Abusive Language Detection with Graph Convolutional Networks

Pushkar Mishra[★], **Marco Del Tredici**[♣], **Helen Yannakoudakis**[♠], **Ekaterina Shutova**[♣]
[★] Facebook AI, London, United Kingdom
[♣] ILLC, University of Amsterdam, The Netherlands
[♠] The ALTA Institute, Dept. of CS & Technology, University of Cambridge, United Kingdom
pushkarmishra@fb.com, hy260@cam.ac.uk, {m.deltredici, e.shutova}@uva.nl

Abstract

Abuse on the Internet represents a significant societal problem of our time. Previous research on automated abusive language detection in Twitter has shown that community-based profiling of users is a promising technique for this task. However, existing approaches only capture shallow properties of online communities by modeling follower–following relationships. In contrast, working with graph convolutional networks (GCNs), we present the first approach that captures not only the structure of online communities but also the linguistic behavior of the users within them. We show that such a heterogeneous graph-structured modeling of communities significantly advances the current state of the art in abusive language detection.

1 Introduction

Matthew Zook (2012) carried out an interesting study showing that the racist tweets posted in response to President Obama's re-election were not distributed uniformly across the United States but instead formed clusters. This phenomenon is known as *homophily*: i.e., people, both in real life and online, tend to cluster with those who appear similar to themselves. To model homophily, recent research in abusive language detection on Twitter (Mishra et al., 2018a) incorporates embeddings for *authors* (i.e., users who have composed tweets) that encode the structure of their surrounding communities. The embeddings (called *author profiles*) are generated by applying a node embedding framework to an undirected unlabeled community graph where nodes denote the authors and edges the follower–following relationships amongst them on Twitter. However, these profiles do not capture the linguistic behavior of the authors and their communities and do not convey whether their tweets tend to be abusive or not.

In contrast, we represent the community of authors as a heterogeneous graph consisting of two types of nodes, authors and their tweets, rather than a homogeneous community graph of authors only. The primary advantage of such heterogeneous representations is that they enable us to model both community structure as well as the linguistic behavior of authors in these communities. To generate richer author profiles, we then propose a semi-supervised learning approach based on graph convolutional networks (GCNs) applied to the heterogeneous graph representation. To the best of our knowledge, our work is the first to use GCNs to model online communities in social media. We demonstrate that our methods provide significant improvements over existing techniques.

2 Related work

Supervised learning for abusive language detection was first explored by Spertus (1997) who extracted rule-based features to train their classifier. Subsequently, manually-engineered lexical–syntactic features formed the crux of most approaches to the task (Yin et al., 2009; Warner and Hirschberg, 2012). Djuric et al. (2015) showed that dense comment representations generated using *paragraph2vec* outperform bag-of-words features. Several works have since utilized (deep) neural architectures to achieve impressive results on a variety of abuse-annotated datasets (Nobata et al., 2016; Pavlopoulos et al., 2017a). Recently, the research focus has shifted towards extraction of features that capture behavioral and social traits of users. Pavlopoulos et al. (2017b) showed that including randomly-initialized user embeddings improved the performance of their RNN methods. Qian et al. (2018) employed LSTMs to generate inter and intra-user representations based on tweets, but they did not leverage community information.

Proceedings of NAACL-HLT 2019, pages 2145–2150
Minneapolis, Minnesota, June 2 - June 7, 2019.

3 Dataset

Following previous work (Mishra et al., 2018a), we experiment with a subset of the Twitter dataset compiled by Waseem and Hovy (2016). Waseem and Hovy released a list of 16,907 tweet IDs along with their corresponding annotations,[1] labeling each tweet as *racist*, *sexist* or *neither (clean)*. Recently, Mishra et al. (2018a) could only retrieve 16,202 of these tweets since some of them are no longer available. This is the dataset we use in our experiments. 1,939 (12%) of 16,202 tweets are *racist*, 3,148 (19.4%) are *sexist*, and the remaining 11,115 (68.6%) are *clean*. The tweets have been authored by a total of 1,875 unique users. Tweets in the *racist* class come from 5 of the users, while those in the *sexist* class come from 527 of them.

