Do Characters Abuse More Than Words?

Yashar Mehdad
Yahoo! Research
Sunnyvale, CA, USA
ymehdad@yahoo-inc.com

Joel Tetreault
Yahoo! Research
New York, NY, USA
tetreaul@gmail.com

Abstract

Although word and character n-grams have been used as features in different NLP applications, no systematic comparison or analysis has shown the power of character-based features for detecting *abusive language*. In this study, we investigate the effectiveness of such features for abusive language detection in user-generated online comments, and show that such methods outperform previous state-of-the-art approaches and other strong baselines.

1 Introduction

The rise of online communities over the last ten years, in various forms such as message boards, twitter, discussion forums, etc., have allowed people from disparate backgrounds to connect in a way that would not have been possible before. However, the ease of communication online has made it possible for both anonymous and non-anonymous posters to hurl insults, bully, and threaten through the use of profanity and hate speech, all of which can be framed as "abusive language." Although detection of some of the more straightforward examples of abusive language can be handled effectively through blacklists and regular expressions, as in "*I am surprised these fuckers reported on this crap*", more complex methods are required to address the more nuanced cases, as in "*Add anotherJEW fined a bi$$ion for stealing like a lil maggot. Hang thm all.*" In that example, there are tokenization and normalization issues, as well as a conscious bastardization of words in an effort to evade blacklists or to add color to the post.

While previous work for detecting abusive language has been dominated by lexical-based approaches, we claim that morphological features play a more significant role in this task. This is based on the observation that user language evolves either consciously or unconsciously based on standards and guidelines imposed by media companies that users must adhere to, in conjunction with regular expressions and blacklists, to catch bad language and consequently remove a post. Essentially, users learn over time not to use common lexical items and words to convey certain language. Thus, characters often play an important role in the comment language. Characters, in combination with words, can act as basic phonetic, morpho-lexical and semantic units in comments such as "*ki11 yrslef a$$hole*". Character n-grams have been proven useful for other NLP tasks such as authorship identification (Sapkota et al., 2015), native language identification (Tetreault et al., 2013) and machine translation (Nakov and Tiedemann, 2012), but surprisingly have not been the focus in prior work for abusive language.

In this paper, we investigate the role that character n-grams play in this task by exploring their use in two different algorithms. We compare their results to two state-of-the-art approaches by evaluating on a corpus of nearly 1M comments. Briefly, our contributions are summarized as follows: **1)** character n-grams outperform word n-grams in both algorithms, and **2)** the models proposed in this work outperform the previous state-of-the-art for this dataset.

2 Related Work

Prior work in abusive language has been rather diffuse as researchers have focused on different aspects ranging from profanity detection (Sood et al., 2012) to hate speech detection (Warner and Hirschberg, 2012) to cyberbullying (Dadvar et al., 2013) and to abusive language in general (Chen et al., 2012; Djuric et al., 2015b).

The overwhelming majority of this work has

Proceedings of the SIGDIAL 2016 Conference, pages 299–303,
Los Angeles, USA, 13-15 September 2016.

focused on using supervised classification with canonical NLP features. Token n-grams are one of the most popular features across many works (Yin et al., 2009; Chen et al., 2012; Warner and Hirschberg, 2012; Xiang et al., 2012; Dadvar et al., 2013). Hand-crafted regular expressions and blacklists also feature prominently in (Yin et al., 2009; Sood et al., 2012; Xiang et al., 2012).

Other features and methodologies have also been found useful. For example, Dadvar et al. (2013) found that in the task of identifying cyberbullying in YouTube comments, a small performance improvement could be gained by including features which model the user's past behavior. Xiang et al. (2012) tackled detecting offensive tweets via semi-supervised LDA approach. Djuric et al. (2015b) use a paragraph2vec approach to classify language on user comments as abusive or clean. Nobata et al. (2016) was the first to evaluate many of the above features on a common corpus and showed an improvement over Djuric et al. (2015b). In this paper, we directly compare against the two works by using the same dataset.

