A Review of Standard Text Classification Practices for Multi-label Toxicity Identification of Online Content

Isuru Gunasekara
IMRSV Data Labs
Ottawa, Canada
isuru@imrsv.ai

Isar Nejadgholi
IMRSV Data Labs
Ottawa, Canada
isar@imrsv.ai

Abstract

Language toxicity identification presents a gray area in the ethical debate surrounding freedom of speech and censorship. Today's social media landscape is littered with unfiltered content that can be anywhere from slightly abusive to hate inducing. In response, we focused on training a multi-label classifier to detect both the type and level of toxicity in online content. This content is typically colloquial and conversational in style. Its classification therefore requires huge amounts of annotated data due to its variability and inconsistency. We compare standard methods of text classification in this task. A conventional one-vs-rest SVM classifier with character and word level frequency-based representation of text reaches 0.9763 ROC AUC score. We demonstrated that leveraging more advanced technologies such as word embeddings, recurrent neural networks, attention mechanism, stacking of classifiers and semi-supervised training can improve the ROC AUC score of classification to 0.9862. We suggest that in order to choose the right model one has to consider the accuracy of models as well as inference complexity based on the application.

1 Introduction

While the sheer volume of online content presents a major challenge in information management, we are equally plagued by our current inability to effectively monitor its contents. In particular, social media platforms are ridden with verbal abuse, giving way to an increasingly unsafe and highly offensive online environment. With the threat of sanctions and user turnover, governments and social media platforms currently have huge incentives to create systems that accurately detect and remove abusive content.

When considering possible solutions, the binary classification of online data, as simply toxic and non-toxic content, can be very problematic. Even with very low error rates of misclassification, the removal of said flagged conversations can impact a user's reputation or freedom of speech. Developing classifiers that can flag the type and likelihood of toxic content is a far better approach. It empowers users and online platforms to control their content based on provided metrics and calculated thresholds.

While a multi-label classifier would yield a more powerful application, it's also a considerably more challenging natural language processing problem. Online conversational text contains shortenings, abbreviations, spelling mistakes, and ever evolving slang. Huge annotated datasets are needed so that the models can learn all this variability across communities and online platforms (Chandrasekharan et al., 2017). Furthermore, building a representative and high volume annotated dataset of social media contents for multiple types of toxicity can be exhaustive. It is a subjective, disturbing and time consuming task. Critical consideration of the relationships between different subtasks is needed to label this data (Waseem et al., 2017). Additionally, the annotated datasets will always be unbalanced since some types of toxic content are much more prevalent than others.

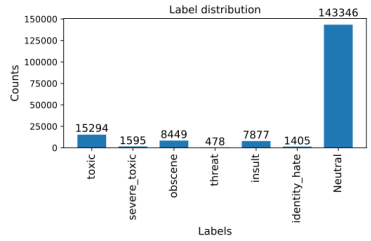
Some of the back-end approaches to this task have been well researched. Hand-authoring syntactic rules can be leveraged to detect offensive content and identify potential offensive users on social media (Chen et al., 2012). Also, morphological, syntactic and user behavior level features have been shown to be useful in learning abusive behavior

Proceedings of the Second Workshop on Abusive Language Online (ALW2), pages 21–25
Brussels, Belgium, October 31, 2018.

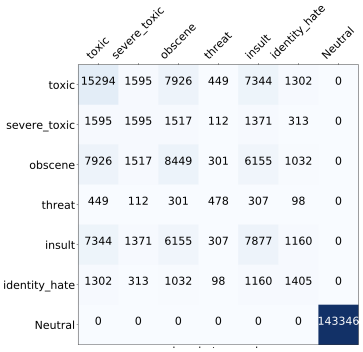
(Papegnies et al., 2017; Buckels et al., 2014; Yin et al., 2009; Chen et al., 2012). Conventional machine learning classifiers such as SVM classifiers (Nobata et al., 2016) and linear regressions models (Davidson et al., 2017; Xiang et al., 2012) have also been used to effectively detect abusive online language. Deep learning models with word embeddings as text representations are state-of-the-art text classification solutions that show effectiveness in many tasks such as sentiment analysis and the detection of hate speech (Gambäck and Sikdar, 2017). Although all these methods are well studied and established, it is not always clear what the best choice for a specific task is due to the trade-off between acquired success rate of the classification model and the complexities of its deployment and inference.