4 Approach

4.1 Representing online communities

We create two different graphs: the first one is identical to the community graph of Mishra et al. (2018a) (referred to as the *community* graph). It contains 1,875 nodes representing each of the authors in the dataset. Two authors/nodes are connected by a single undirected edge if either one follows the other on Twitter. There are 453 *solitary authors* in the graph who are neither followed by nor follow any other author in the dataset. This graph is homogeneous, i.e., it has nodes (and hence edges) of a single type only.

Our second graph is an extended version of the first (referred to as the *extended* graph) that additionally contains nodes representing the tweets of the authors. Specifically, in addition to the 1,875 author nodes, the graph contains 16,202 tweet nodes. Each tweet node is connected to a single author node, denoting that the tweet is elicited from that particular author. This graph is no longer homogeneous since it contains nodes and edges of two different types.

4.2 Generating author profiles

We first describe the approach of Mishra et al. (2018a) that learns author embeddings using *node2vec* (Grover and Leskovec, 2016); this serves as our baseline. We then move on to our semi-supervised approach based on graph convolutional networks (Kipf and Welling, 2017).

[1] https://github.com/ZeerakW/hatespeech/

Node2vec. *Node2vec* extends the *word2vec* skip-gram model (Mikolov et al., 2013) to graphs in order to create low-dimensional embeddings for nodes based on their position and neighborhood. Specifically, for a given graph with nodes $V = \{v_1, v_2, \ldots, v_n\}$, *node2vec* aims to maximize the following log probability:

$$\sum_{v \in V} \log \mathrm{P}\left(N_s(v) \mid v\right)$$

where $N_s(v)$ denotes the neighbor set of node v generated using neighbor sampling strategy s. The framework utilizes two different strategies for sampling neighbor sets of nodes: Depth-First Sampling (DFS) and Breadth-First Sampling (BFS). The former captures the structural role of nodes, while the latter captures the local neighborhood around them. Two hyper-parameters control the overall contribution of each of these strategies. Following Mishra et al. (2018a), we initialize these parameters to their default value of 1 and set the embedding size and number of iterations to 200 and 25 respectively. Since *node2vec* cannot produce embeddings for nodes without edges, we map the *solitary authors* to a single zero embedding as done by Mishra et al.

Graph convolutional networks. We propose an approach for learning author profiles using GCNs applied to the *extended* graph. In contrast to *node2vec*, our method allows us to additionally propagate information with respect to whether tweets composed by authors and their communities are abusive or not. Specifically, as labels are available for a subset of nodes in our graph (i.e., the tweet nodes), we frame the task as a graph-based semi-supervised learning problem, allowing the model to distribute gradient information from the supervised loss on the labeled tweet nodes. This, in turn, allows us to create profiles for authors that not only capture the structural traits of their surrounding community but also their own linguistic behavior based on the types of tweets that they have composed.

We consider a graph $G = (V, E)$, where V is the set of nodes ($|V| = n$) and E is the set of edges. A denotes the adjacency matrix of G. We assume that A is symmetric ($A_{ij} = A_{ji}$), and that all nodes in G have self loops ($A_{ii} = 1$). The significance of these assumptions is explained in Kipf and Welling (2017). Let D be the diagonal degree matrix defined as $D_{ii} = \sum_j A_{ij}$, and $F \in \mathbb{R}^{n \times m}$

be the input feature matrix that holds feature vectors of length m for the nodes in G. We can now recursively define the computation that takes place at the i^{th} *convolutional layer* of a k-layer GCN as:

$$O^{(i)} = \sigma\,(\,\widetilde{A}\,O^{(i-1)}\,W^{(i)}\,)$$

with the computation at the first layer being:

$$O^{(1)} = \sigma\,(\,\widetilde{A}\,F\,W^{(1)}\,)$$

Here, σ denotes an activation function; $\widetilde{A} = D^{-\frac{1}{2}}A\,D^{-\frac{1}{2}}$ is the normalized adjacency matrix; $W^{(i)} \in \mathbb{R}^{d_{i-1}\times d_i}$ is the weight matrix of the i^{th} convolutional layer; $O^{(i-1)} \in \mathbb{R}^{n\times d_{i-1}}$ represents the output from the preceding convolutional layer, where d_i is the number of hidden units in the i^{th} layer (note that $d_0 = m$, i.e., the length of the input feature vectors).

