II. Literature Review

Detecting toxic language is often more difficult than what appears to be. A sophisticated tool should be able to detect obfuscated abuses, understand the context of a conversation, detect insults in a sarcastic comment. The definition of online abuse is quite wide. Different works have focused on different aspects of this definition. [3] focuses on detecting hate speech towards a particular ethnic group, [4] [5] focus on specific social platforms like YouTube, Twitter, while others, such as [6] concentrate on detecting profanity.

A. Traditional methods

In this section, we will discuss the previous works which applied the traditional NLP and text classification methods (non-neural networks) to toxic speech detection problem. Some works created a blacklist (a collection of words knows as hateful or insulting) for handling this problem. [5] [7] prepare a list containing words more focused towards ethnic slurs, LGBT slang or offensive words related to handicapped people. However, these lists need to be updated regularly. Moreover, an insult offensive to one community might be completely fine to use in another community. As noted in [8], a blacklisted word might be a non-offensive word in a different context. Eg. *suck* might be offensive when used in the sentence *"You suck!!"*, however, completely innocuous when used in *"I sucked my thumb as child"* comment.

[9] is one of the earlier works on the toxic language detection. It uses a supervised classification approach which uses combination of character and word n-grams and manually developed regular expression patterns. [4] along with n-gram and regular expressions, uses a combination of lexical and parser features such as comment and token lengths, capitalization, words that cannot be found in English dictionaries and the number of non-alpha numeric characters present in the tokens, for detecting abusive language in YouTube comments. [5] [10] performs word clustering using LDA, which produces for each word a topic distribution indicating the degree to which a word belongs to a particular topic.

Some previous works have used sentiment analysis to detect hate speech because a hateful comment can often be ascribed to negative sentiment. [11] includes the count of positive,

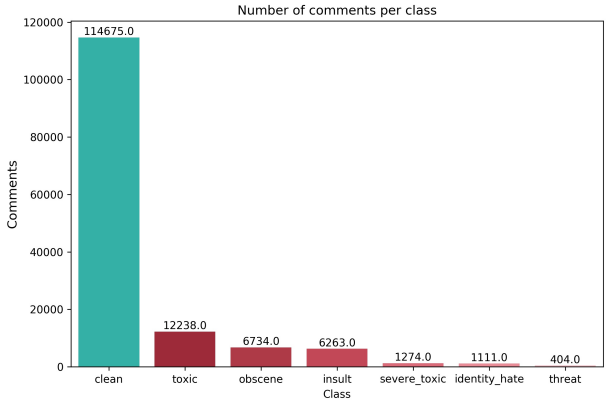

Fig. 1. Distribution of labels in the dataset

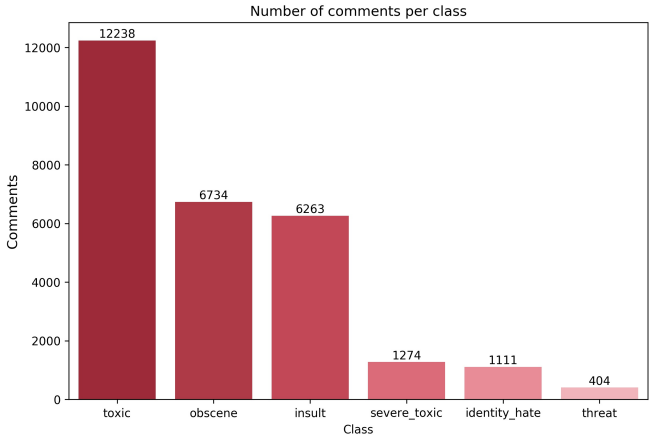

Fig. 2. Distrubution of comments among toxic classes

negative and neutral words in a comment as one of features in its model.

[7] uses syntactic features like Parts-of-Speech (POS) tags, dependency relations, for hate speech detection. The major motivation was to capture long range dependencies between words which n-grams may not be able to do. For the comment, *"Jews are lower class pigs"*, n-grams might fail to detect this as toxic comment. However, dependency parser would generate a tuple *are(Jews, pigs)* where Jews and pigs are children of are.