In our work, we used the Wikimedia Toxicity dataset to investigate how various methods of designing a standard text classifier can impact the classification success rate as well as its inference cost. This dataset was published and used for a Kaggle competition. In the context of the competition, it is a common practice to train multiple large size models and ensemble them to get the highest results, tailored for the competition test set. Here, however, we only looked at standard classification models that are suitable to be deployed and used for inference in real-time. For text representations, we looked at frequency-based methods and multiple word embeddings. For classification models, we considered neural network models that can learn sentence representation using recurrent neural networks and attention layers. We also investigated stacking classifiers and used them to automatically label the unannotated part of the dataset to be added to the training set. This paper highlights how we compared various standard methods to help identify what the best practices for this application are.

The paper is organized as follows. In Section 2, we describe the dataset, annotation, cleaning and augmentation steps that we applied. In Section 3, we review some of the commonly used text representation methods and look at how representation of text can impact the classification results. In Section 4, we compare neural network models that are effective in learning long sequences. In Section 5, we investigate how stacking two classifiers can improve results. In Section 6, we investigate the impact of using automatically labeled datasets to further train the classifiers and Section 7 discusses our findings.



(a) counts of classes in annotated dataset



(b) overlap between class pairs

Figure 1: The counts and overlap of classes in training dataset

2 Dataset

In this work, we used Wikimedias Toxicity Data Set (Wulczyn et al., 2016b,a). This dataset is also available on Figshare https://figshare.com/articles/Wikipedia_Detox_Data/4054689 as the Wikipedia Human Annotations of Toxicity on Talk Pages and contains about 215K annotated examples from Wikipedia Talk pages. The dataset has been annotated by Kaggle based on asking 5000 crowd-workers to rate Wikipedia comments according to their toxicity (which they evaluated based on how likely they were to make others leave the conversation). The labels include seven types: neutral, toxic, severe toxic, obscene, threat, insult and identity hate. This dataset was published in two parts namely train and test set. The train set has 159571 annotated comments while the test set includes about 160k entries. However, only 63978 of test comments are identified as valid and annotated, which are used here as test set. There are more than 24 million words in this dataset yet the vocabulary size is only 495147. This is a very unbalanced dataset and a sample can get more than one label. Figure 1 shows

the count of multiple labels in the train set as well as the training labels' overlaps. For all the experiments the AUC score is calculated which is the area under the curve (true positive rate vs the false positive rate) is calculated for the test set as the evaluation metric.
All classes except for the non-toxic examples are augmented through translation to French, Dutch and Spanish before translating back to English. Using this method, we get slightly different sentences and the label is preserved. Punctuation was removed and a set of very common word variations (including abbreviations) on social media were found and replaced by the original word. This cleaning reduced the vocabulary from 495147 to 434161.

3 Text Representation

We investigated word tf-idf and character tf-idf as frequency-based text representations and compared them with representing text using average of word embeddings. For these experiments stop words are removed from text. Character level tf-idf is calculated for character n-grams where $n = 1,\ldots,6$. Word level tf-idf is calculated for word n-grams where $n = 1,2,3$. A fastText skip-gram model (Bojanowski et al., 2016) is trained to obtain 50D word embedding vectors for character level n-gram features where $n = 1,\ldots,6$ and word n-gram features where $n = 1,2,3$. We also used pre-trained word embeddings, including Glove (Pennington et al., 2014) and 300D fastText vectors. In order to evaluate the impact of text representation, we trained seven one-vs-rest SVM classifiers to predict the labels independently. Table 1 shows the results obtained from our experiments. Our results show that word level tf-idf fails to achieve accurate classification when the data is informal and conversational. However, if character level tf-idf is added to the representations, results will improve drastically. Training a specialized word embedding is not shown to be effective in our experiments. The low volume of the training set can be attributed to this observation. Pre-trained fastText is shown to slightly outperform Glove since it can assign vectors to every word while Glove discards the OOV words. Based on these results we chose to represent the text with pre-trained fastText embedding for the rest of the experiments.