In our experiments, we apply a 2-layer GCN to the *extended* graph.[2] Specifically, our GCN performs the following computation, yielding a softmax distribution over the 3 classes in the dataset for each of the nodes:

$$O = softmax\,(\,\widetilde{A}\;ReLU\,(\,\widetilde{A}\,F\,W^{(1)}\,)\,W^{(2)})$$

We set the input feature vectors in F to be the binary bag-of-words representations of the nodes (following Kipf and Welling 2017); for author nodes, these representations are constructed over the entire set of their respective tweets. Note that F is row-normalized prior to being fed to the GCN. We set the number of hidden units in the first convolutional layer to 200 in order to extract 200-dimensional embeddings for author nodes so that they are directly comparable with those from *node2vec* . The number of hidden units in the second convolutional layer is set to 3 for the output $O \in \mathbb{R}^{n\times 3}$ of the GCN to be a softmax distribution over the 3 classes in the data.

The GCN is trained by minimizing the cross-entropy loss with respect to the labeled nodes of the graph. Once the model is trained, we extract 200-dimensional embeddings $E = \widetilde{A}\,F\,W^{(1)}$ from the first layer (i.e., the layer's output without activation). This contains embeddings for author nodes as well as tweet nodes. For our experiments on author profiles, we make use of the former.

[2]Stacking more layers does not improve results on the validation set further.

4.3 Classification methods

We experiment with five different supervised classification methods for tweets in the dataset. The first three (LR, LR+AUTH, LR+EXTD) serve as our baselines,[3] and the last two with GCNs[4] are the methods we propose.

LR. This method is adopted from Waseem and Hovy (2016) wherein they train a logistic regression classifier on character n-grams (up to 4-grams) of the tweets. Character n-grams have been shown to be highly effective for abuse detection due to their robustness to spelling variations.

LR + AUTH. This is the state of the art method (Mishra et al., 2018a) for the dataset we are using. For each tweet, the profile of its author (generated by *node2vec* from the *community* graph) is appended onto the tweet's character n-gram representation for training the LR classifier as above.

LR + EXTD. This method is identical to LR + AUTH, except that we now run *node2vec* on the *extended* graph to generate author profiles. Intuitively, since *node2vec* treats both author and tweet nodes as the same and does not take into account the labels of tweets, the author profiles generated should exhibit the same properties as those generated from the *community* graph.

GCN. Here, we simply assign a label to each tweet based on the highest score from the softmax distribution provided by our GCN model for the (tweet) nodes of the *extended* graph.

LR + GCN. Identical to LR + EXTD, except that we replace the author profiles from *node2vec* with those extracted by our GCN approach.

5 Experiments and results

5.1 Experimental setup

We run every method 10 times with random initializations and stratified train–test splits. Specifically, in each run, the dataset is split into a randomly-sampled train set (90%) and test set (10%) with identical distributions of the 3 classes in each. In methods involving our GCN, a small part of the train set is held out as validation data to prevent over-fitting using *early-stopping* regularization. When training the GCN, we only have

[3]The implementations of the baselines are taken from https://github.com/pushkarmishra/AuthorProfilingAbuseDetection.

[4]The code we use for our GCN models can be found at https://github.com/tkipf/gcn.

