Toxicity Detection: An Analysis of Lexical and Machine Learning Approaches

Sabrina Knappe
McGill University
260728227
sabrina.knappe@mail.mcgill.ca

Beatrice Lopez
McGill University
260654565
beatrice.lopez@mail.mcgill.ca

Jizhou Wang
McGill University
260518646
jizhou.wang@mail.mcgill.ca

Abstract

Toxicity detection has become a relevant topic with the increasing ubiquity of social networks, where toxic and hateful comments are rampant. In this paper, we explore the task of toxicity detection and compare lexical and machine learning approaches by implementing recurrent and convolutional neural networks and SO-CAL (a dictionary-based method). We use the Civil Comments dataset from the Jigsaw Unintended Bias in Toxicity Classification Kaggle competition to train the neural networks and evaluate them both with a ROC-AUC bias metric. In this paper, we analyse the contrast in performance as well as the portability and interpretability of these methods while taking in consideration of the phenomenon of unintended bias in the task of toxicity detection.

1 Introduction

This project concerns the topic of toxicity detection - arguably a task under the more general one of sentiment analysis. Toxicity detection is a helpful tool in online environments, where there are high levels of abuse, harassment, and offensive hate speech which users may wish to block or filter. Toxicity detection can be understood as the task of classifying extremely negative sentiment. There have been various methodologies and models introduced to solve this task and in this project, we desire to compare two particular methods: machine learning and lexicon sentiment extractors. In this project, we hope to compare the performance of these two methods. We hypothesize that the lexical approach allows more portability (i.e. generalizes better to different domains and contexts), more interpretability, as well as avoiding unintended bias that currently exists in machine learning models.

Significant work has already been done on the topic of Toxicity Detection, given its relevance to social media networks where toxic comments, although often unwanted, are commonplace. Toxicity detection has been an appealing problem to those developing machine learning techniques, which have been frequently used to detect toxicity online.

In general, neural networks requires a large amount of data samples for training. Applying such models to different domains requires new training data which leads to the issue of portability. There is also a lack of transparency with neural networks in a sense that it is difficult to identify the errors and biases such as false positives acquired by trained models since the hidden layers of deep neural networks cannot be easily interpreted. Machine learning methods implemented without proper tuning can also exhibit errors from overfitting. In many online environments, this could mean that an association is learned between words denoting identity and toxicity (Hutchinson et al., 2019). Identity words such as ”gay”, ”black”, and ”muslim” are often used in comments meant to attack and demean. Because of this connection, machine learning models may associate identity words themselves with toxic meaning and wrongly classify innocuous comments such as ”Gay rights are important” as toxic.

We hypothesize that lexicon-based methods avoid issues in difficulty in lack of transparency, portability and unintended bias that currently exist in machine learning models. In this approach, the lexicon model makes use of a dictionary consisting of the text's individual words with sentiment scores assigned to each word according to its polarity. Through a lexical model's dictionary, one can easily apply the model to another domain or context by simply updating the dictionary with the vocabulary of that domain as well as avoid the unintended bias. We chose to compare lexical and

machine learning approaches in toxicity detection, specifically Tabaoda et al.'s Semantic Orientation Calculator (SO-CAL), introduced in Lexicon-Based Methods for Sentiment Analysis (Taboada et al., 2011), as well as long short-term memory (LSTM)(Hochreiter and Schmidhuber, 1997) and convolutional neural network (CNN) methods, which were some of the top performing models in the Kaggle competition for decreasing bias. We used and modified these methods in order to compare their efficacy in accurately detecting toxicity.

2 Related Work

A fair amount of previous work has been done on the topic of alleviating bias in toxicity detection. The paper Unintended Machine Learning Biases as Social Barriers for Persons with Disabilities (Hutchinson et al., 2019) explores the bias of machine learning algorithms when classifying the toxicity of sentences containing words related to disability. They find that "the data used to develop such models reflects topical biases in social discourse which may explain such biases in the models." This means that the negative way in which certain topics are talked about may lead to an association with an actually neutral topic with toxicity. This result led us to form our hypothesis that a lexical-based model would better avoid suffering from this unintended bias. We also consulted Toxic Comment Classification and Unintended Bias(Hamida et al., 2019), which details how convolutional neural networks (Kim, 2014) can be adapted to improve accuracy and reduce overfit. Hamida et al. compared performance between Convolutional Neural Networks and baseline methods like Naive Bayes and Logistic Regression. Their work was done on the same dataset we used for our analyses, and so provides a useful comparison with our own results. Similarly, the paper Challenges for Toxic Comment Classification: An In-Depth Error Analysis (van Aken et al., 2018) looks at how accurately various models classify toxicity. They explore both typical methods such as LSTM and CNN and an ensemble method. Van Aken et al. also explored various types of errors that occur and what methods tend to cause them. This helped us shape our hypothesis as to what errors our methods might be prone to. In their paper Sentiment analysis of player chat messaging in the video game StarCraft 2: Extending a lexicon-based model (Thompson et al., 2017) SO-CAL is used on StarCraft 2 chat messages in order to see how it can be extended to more specific domains. Their work found that the extension of SO-CAL to a more specified domain can be done "with modest effort." This paper helped guide our implemenation and modification of SO-CAL.