Table 1: Comparison of different text representation methods in training one-vs-rest SVM classifiers

Representation	**AUC**
word tfidf	0.5423
char and word tfidf	0.9763
Average of 50D trained fasttext	0.8765
Average of Glove	0.9725
Average of 300D Pretrained fasttext	0.9782

4 Neural Network Classification Models

While word embeddings are a semantic representation of words, bidirectional neural networks are the technology known for generating a semantic representation for a given sequence of words. Bidirectional recurrent neural networks learn the meaning of a sentence not only from the individual words but by processing the dependencies of the surrounding words through forward and backward connections. Both bi-LSTM (Chen et al., 2016) and bi-GRU (Chung et al., 2015) architectures are shown to perform well in sentence representation. LSTM and GRU layers have a proficient learning ability for long text, because they can control how much information should be received in the current step, how much should be forgotten, and how much information should be passed back.
Attention layers (Parikh et al., 2016; Felbo et al., 2017) are mechanisms suitable for converting sequence representations, which are usually in the form of matrices, to a vector representation that is tailored for the desired classification tasks. We investigated the impact of leveraging these technologies by training and testing of two neural network structures shown in Figures 2a and b. Pre-trained fasttext embeddings are used and stop words are not removed, since we want the LSTM and attention layer learn the complete sequences. The neural network shown in Figure 2a which contains two layers of biLSTM to encode the information of sequences achieves 0.9842 and the one shown in Figure 2b which uses attention mechanism to combine the context information from embedding layer and the sequence information from each biLSTM layer to get a summary vector of the sentence, reaches 0.9844 in AUC.

5 Stacking of Classifiers

Stacking of classifiers is a standard way of increasing the accuracy of a classification task by combining the predictions of multiple classifiers to-

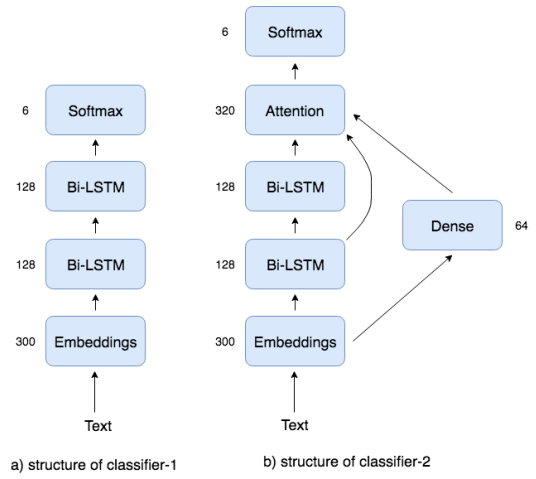


Figure 2: Structure of neural network classifiers trained and tested in this work