[12] combines crowd-sourcing and machine learning techniques to tackle the toxic comment classification problem. They use Wikipedia dataset for their analysis. As per this paper, majority of the personal attacks are not the result of a few malicious users nor primarily the consequence of allowing anonymous contributions

B. Neural Methods

Neural Networks have been successfully applied to the area of natural language processing. RNNs, in particular LSTMs are known to perform well on the sequential data and have been successfully used for NLP research [13]–[17]. Many recent works have explored CNNs for the text classification. [18]–[21] notes that character level convolutional neural networks are effective for text classification tasks. [22] introduced recurrent CNN model which captures both contextual information with the recurrent structure and uses convolutional network for the text representation. Recurrent CNNs have been shown to outperform plain CNNs and other standard classifiers.

Recently, word embeddings techniques based on neural networks have been used for the similar purposes. In the context of abusive language, as we need to classify sentences or passages rather than individual words, approaches similar to word2vec for the sentences were used. One simple way to create paragraph/sentence vectors is by averaging the vectors of all words appearing in a paragraph/sentence. However, as noted in [7], this technique is not very effective even when word embeddings were obtained from a domain specific corpus. [23] uses paragraph2vec (paragraph embeddings) to learn a low dimensional representation of a comment. Embedding approaches address the problems of high dimension and sparsity.

III. Problem Formulation

The goal of this work is to automatically detect various forms of "toxic" comments from a corpus of Wikipedia talk pages (also known as discussion pages). We formulate this as a six-class multi-headed classification task, where each class corresponds to a different type of online attack. The classes are as follows

- toxic — a negative or distasteful comment
- severe toxic — an extremely negative comment, often containing swear words
- obscene — a comment containing strongly offensive and inappropriate language
- insult — a disrespectful or scornful comment
- identity hate — a comment directly attacking individuals identifying with a certain identity (such as gender, race, religion, etc.)
- threat – a comment made with an intent to cause damage or danger to an individual or individuals

A comment may belong to none, or multiple of the classes listed above. For example a comment *"karma is a fucking bitch"* would be flagged as both toxic as well as obscene. On the contrary, a comment *"I appreciate your views!"* would not be flagged as any of these toxic classes. Figure 1 shows the overall distribution of comments among previously mentioned toxic classes.

We report our results using the class-wise mean Area under the Receiver Operating curve (AU-ROC), and log-loss metrics.

IV. Data Description

To train and evaluate our model we used the Jigsaw Toxic Comments corpus which contains a total of 300,000 comments scraped from Wikipedia talk page edits. From the corpus, around 160,000 comments were labelled with none or multiple of the six toxic classes and 140,000 were unlabelled.

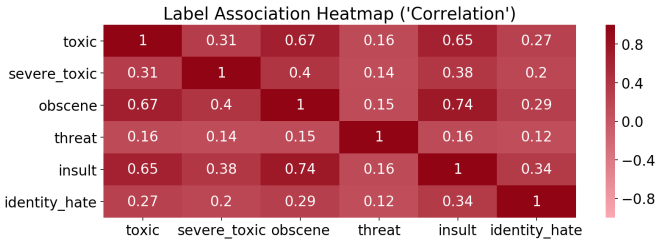

Fig. 3. Correlation Matrix - relation among different toxic classes

Fig. 4. WordClouds for toxic and clean comments

The comments were human annotated using crowd sourcing methods. For our experiments, we used a random split of 70% of the labelled data for training, 10% for validation and 20% for testing.

A. Exploratory Data Analysis

We performed exploratory data analysis on our training set to get to know our text corpus better and understand the nature of toxic comments.

1) Comment Length: The average comment length was 395 words with a standard deviation of 594. The maximum length of a comment was 5895 words, indicating that comments in our corpus do not have a standardized length and can vary immensely.

2) Class Correlation: From the correlation matrix (Figure 3) of our toxic classes we find that there is only a 0.31 overlap between 'toxic' and 'severe toxic' indicating that there is a graduation between the categories. Comments tagged as 'insult' and 'obscene' have a correlation of 0.74 which intuitively makes sense, as insults tend to be obscene in nature.

3) Class Co-occurrence: From the co-occurrence matrix of our classes we find that all comments tagged as 'severe toxic' are tagged with 'toxic' 100% of the time. Likewise, 'threat' comments are toxic 95% of the time, and 'obscene' comments are toxic 94% of the time.

4) Class Frequency: The dataset contains many negative samples. On plotting the class frequencies as shown in Figure 1, we see that non-toxic comments dominate the corpus, with 80% of the data being non-toxic. This makes sense since most comments online are not necessarily toxic. As shown in Figure 2, within the toxic classes, the 'toxic' class dominates the dataset with 44% of the data belonging to the 'toxic' category, followed by approximately 24% each in 'obscene' and 'insult'. These observations are in line with our observations from the class co-occurrence matrix. Additionally, we note that there are only 404 samples belonging to the 'threat' class and approximately 1000 samples in 'severe toxic' and 'identity hate'. These three classes combined add up to only 10% of the whole training corpus.