3 Dataset and Setup

3.1 Preprocessing

We used the Civil Comments Dataset (Research Data, 2018) from the Jigsaw Unintended Bias in Toxicity Classification Kaggle competition for training and testing. The training set consist of 1,804,874 data samples and the test set consist of 97,320 data samples. We've selected 2,000 samples from the test set balanced with toxic and non-toxic phrases from each subgroup for final evaluation. Exploratory Data Analysis was conducted to determine words that are needed for normalization and cleaning, especially words with toxicity as many of them are censored with asterisks. We created a normalized version of the data, making texts as more suitable inputs into our models. Because the data are composed of online comments, they have quirks such as emojis, irregular spelling of words, abbreviations, contractions, foreign/irregular/Unicode characters and words that are not used in a formal offline context.

Some of the normalizing includes replacement of characters and words into a common English alphabetic/dictionary form. For example, "censored" versions of profane words with the proper English spelling of the word themselves e.g., "sh*t" becomes "shit". There is also considerable correction of misspellings, the removal of special characters in favor of spaces and Language specific characters are set to the same character as we're not considering foreign language as a part of our model prediction task. Emojis are translated into English aliases using a GitHub package. (Kim and Wurster, 2014) Hashtags, User, Url, Email and numbers such as phone numbers, dates, time and money are normalized and segmented using a Social Network preprocessor called Ekphrasis on GitHub. (Baziotis et al., 2017)

This normalized data is used in both the lexical and neural network approach. For SO-CAL, no training is required and so only the normalized dataset of 2,000 samples was used for evalu-

ation. Further preprocessing in SO-CAL includes tokenization, sentence splitting, and POS tagging.

3.2 Evaluation Metrics

Our aim in this paper is to compare the performance of lexical and machine learning models in toxicity detection tasks also noting the issue of unintended bias. To this aim, we use the metrics used in the Kaggle competition to assess their accuracy in toxicity detection tasks. The AUC-ROC curve is a useful tool for measuring bias(Fawcett, 2006). It compares the number of true positives and false positives. This is useful when evaluating toxicity detection because of solutions to the problem are often prone to false positives (Thompson et al., 2017). Kaggle sets up three types of AUC measures: Subgroup AUC measures the ability of the model to differentiate between toxic and non-toxic comments that mention an identity, BPSN (Background Positive, Subgroup Negative) AUC measures the ability of the model to distinguish between non-toxic examples that mention the identity with toxic examples that do not, and finally BNSP (Background Negative, Subgroup Positive) AUC measures the ability of the model to distinguish between toxic examples that mention the identity and non-toxic examples that do not. The lower these AUC scores are, the less the neural networks are overfitting the data. These AUC scores are combined into one overall AUC. We then use this measure to analyze how well each model assesses toxicity.

4 Proposed Approach

Outlined below are the models that we used to identify toxic comments.

4.1 LSTM and CNN

For our neural networks, word embedding matrices were used as inputs. To build these word embedding matrices, a dictionary of words is defined based on all the inputs in the training and test set. Then each of these words are mapped to pre-trained word embeddings used as inputs into the neural network models. Normalization was a necessary preprocessing step to maximize the overlap of words in the dictionary and words in the pre-trained word embeddings. We have tried combinations of Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) embeddings. Neural networks also require inputs phrase of same embedding length therefore we have pre-padded each phrase to a length of 400 words embeddings.

LSTM is a Recurrent Neural Network (Mikolov et al., 2010) that tries to tackle the fundamental problem of vanish and exploding gradient(Bengio et al., 1994) when training neural networks. We used two bidirectional-LSTM layers with 128 and 256 hidden units respectively and concatenated their pooled outputs into a final linear hidden layer.

CNN is most commonly used for image classification tasks such as ImageNet.(Krizhevsky et al., 2012) Although images may have many local features that determine its characteristics and overall class, one can comparably argue that phrases pertain such local structure analogous to n-gram models. We used three Convolution layers of 3x3, 4x4 and 5x5 kernel sizes with 100 filters each. The outputs of these layers are batch normalized(Ioffe and Szegedy, 2015) and pooled. A dropout of 0.5 is then applied before feeding into a final linear hidden layer.