3 Methodology

In general, it is not obvious how to transform comments with different lengths and characteristics to a representation that moves beyond bag of words or words/ngrams based classification approaches. For our work we employ several supervised classification methods with lexical and morphological features to measure various aspects of the user comment. A major difference of our classification phase with previous work in this area is that we use a hybrid method based on discriminative and generative classifiers. As in prior work, we constrained our work to binary classification with comments being abusive or not. Our features are divided into three main classes: tokens, characters and distributional semantics. Our motivation behind using light-weight features, instead of deeper linguistic features (e.g., part of speech tags), is two-fold: i) light-weight features are computationally much less expensive than syntactic or discourse features, and ii) it is very challenging to preprocess noisy and malformed text (i.e., comments) to extract deeper linguistic features.

We explore three different methods for abusive language detection. The first, based on distributional representation of comments (C2V), is meant to serve as a strong baseline for this task. The next two, RNNLM and NBSVM, we use as methodologies for which to explore the impact of character-based vs. token-based features.

3.1 Distributional Representation of Comments (C2V)

The ideas of distributed and distributional word and text representations has supported many applications in natural language processing successfully. The related work is largely focused on the notion of word and text representations (as in (Djuric et al., 2015a; Le and Mikolov, 2014; Mikolov et al., 2013a)), which improve previous works on modeling lexical semantics using vector space models (Mikolov et al., 2013a). More recently, the concept of embeddings has been extended beyond words to a number of text segments, including phrases (Mikolov et al., 2013b), sentences and paragraphs (Le and Mikolov, 2014) and entities (Yang et al., 2014). In order to learn vector representation we develop a comment embeddings approach akin to Le and Mikolov (2014) which is different from the one used in Djuric et al. (2015a) since our representation doesn't model the relationships between the comments (e.g., temporal). Moreover, given the similarity with a prior state-of-the-art approach (Djuric et al., 2015b), this method can also be used as a strong baseline.

In order to obtain the embeddings of comments we learn distributed representations for our comments dataset. The comments are represented as low-dimensional vectors and are jointly learned with distributed vector representations of tokens using a distributed memory model explained in Le and Mokolov (2014). In this work, we train the embeddings of the words in comments using a skip-bigram model (Mikolov et al., 2013a) with window sizes of 5 and 10 using hierarchical softmax training. We also experiment with training two low-dimensional models (100 and 300 dimensions). We limit the number of iterations to 10. For the classification phase we use the Multi-core LibLinear Library (Lee et al., 2015) logistic regression classifier over the resulting embeddings.

3.2 Recurrent Neural Network Language Model (RNNLM)

The intuition behind this model comes from the idea that if we can train a reasonably good language model over the instances for each class, then it will be straightforward to use Bayes rule to predict the class of a new comment. Language

models typically require large amounts of data to achieve a decent performance, but there are currently no large-scale datasets for abuse detection. To overcome this challenge, we exploit the power of recurrent neural networks (RNNs) (Mikolov et al., 2010) which demonstrated state-of-the-art results for language models with less training data (Mikolov, 2012). Another advantage of RNNs is their potential in representing more advanced patterns (Mikolov, 2012). For example, patterns that rely on characters that could have occurred at variable comments can be encoded much more effectively with the recurrent architecture.

We train models for both classes of abusive language in comments (abuse and clean): a) token n-grams for $n = 1..5$, and b) character n-grams for $n = 1..5$ preserving the space character, to investigate our character vs. words claim. During testing, we estimate the ratio of the probability of the comment belonging to each class via Bayes rule. In this way, if the probability of a comment given the abusive language model is higher than its probability given the non-abusive language model, then the comment is classified as abusive and vice versa (Mesnil et al., 2014) and their ratio is used to calculate the AUC metric.

For the experiments we use the RNNLM toolkit developed by (Mikolov et al., 2011). We use 5% of the training set for validation and the rest for training the language model. We train one word ($word$) and two character based language models ($char_1$ & $char_2$). For the $word$ and $char_1$ language models we set the size of hidden layers to 50 with 200 hashes of for direct connections and 4 steps to propagate error back ($bptt$). In order to train a better character-based language model (i.e., $char_2$) we increase the number of hidden layers to 200 and $bptt$ set to 10. Although training a character-based RNN language model with 200 hidden layers takes much longer, our secondary goal is to measure the gains in performance with this more intensive training.