5) Term Frequency: We used the Term Frequency-Inverse Document Frequency (TF-IDF) statistic in order to determine the most frequent and important terms within toxic classes as shown in Figure 5. We observe that within each toxic class, the most frequent term has almost double the TF-IDF score of the second-most important term. Out of the top five terms per class, two of the terms occur in all six toxic classes, and the same two terms are also the two most frequent terms in four of the classes. This shows that not only do toxic comments from our corpus contain certain terms almost doubly as often than other terms, but also contain the same frequent words across classes. Additionally, we generated wordClouds to visualize the word distribution among our dataset. These clouds give greater prominence to words that appear more frequently in the source text. Figure 4 shows the words appearing in toxic and clean comments.

B. Dataset Issues

From our exploratory analysis we note various interesting observations about our dataset. The dataset is not necessarily an ideal one and it has many implicit biases and issues. Out of these, the most important to note are those that indicate that our dataset is highly imbalanced.

The class distribution is imbalanced due to the high frequency of negative samples (non toxic comments) and extremely low frequency of certain classes. Additionally, an average of 95% of comments labelled as 'toxic' had an additional class label attached to it.

The term distribution within the toxic comments is imbalanced as well. First, there is a very high frequency of the same few words across all toxic categories. Second, within each specific toxic category the most frequent words have almost double the TF-IDF score of the second-most frequent word. Out of the top ten most frequent words, three of the words are in the top ten for all six classes, and four of the words are in the top ten for five of the classes.

This imbalance is concerning as it may lead to overfitting of any model which could just be highly sensitive to the most frequent words. Furthermore, if our model predicts that a comment belongs to any of the 5 classes other than 'toxic', automatically predicting that the comment is 'toxic' as well would be a correct guess 95% of the time, due to the way the dataset is labelled.

Another point of concern is that some comments are misclassified. For example, an ASCII drawing of a vulgar hand symbol is not labelled with any of the toxic categories. This could be in part due to how the human annotators were shown the comment during the labelling process. Although we do not have an exact count of how many such comments exist in the corpus, we estimate that there would be at least 10-20% of samples with at least one label misclassified.