Spacial dropouts(Srivastava et al., 2014) is applied to both neural network models at the embedding level. Binary Cross Entropy loss, Adam Optimizer(Kingma and Ba, 2014) and an adaptive scheduler for learning rate starting at 0.001 was applied during training. These neural networks also trained based on values of other auxiliary columns such as 'severe toxicity', 'obscene', 'insult' provided by the data set to predict the main column 'toxicity'. The output probability of the 'toxicity' column is calculated with a softmax function from the final linear hidden layer.

We've decided to combine these neural network structures along with a combination of word embeddings matrices from normalized and non-normalized data to form baseline models for testing. Given the time constrain of this project, the models are setup such that their total training runtime are similar in length. The model selected for evaluation is the model with the lowest loss during validation phase.

4.2 SO-CAL

The general approach of lexical methods in sentiment analysis is to derive sentiment scores of text from the sentiment scores of the individual words in the text. Dictionaries that document the sentiment scores of words are implemented and then a method is developed to aggregate the scores of the

words (Taboada and Stede, 2011).

SO-CAL implements five dictionaries for adjectives, nouns, adverbs, and intensifiers that bear sentiment. Sentiment scores for each word are rated on a scale from -5 to 5. Words that do not bear sentiment are not stored in the dictionaries and are automatically assigned a neutral sentiment score of 0. SO-CAL parses the text for sentiment-bearing words, retrieves their sentiment scores, and then calculates the text's sentiment score. SO-CAL also analyses the syntactic structure of the text and adjusts sentiment scores where needed. For example, the model takes notes of intensifiers and negators that that change the polarity of words.

The model was evaluated on the normalized dataset of 2000. It was run on various cut-off sentiment score thresholds varying from -4 to 0 to and the results are analyzed below. We manually looked through the dictionaries implemented to ensure that none of the subgroup identities are assigned undue negative scores and causing unintended bias.

5 Results

5.1 Word Embeddings Coverage

Tables 1 and 2 (5) summarizes the results of the embeddings vocabulary coverage for the machine learning methods from the original and normalized version of the dataset. Normalizing the dataset allowed us to drastically increase the amount of found vocabulary. This improved our model runtime, size, and ROC-AUC overall score.

5.2 Model parameters and Runtimes

The total runtime for training the models were equal in terms on epochs trained. As shown in Table 3, LSTM has many more parameters to train on compared to CNN and more than twice the runtime per epoch. Adding an additional embedding increased the number of trainable parameters by more than 300k and runtime per epoch by more than 100 seconds in LSTM and more than 200 seconds in CNN.

5.3 LSTM vs CNN, Normalized vs Original, Single vs Ensemble Embeddings

As shown in Table 3 and 4, LSTM did well above a random baseline of 0.5 in terms of accuracy and overall AUC. CNN doesn't fair too well in terms of accuracy at around 0.57. The normalized and ensemble embedding in both neural network models display an advantage in overall AUC and loss. Interestingly, LSTM with only 'word2vec' embeddings has a higher accuracy compared to LSTM combined with 'glove' embeddings. This may be because of the total runtime limit set as models with more parameters are able to increase accuracy further given more runtime.

The best accuracy and loss were achieved around the fifth iteration, meaning that many epochs were needed in order to achieve optimal results.

5.4 SO-CAL

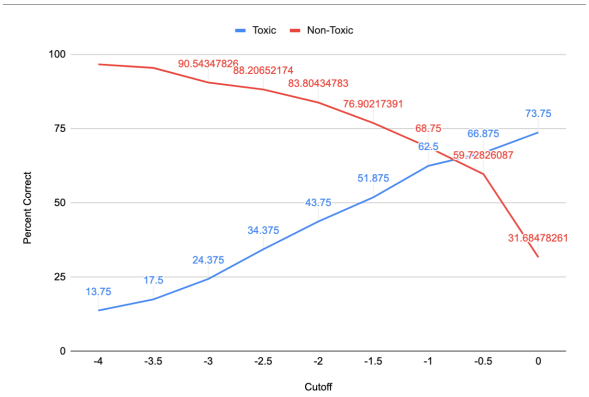

Figure 1: Percent correct by cutoff for SO-CAL

We found that SO-CAL's results were a tradeoff between percent correct for toxic examples and percent correct for non-toxic examples. There is a sweet spot at the cutoff of -1 where the results for the two are about balanced, with the toxic comments being correctly identified 62.5% of the time and the non-toxic comments being correctly identified 68.75% of the time (see Table 5) and an overall accuracy of 68.25%, higher than the CNN.