3.3 Support Vector Machine with Naive Bayes Features (NBSVM)

Naive Bayes (NB) and Support Vector Machines (SVM) have been proven to be effective approaches for NLP applications such as sentiment and text analysis. Wang and Manning (2012) showed the power of combining these two generative and discriminative classifiers where an SVM is built over NB log-count ratios as feature values and demonstrated that this combination outperforms the standalone NB and SVM in many tasks using token n-gram features. However, to the best of our knowledge, the effect of character-based NB feature values has not been experimented.

In this work, besides using token n-grams ($n = 1..5$) features, for character level features we compute the log-ratio vector between the average character n-gram counts ($n = 1..5$) from abusive and non-abusive comments. In this way, the input to the SVM classifier is the log-ratio vector multiplied by the binary pattern for each character ngram in the comment vector. For SVM classification we use the Multi-core LibLinear Library (Lee et al., 2015) in its standard setting.

4 Evaluation

4.1 Experimental Setup

We use the same dataset employed in Djuric et al. (2015b) and Nobata et al. (2016). The labels came from a combination of in-house raters, users reactively flagging bad comments and abusive language pattern detectors. To date, this is the largest dataset available for abusive language detection. We use this dataset so as to directly compare with that prior work, and in doing so, we also adopt their evaluation methodology and employ 5-fold cross-validation and report AUC, in addition to recall, precision and F-1. As an additional baseline, we developed a token n-gram classifier with $n = 1..5$ using a logistic regression classifier.

4.2 Results

Table 1 shows the results of all experiments. The four baselines (Djuric et al. (2015), Nobata et al. (2016), token n-grams and C2V) are listed in the first seven rows, and the NBVSM and RNNLM experiments are listed under the double line. We also show the results of a method which combines the token n-grams with the features from the best performing versions of the C2V, NBSVM and RNNLM classes, using our SVM classifier ("Combination").

In terms of overall performance, all methods improved on or tied the Djuric et al. (2015b), C2V and token n-gram baselines. The top performing baseline and current state-of-the-art, Nobata et al. (2016), which consists of a comprehensive combination of a range of different features, is bested by NBSVM using solely character n-grams (77 F-

Method	Rec.	Prec.	F-1	AUC
Djuric et al.	-	-	-	80
Nobata et al.	79	77	78	91
Token n-grams	76	70	73	84
C2V d300w10*	58	77	66	85
C2V d300w5	57	76	66	84
C2V d100w10	56	75	65	82
C2V d100w5	56	76	64	82
NBSVM (word)*	60	**84**	70	89
NBSVM (char)*	72	83	**77**	**92**
RNNLM (word)*	72	59	65	82
RNNLM ($char_1$)*	**78**	60	68	85
RNNLM ($char_2$)	68	68	68	85
Combination	75	**84**	**79**	**93**

Table 1: Results on Djuric et al. (2015) data

1 and 92 AUC). This shows that a light-weight method using character level features can outperform more intricate methods for this task. Moreover, the combination of the best features (marked by * in the Table 1) outperforms all other methods in all measures save for recall. This shows that by increasing the number of relevant features we can improve precision with just a small loss in recall.

For both NBSVM and RNNLM methods, character n-grams outperform their token counterparts (7 and 3 points F-1 score respectively). As most prior work has made use of blacklists and word n-grams, this proves to be an effective method for improving performance.

Comparing the two RNNLM character-based models, using a deeper RNN model improves the precession by 8 points at the loss of 10 points in recall. This finding fits our expectations since, in general, a greater number of hidden layers is needed to achieve a good performance in a character-based language model. We can conclude that for applications which aim at higher recall for abusive language detection, lower hidden layers (e.g., 50) can provide a sufficient performance. However, it should be noted that the more intensive training done in the $char_2$ experiment does not improve upon the 68 F-1 score in $char_1$.