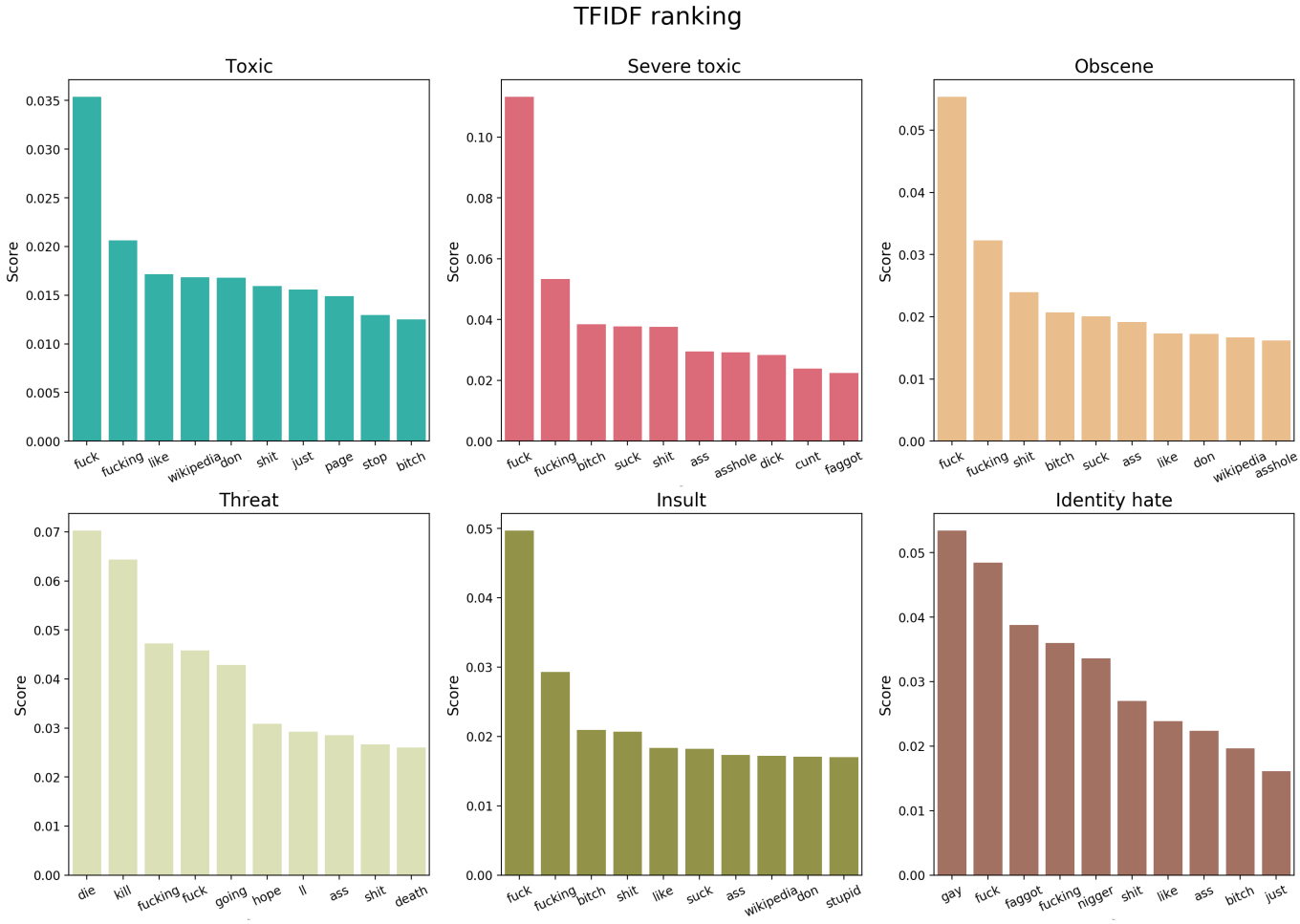

Fig. 5. Most frequent and important words within toxic classes

V. Methodology

A. Data Pre-Processing

Before applying any of the text classification methods, we performed a few pre-processing steps on the dataset. After applying standard data cleaning methods, such as making characters lower-case, removing stop words, we use techniques as introduced in [24] to make our model robust towards obfuscation attacks. Using *!d10ts* instead of *idiots* in a comment is an example of obfuscation attack. To protect our model against such type of adversarial attacks, we are using a multi-step approach.

1) We use a dataset generated by [24] containing 1000 variations of each common toxic word. First, we perform the token look up in this data. This dataset is generated by substituting characters with homoglyphs or by repeating the letters as in [25].
2) In case no match is found in Step 1, we perform regex matching for known patterns of common toxic words to flag the toxic word.

B. NB-SVM

We explored the Naive Bayes - Support Vector Machine (NB-SVM) approach as introduced in [26] along with the traditional Bag of Words (BoW) model for extracting features. This representation discards any information about the order or the structure of the words in the document. We built our Document Term Matrix using three feature extraction strategies — word n-grams, character n-grams and combination of word-character n-grams. This matrix included the TF-IDF scores of all the features. The character level n-grams features performed better than word n-grams because of their ability to handle variations in spelling which is often seen in the abusive speech. For comments such as *"ki11 yrslef a$$hole"*, word n-grams fails to flag these comments as toxic in contrast to character level methods.

C. Neural Networks

We experimented with an array of neural network architectures for toxic speech classification. One of the first choices was to use RNN's which gives great performance on the sequential data. However, as noted in [27], RNNs suffer from the problem of vanishing and exploding gradients and hence fail to keep track of long range dependencies. To address this issue, we investigated GRU [28] and LSTM [29] based models, which are variations of RNN using "gating" approach. An LSTM block is made of three gates — *input, output and forget*

TABLE I
RESULTS OF NAIVE BAYES, RNNS AND COMBINATION OF RNN AND CNN MODELS

Model	*AUC*	*Log Loss*
NB-SVM		
Word N-grams	0.9510	0.0710
Char N-grams	0.9858	0.0476
Char + Word N-grams	0.9838	0.0493
RNN		
LSTM	0.9791	0.0481
Bi-LSTM	0.9790	0.0470
Bi-GRU + FastText	0.9878	0.0419
Bi-GRU + FastText + Conv	0.9856	0.0440
Bi-LSTM + FastText + Conv	0.9870	0.0429

gate, which help it to remember a value for an arbitrary long time; forget the value when it is not important to remember anymore or output the value. A GRU cell on the other hand is a simplified version of LSTM which combines forget and input gates into a single *update gate* and has an additional *reset gate*.