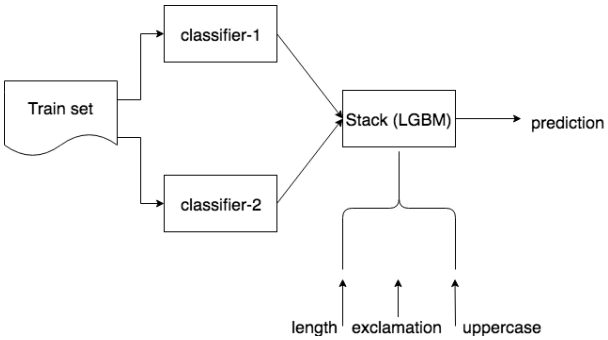


Figure 3: A schematic of applied stacking method

gether (Merz, 1999). In this method, a supervisor model is trained and learns how to combine the predictions of different types of models that differ in their variance, bias and capability of dealing with noise (Sluban and Lavrač, 2015). Figure 3 describes the stacking method applied in this work. We used a Light Gradient Boosting Machine (LGBM) stacking model which is a gradient boosting library implemented by Microsoft (Ke et al., 2017). LGBM is an implementation of fast gradient boosting on decision trees. Given a set of features, this classifier learns a linear combination of the predictions of preliminary classifiers to predict the label. The output of softmax layer from both classifiers (probabilities predicted for 6 classes) is fed to the LGBM. Also, the length of the text, frequency of exclamation marks and frequency of capital letters are considered as LGBM features. The LGBM classifier reached a 0.9847 score.

Table 2: comparison of different classification models

classifier	training	AUC
classifier-1	supervised	0.9842
classifier-2	supervised	0.9844
LGBM	supervised	0.9847
classifier-1	semi-supervised	0.9860
classifier-2	semi-supervised	0.9862

6 Semi-supervised Training

In this section, we investigate the impact of pseudo-labeling as a semi-supervised training method (Lee, 2013). Simply put, we split the test dataset into 10 folds. We then trained the two classifiers described in Section 4, in a supervised fashion, with both training set and 9 folds of test set. For test set, pseudo-labels are used which are the predictions calculated by the best classifier (the LGBM model) as if they were true labels. The trained classifier is tested on the 10th fold and the experiment is repeated for all 10 folds. This method has shown to be equivalent to entropy regularization (Grandvalet and Bengio, 2005) and makes up for dissimilarities of distributions between test and train dataset. Semi-supervised training of classifier-1 and classifier-2 improves the AUC score to 0.9860 and 0.9862 respectively.

7 Conclusion

Our investigation reveals that in the domain of conversational text, choosing the right text representation is crucial. Comparisons between multiple standard text representation techniques show that character-level representations outperform word-level representations in case of conversational text. Even with conventional SVM one-vs-rest classifiers, drastic improvement can be achieved when the text representation includes character level tfidf instead of only word level tfidf vectors (Table 1). We also showed that using various state-of-the-art classification techniques including sequence modeling neural network models, attention mechanisms and stacking of classifiers can slightly improve the AUC score of classification. Moreover, we demonstrated that further training of models through automatic labeling of unannotated datasets can improve the success rate of the classifier (Table 2). However, significance of these improvements depends on the application, inference cost and complexity and the amount of data that has to be processed during inference. Our

research gave life to a language toxicity identification tool, which will be presented alongside this paper.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Erin E Buckels, Paul D Trapnell, and Delroy L Paulhus. 2014. Trolls just want to have fun. *Personality and individual Differences*, 67:97–102.

Eshwar Chandrasekharan, Mattia Samory, Anirudh Srinivasan, and Eric Gilbert. 2017. The bag of communities: Identifying abusive behavior online with preexisting internet data. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 3175–3187. ACM.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. 2016. Enhancing and combining sequential and tree lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 71–80. IEEE.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2015. Gated feedback recurrent neural networks. In *International Conference on Machine Learning*, pages 2067–2075.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009*.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *arXiv preprint arXiv:1708.00524*.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90.

Yves Grandvalet and Yoshua Bengio. 2005. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154.

Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2.

Christopher J Merz. 1999. Using correspondence analysis to combine classifiers. *Machine Learning*, 36(1-2):33–58.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pages 145–153. International World Wide Web Conferences Steering Committee.

Etienne Papegnies, Vincent Labatut, Richard Dufour, and Georges Linares. 2017. Impact of content features for automatic online abuse detection. *arXiv preprint arXiv:1704.03289*.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

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.

Borut Sluban and Nada Lavrač. 2015. Relating ensemble diversity and performance: A study in class noise detection. *Neurocomputing*, 160:120–131.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: a typology of abusive language detection subtasks. *arXiv preprint arXiv:1705.09899*.

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

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2016b. Wikipedia detox. *figshare*.

Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. 2012. 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*, pages 1980–1984. ACM.

Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D Davison, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2. *Proceedings of the Content Analysis in the WEB*, 2:1–7.