6 Discussion and Conclusions

The experiments and results of this project have amply allowed for the exploration of two different approaches to toxicity detection as well as analysis of the unintended bias effect. Both our ma-

Embeddings	Found Vocab	Found Texts	Embedding size
word2vec (w2v)	0.4374	0.8619	300
glove (g)	0.591	0.9923	300

Table 1: Summary of embedding coverage for the original dataset

Embeddings	Found Vocab	Found Texts	Embedding size
word2vec (w2v)	0.7068	0.8628	300
glove (g)	0.7765	0.9947	300

Table 2: Summary of embedding coverage for normalized dataset

Models	Accuracy	Loss	Runtime(s/epoch)	Overall AUC	Trainable Parameters	Size (MB)
LSTM (w2v)	0.9547	0.2827	1092	0.9162	1,364,487	415.4
LSTM (w2v, g)	0.9328	0.2751	1251	0.9206	1,671,687	818.7
CNN (w2v)	0.5754	0.5401	349.5	0.8778	543,207	406
CNN (w2v, g)	0.5779	0.5287	601.9	0.8855	903,207	809.9

Table 3: The results of machine learning techniques on normalized data.

Models	Accuracy	Loss	Runtime(s/epoch)	Overall AUC	Trainable Parameters	Size (MB)
LSTM (w2v)	0.8037	0.2789	1134	0.9144	1,364,487	688.9
LSTM (w2v, g)	0.7998	0.2754	1255	0.9191	1,671,687	1339
CNN (w2v)	0.5611	0.5412	351.1	0.8721	543,207	679.5
CNN (w2v, g)	0.5828	0.499	601	0.8841	903,207	1330

Table 4: The results of machine learning techniques on the original data.

Actual/Predicted	Toxic	Non-toxic
Toxic	62.5%	31.25%
Non-Toxic	37.5%	68.75%

Table 5: Confusion matrix for SO-CAL with a cutoff sentiment score of -1

chine learning and lexical methods achieve significant accuracy given the distinct nature of the approaches. Notably, our machine learning models do well in minimizing the unintended bias effect while still achieving strong accuracy and SO-CAL avoids unintended bias through its customized dictionaries, but also achieves strong accuracy given that the model does not require training data and completes the task in minutes compared to hours. We note that for the task of toxicity detection it is practically crucial to minimize the amount of false positives (i.e. it is more important that non-toxic comments or posts are not incorrectly labeled as toxic).

While our CNN and LSTM models perform very well, there is always opportunity for improvement. A future experiment might implement more recently explored models such as Google's BERT (Devlin et al., 2018), which trains on a fuller context, being bidirectional and unsupervised.

In conclusion, we have well-performing machine learning models that minimize unintended bias and we have a strong lexical approach that allows for portability, interpretability, and simple computational power. As shown by Thompson et al. (Thompson et al., 2017), SO-CAL can easily be transferred to new domains simply by implementing dictionaries for those domains.

7 Statement of Contributions

Jizhou Wang was responsible for the development and evaluation of the neural networks as well as preprocessing of text. Beatrice Lopez was responsible for the implementation and evaluation of SO-CAL. Sabrina Knappe assisted with research and reading of related work, the evaluation of SO-CAL, as well as took the lead with the writing of this paper. All team members contributed to the paper.

References

Betty van Aken, Julian Risch, Ralf Krestel, and Alexander Löser. 2018. Challenges for toxic comment classification: An in-depth error analysis. *CoRR*, abs/1809.07572.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tom Fawcett. 2006. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874.

Chady Ben Hamida, Victoria Ge, and Nolan Miranda. 2019. Toxic comment classification and unintended bias.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ben Hutchinson, Vinodkumar Prabhakaran, Emily Denton, Kellie Webster, Yu Zhong, and Stephen Craig Denuyl. 2019. Unintended machine learning biases as social barriers for persons with disabilities.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Taehoon Kim and Kevin Wurster. 2014. Emoji. https://github.com/carpedm20/emoji.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

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

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Civil Research Data. 2018. data.json.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Julian Brooke Milan Tofiloski Kimberly Voll Taboada, Maite and Manfred Stede. 2011. So-cal. https://github.com/sfu-discourse-lab/SO-CAL.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Comput. Linguist.*, 37(2):267–307.

Joseph J Thompson, Betty HM Leung, Mark R Blair, and Maite Taboada. 2017. Sentiment analysis of player chat messaging in the video game starcraft 2: Extending a lexicon-based model. *Knowledge-Based Systems*, 137:149 – 162.

A Links to Code

- https://colab.research.google.com/github/imbealopez/Toxicity-Detection/blob/master/ToxicityDetection.ipynb
- https://github.com/imbealopez/Toxicity-Detection