For both GRU and LSTM based models, we experimented with simple and their bidirectional version to examine different the behavior of our model. Along with the LSTM/GRU layers, we added dropout and max pooling layers to our model architecture as well. Additionally, We tried adding convolutional layer to the existing RNN based model and evaluated the model performance.

In both RNN and CNN based models, we included an embedding layer. We investigated performance of random word embeddings and pretrained FastText [30] word models. For our experiments, we used 300 dimensional word embeddings pretrained on the Common Crawl. There are other popular pretrained word embeddings like word2vec, GloVe. However our choice of FastText word embeddings was motivated by one of the major differences in FastText and GloVe/word2vec word embeddings — while the former treats each word as composed of character n-grams, latter models treat each word in corpus like an atomic entity.

VI. RESULT

In Table 1, we present the results of evaluation of the different approaches used for toxic comment classification. In Naive Bayes - Support Vector Machine approach, we created a bag of words representation as a term document matrix based on TF-IDF. We found that tokenizing words based on TF-IDF leads to better results. We compared three different feature extraction techniques used for building term document matrix as shown in Table 1. As the results show, char N-grams outperformed both Wwrd N-grams and combination of char and word N-grams by achieving an accuracy of 98.5%. Therefore, we can conclude that character-level information is more useful than word-level information to distinguish between clean and toxic text. This conclusion is in compliance with the results proposed by [31] in which they tried to evaluate the performance of words and character n-grams to discriminate between spam and sincere messages.

In an effort to improve the toxic comment accuracy, we exploited different variations of recurrent neural network models. First, we found that the Bidirectional models tend to work better than the base models as they retain the information about the past and the future whereas unidirectional models only preserve the past information. This difference comes from the underlying architecture of these two models. Bidirectional models are designed in such a way that they are able to preserve information in any point of time with the help of combination of two hidden states.

As shown in table 1, Bidirectional GRU along with FastText performed better than all other models. In this model, unlike LSTM and Bi-LSTM where we used random word embeddings layer, we used diverse pretrained word embeddings. Using pretrained embedding helps capturing hidden information about words more effectively and accurately as word vectors are learned on different sources and hence, it helps boosting the performance of the classifier which is quite evident from our results. Inspired by the NB-SVM model results, we decided to use FastText out of other options such as GloVe, word2vec because FastText considers each word a combination of character ngrams. Also, it has the ability to handle rare and out of vocabulary words.

Another finding worth mentioning is that even though the NB-SVM and the neural network models have comparable results, handcrafted model (NB-SVM) takes significantly less time for training than the deep learning models. This is self-explanatory because in neural networks, we randomly initialize the feature weights and the network learn these weights while minimizing the error using optimization algorithms such as gradient descent. It could take a while before these weights start to converge and the model training gets terminated. Similarity, GRU and LSTM have very similar results, but LSTM takes significantly large time for model training because of its complex structure as compared to GRU model.

To handle adversarial attacks, we made our preprocessing step more robust. Having added some variations to the data cleaning part made it easier for our model to detect the incorporated punctuation marks or other special characters in the text which might be toxic in nature. By using these integrated techniques for tackling special kinds of attacks, our model can handle obfuscation and a known list of homoglyph substitution successfully which would not have been possible otherwise. For example, our model can correctly recognize the toxic text like ”What an idiiiiiiiiiot” and would be able to interpret it as it was like ”What an idiot”.

VII. CONCLUSION AND FUTURE WORK

With the increasing use of social media in our daily lives, a sophisticated tool for toxic comment detection is the need of the hour. Building such a model will help in creating a respectful environment for online discussions. Despite the fact that multiple researchers have worked and still are working on this problem, we are still long way to go to develop an intelligent tool which can deal with all kind of adversarial attacks.

Despite all of our models achieving state of the art theoretical performance in terms of AUC, we are not yet convinced that our models are practically effective. As stated earlier, our dataset is inherently imbalanced towards certain words, and classes. This could be a major factor for our models to have overfitted on the sample data. We leave it for future work to perform train-test augmentation of the dataset, and dealing with the various dataset specific biases, to make the results more realistic and universally applicable.