Method	Racism			Sexism			Overall		
	P	R	F_1	P	R	F_1	P	R	F_1
LR	**80.59**	70.62	75.28	83.12	62.54	71.38	83.18	75.62	78.75
LR + AUTH	77.95	78.35	78.15	87.28	78.41	82.61	85.26	83.28	84.18
LR + EXTD	77.95	78.35	78.15	87.02	78.73	82.67	85.17	83.33	84.17
GCN†	74.12	64.95	69.23	82.48	**82.22**	82.35	81.90	79.42	80.56
LR + GCN†	79.08	**79.90**	**79.49**	**88.24**	80.95	**84.44**	**86.23**	**84.73**	**85.42**

Table 1: The baselines (LR, LR + AUTH/EXTD) vs. our GCN approaches (†) on the racism and sexism classes. *Overall* shows the macro-averaged metrics computed over the 3 classes: *sexism*, *racism*, and *clean*.

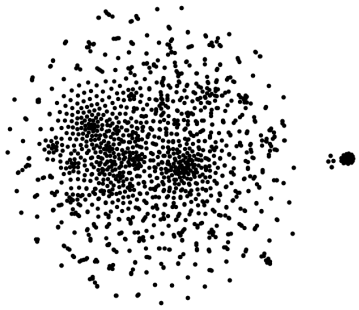
labeled tweet nodes for those tweets in the *extended* graph that are part of the train set. Our GCN is trained using the parameters from the original paper (Kipf and Welling, 2017): *Glorot* initialization (Glorot and Bengio, 2010), ADAM optimizer (Kingma and Ba, 2015) with a learning rate of 0.01, *dropout* regularization (Srivastava et al., 2014) rate of 0.5, 200 training epochs with an early-stopping patience of 10 epochs.

5.2 Results and analysis

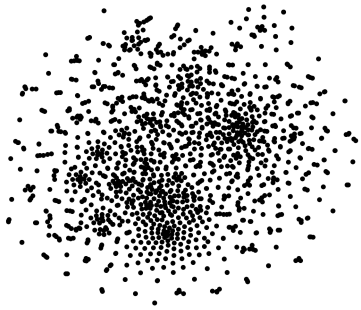
In Table 1, we report the mean precision, recall, and F_1 on the *racism* and *sexism* classes over the 10 runs. We further report the mean macro-averaged precision, recall, and F_1 for each method ('Overall') to investigate their overall performance on the data. LR + GCN significantly ($p < 0.05$ on paired t-test) outperforms all other methods. The author profiles from *node2vec* only capture the structural and community information of the authors; however, those from the GCN also take into account the (abusive) nature of the tweets composed by the authors. As a result, tweets like "*#MKR #mkr2015 Who is gonna win the peoples choice?*" that are misclassified as sexist by LR + AUTH (because their author is surrounded by others producing sexist tweets) are correctly classified as clean by LR + GCN.

GCN on its own achieves a high performance, particularly on the *sexism* class where its performance is typical of a community-based profiling approach, i.e., high recall at the expense of precision. However, on the *racism* class, its recall is hindered by the same factor that Mishra et al. (2018a) highlighted for their *node2vec*-only method, i.e., that racist tweets come from 5 unique authors only who have also contributed sexist or clean tweets. The racist activity of these authors is therefore eclipsed, leading to misclassifications of their tweets. LR + GCN alleviates this problem by incorporating character n-gram representations of the tweets, hence not relying solely on the linguistic behavior of their authors.

Figure 1 shows the t-SNE (van der Maaten and Hinton, 2008) visualizations of *node2vec* author profiles from the *community* and *extended* graphs. Both visualizations show that some authors belong to densely-connected communities while others are part of more sparse ones. The results from LR + AUTH and LR + EXTD have insignificant differences, further confirming that their author profiles have similar properties. In essence, *node2vec* is unable to gain anything more from the *extended* graph than what it does from the *community* graph.



(a) Author profiles from the *community* graph



(b) Author profiles from the *extended* graph

Figure 1: Visualizations of the *node2vec* author profiles from the *community* and *extended* graphs.