The C2V experiments had the worst performance of all three metrics with the best performance resulting in an F-1 score of 66 using a 300-dimensional vector and a 10 word window (d300w10) while still improving upon the previous approach using paragraph2vec (Djuric et al., 2015b). As one would expect, decreasing the dimensionality of the embedding and the context window results in a loss of performance of as much as 18 F-1 score points (d100w5). However, based on our experiments (not included in the Table), increasing the window size over 10 causes a significant drop in performance. This is due to the fact that most of the comments are rather short (usually under ten tokens) and thus any increase in window length would have no positive impact.

Finally, we performed a manual error analysis of the cases where the character-based approaches and the token-based approaches differed. Naturally, the character-based approaches fared best in cases with irregular normalization or obfuscation of words. For instance, strings with a mixture of letters and digits (i.e., "*ni9*") were caught more readily by the character based methods. There were cases where none of the approaches and methods correctly detected the abuse, usually because the specific trigger words were rare or because the comment was nuanced.

We do note that there are many different types of online communities, and that in communities with little to no moderation, character and word n-grams may perform similarly since the writers may not feel it necessary to obfuscate their words. However, in the many communities where authors are aware of standards, the task becomes much more challenging as authors intentional obfuscate in a myriad of creative ways (Laboreiro and Oliveira, 2014).

5 Conclusions

In this paper, we have made focused contributions into the task of abusive language detection. Specifically, we showed the superiority of simple character-based approaches over the previous state-of-the-art, as well as token-based ones and two deep learning approaches. These light-weight features, when coupled with the right methods, can save system designers and practitioners from writing many regular expressions and rules as in (Sood et al., 2012; Xiang et al., 2012). For future work, we are planning to adapt C2V to the character level.

Acknowledgment

We would like to thank the anonymous reviewers for their comments. We also thank Chikashi Nobata for his contribution to this paper.

References

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 Social Computing (SocialCom)*, pages 71–80. IEEE.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *Advances in Information Retrieval*, pages 693–696. Springer.

Nemanja Djuric, Hao Wu, Vladan Radosavljevic, Mihajlo Grbovic, and Narayan Bhamidipati. 2015a. Hierarchical neural language models for joint representation of streaming documents and their content. In *Proceedings of the International World Wide Web Conference (WWW)*.

Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015b. Hate speech detection with comment embeddings. In *Proceedings of the International World Wide Web Conference (WWW)*.

Gustavo Laboreiro and Eugénio Oliveira. 2014. What we can learn from looking at profanity. In *Computational Processing of the Portuguese Language*, pages 108–113. Springer.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In Tony Jebara and Eric P. Xing, editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*.

Mu-Chu Lee, Wei-Lin Chiang, and Chih-Jen Lin. 2015. Fast matrix-vector multiplications for large-scale logistic regression on shared-memory systems. In Charu Aggarwal, Zhi-Hua Zhou, Alexander Tuzhilin, Hui Xiong, and Xindong Wu, editors, *ICDM*, pages 835–840. IEEE.

Grégoire Mesnil, Tomas Mikolov, Marc'Aurelio Ranzato, and Yoshua Bengio. 2014. Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews. *CoRR*, abs/1412.5335.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *11th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 1045–1048.

Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and Jan Honza Cernocky. 2011. Rnnlm - recurrent neural network language modeling toolkit. IEEE Automatic Speech Recognition and Understanding Workshop, December.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Tomáš Mikolov. 2012. *Statistical Language Models Based on Neural Networks*. Ph.D. thesis.

Preslav Nakov and Jörg Tiedemann. 2012. Combining word-level and character-level models for machine translation between closely-related languages. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju Island, Korea, July.

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 (WWW)*, pages 145–153.

Upendra Sapkota, Steven Bethard, Manuel Montes, and Thamar Solorio. 2015. Not all character n-grams are created equal: A study in authorship attribution. In *Proceedings of the NAACL-HLT '15*, Denver, Colorado, May–June.

Sara Owsley Sood, Judd Antin, and Elizabeth F Churchill. 2012. Using crowdsourcing to improve profanity detection. In *AAAI Spring Symposium: Wisdom of the Crowd*.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, Atlanta, Georgia, June.

Sida Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the ACL '12*, pages 90–94.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, Montréal, Canada, June.

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 CIMK '12*, pages 1980–1984.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *CoRR*, abs/1412.6575.

Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D Davison, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7.