In the future work, we are planning to perform train/test data augmentation by converting the English sentences present in the dataset to a foreign language and then converting it back to English. Moreover, most of the existing models are based on English langauge. We plan to extend make this model language independent by training the word embeddings instead of using pre-trained embeddings. We also want to include comment's context, past behaviour of commentator, and comment thread history as additional features in our models and evaluate their performance. In addition to that we want to investigate the performance of attention based neural network models [20] for this problem.

REFERENCES

[1] Maeve Duggan. 5 facts about online harassment, 2017.

[2] David D. Luxton, Jennifer D June, and J. Miller . L. Fairall. Social media and suicide: a public health perspective. *American journal of public health*, 102 Suppl 2:S195–200, 2012.

[3] William Warner and Julia Hirschberg. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, LSM '12, pages 19–26, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[4] Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. Detecting offensive language in social media to protect adolescent online safety. In *Proceedings of the 2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust*, SOCIALCOM-PASSAT '12, pages 71–80, Washington, DC, USA, 2012. IEEE Computer Society.

[5] Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 1980–1984, New York, NY, USA, 2012. ACM.

[6] Sara Sood, Judd Antin, and Elizabeth Churchill. Profanity use in online communities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1481–1490, New York, NY, USA, 2012. ACM.

[7] Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 145–153, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[8] Sara Owsley Sood, Judd Antin, and Elizabeth F. Churchill. Using crowdsourcing to improve profanity detection. In *AAAI Spring Symposium: Wisdom of the Crowd*, volume SS-12-06 of *AAAI Technical Report*. AAAI, 2012.

[9] Dawei Yin, Brian D. Davison, Zhenzhen Xue, Liangjie Hong, April Kontostathis, and Lynne Edwards. Detection of harassment on web 2.0, 2009.

[10] Haoti Zhong, Hao Li, Anna Squicciarini, Sarah Rajtmajer, Christopher Griffin, David Miller, and Cornelia Caragea. Content-driven detection of cyberbullying on the instagram social network. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 3952–3958. AAAI Press, 2016.

[11] Cynthia Van Hee, Els Lefever, Ben Verhoeven, Julie Mennes, Bart Desmet, Guy De Pauw, Walter Daelemans, and Véronique Hoste. Detection and fine-grained classification of cyberbullying events. In *Proceedings of the 10th Recent Advances in Natural Language Processing (RANLP 2015)*, Hissar, Bulgaria, 09/2015 2015.

[12] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. Ex machina: Personal attacks seen at scale. *CoRR*, abs/1610.08914, 2016.

[13] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[14] Xingxing Zhang and Mirella Lapata. Chinese poetry generation with recurrent neural networks. In *EMNLP*, 2014.

[15] Minlie Huang, Yujie Cao, and Chao Dong. Modeling rich contexts for sentiment classification with LSTM. *CoRR*, abs/1605.01478, 2016.

[16] Xin Wang, Yuanchao Liu, Chengjie Sun, Baoxun Wang, and Xiaolong Wang. Predicting polarities of tweets by composing word embeddings with long short-term memory. In *ACL*, 2015.

[17] Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. *CoRR*, abs/1611.06639, 2016.

[18] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626, 2015.

[19] Cícero Nogueira dos Santos and Maíra A. de C. Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, 2014.

[20] Zu-Fan Zhang, Yang Zou, and Chenquan Gan. Textual sentiment analysis via three different attention convolutional neural networks and cross-modality consistent regression. *Neurocomputing*, 275:1407–1415, 2018.

[21] Spiros V. Georgakopoulos, Sotiris K. Tasoulis, Aristidis G. Vrahatis, and Vassilis P. Plagianakos. Convolutional neural networks for toxic comment classification. *CoRR*, abs/1802.09957, 2018.

[22] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2267–2273. AAAI Press, 2015.

[23] Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 29–30, New York, NY, USA, 2015. ACM.

[24] Nestor Rodriguez and Sergio Rojas Galeano. Shielding google's language toxicity model against adversarial attacks. *CoRR*, abs/1801.01828, 2018.

[25] Sergio Rojas-Galeano. On obstructing obscenity obfuscation. *ACM Trans. Web*, 11(2):12:1–12:24, April 2017.

[26] Sida Wang and Christopher D. Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 90–94, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[27] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *Trans. Neur. Netw.*, 5(2):157–166, March 1994.

[28] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[29] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[30] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[31] I Kanaris, K Kanaris, I Houvardas, and E Stamatatos. Words versus character n-grams for anti-spam filtering. *International Journal on Artificial Intelligence Tools*, page 10471067, 2007.