Figure 2 shows a t-SNE visualization of the author profiles generated using our GCN approach. Red dots denote the authors who are abusive (sexist or racist) according to our model (i.e., as per

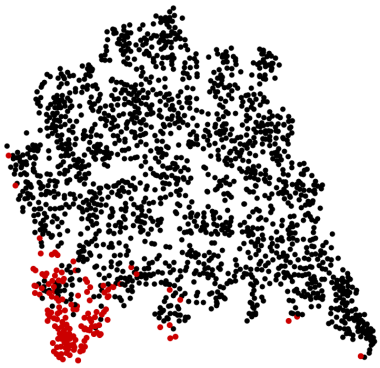


Figure 2: Visualization of the author profiles extracted from our GCN. Red dots represent the authors who are deemed abusive (racist or sexist) by the GCN.

the softmax outputs for the author nodes).[5] The red dots are mostly clustered in a small portion of the visualization, which corroborates the notion of homophily amongst abusive authors.

Despite the addition of improved author profiles, several abusive tweets remain misclassified. As per our analysis, many of these tend to contain URLs to abusive content but not the content itself, e.g., "*@MENTION: Logic in the world of Islam http://t.co/6nALv2HPc3*" and "*@MENTION Yes. http://t.co/ixbt0uc7HN*". Since Twitter shortens all URLs into a standard format, there is no indication of what they refer to. One possible way to address this limitation could be to append the content of the URL to the tweet; however this can lead to misclassifications in cases where the tweet is disagreeing with the URL. Another factor in misclassifications is the deliberate obfuscation of words and phrases by authors in order to evade detection, e.g., "*Kat, a massive c*nt. The biggest ever on #mkr #cuntandandre*". Mishra et al. (2018b) demonstrate in their work that character-based word composition models can be useful in dealing with this aspect.

6 Conclusions

In this paper, we built on the work of Mishra et al. (2018a) that introduces community-based profiling of authors for abusive language detection. We proposed an approach based on graph convolutional networks to show that author profiles that directly capture the linguistic behavior of authors along with the structural traits of their community significantly advance the current state of the art.

[5] Note that there are no such gold labels for authors in the dataset itself.

Acknowledgments

We would like to thank the anonymous reviewers for their useful feedback. Helen Yannakoudakis was supported by Cambridge Assessment, University of Cambridge.

References

Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 29–30. Association for Computing Machinery.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy. PMLR.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, ICLR '15.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

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

Pushkar Mishra, Marco Del Tredici, Helen Yannakoudakis, and Ekaterina Shutova. 2018a. Author profiling for abuse detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1088–1098. Association for Computational Linguistics.

Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2018b. Neural character-based composition models for abuse detection. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 1–10. Association for Computational Linguistics.

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 '16, pages 145–153, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2017a. Deeper attention to abusive user content moderation. In *Proceedings of EMNLP 2017*, pages 1125–1135. Association for Computational Linguistics.

John Pavlopoulos, Prodromos Malakasiotis, Juli Bakagianni, and Ion Androutsopoulos. 2017b. Improved abusive comment moderation with user embeddings. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*, pages 51–55. Association for Computational Linguistics.

Jing Qian, Mai ElSherief, Elizabeth Belding, and William Yang Wang. 2018. Leveraging intra-user and inter-user representation learning for automated hate speech detection. In *Proceedings of the 2018 Conference of the NAACL: Human Language Technologies, Volume 2 (Short Papers)*, pages 118–123. Association for Computational Linguistics.

Ellen Spertus. 1997. Smokey: Automatic recognition of hostile messages. In *Proceedings of the 14th AAAI and 9th IAAI*, AAAI'97/IAAI'97, pages 1058–1065. AAAI Press.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the 2nd Workshop on Language in Social Media*, LSM '12, pages 19–26. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Dawei Yin, Brian D. Davison, Zhenzhen Xue, Liangjie Hong, April Kontostathis, and Lynne Edwards. 2009. Detection of harassment on web 2.0. In *Processings of the Content Analysis in the WEB 2.0*.

Matthew Zook. 2012. Mapping racist tweets in response to president obama's re-election. [Online; accessed 01 October 2018].