ARHNet - Leveraging Community Interaction For Detection Of Religious Hate Speech In Arabic

Arijit Ghosh Chowdhury*
Manipal Institute of Technology
arijit10@gmail.com

Aniket Didolkar*
Manipal Institute of Technology
adidolkar123@gmail.com

Ramit Sawhney
Netaji Subhas Institute of Technology
ramits.co@nsit.net.in

Rajiv Ratn Shah
MIDAS, IIIT-Delhi
rajivratn@iiitd.ac.in

Abstract

The rapid widespread of social media has led to some undesirable consequences like the rapid increase of hateful content and offensive language. Religious Hate Speech, in particular, often leads to unrest and sometimes aggravates to violence against people on the basis of their religious affiliations. The richness of the Arabic morphology and the limited available resources makes this task especially challenging. The current state-of-the-art approaches to detect hate speech in Arabic rely entirely on textual (lexical and semantic) cues. Our proposed methodology contends that leveraging Community-Interaction can better help us profile hate speech content on social media. Our proposed ARHNet (Arabic Religious Hate Speech Net) model incorporates both Arabic Word Embeddings and Social Network Graphs for the detection of religious hate speech.

1 Introduction

Hate speech was a major tool employed to promote slavery in Colonial America, to aggravate tensions in Bosnia and in the rise of the Third Reich. The aim of such speech is to ridicule victims, to humiliate them and represent their grievances as less serious (Gelashvili, 2018). The relationship between religion and hate speech is complex and has been central to recent discussions of hate speech directed at religious people, especially members of religious minorities (Bonotti, 2017). This makes it important to develop automated tools to detect messages that use inflammatory sectarian language to promote hatred and violence against people.

Our work extends on the work done by (Albadi et al., 2018) in terms of exploring the merits of introducing community interaction as a feature in the detection of religious hate speech in Arabic. Most previous work in the area of hate speech detection has targeted mainly English content (Davidson et al., 2017) (Djuric et al., 2015) (Badjatiya et al., 2017). Author profiling using community graphs has been explored by (Mishra et al., 2018) for abuse detection on Twitter. We propose a novel Cyber Hate Detection approach using multiple twitter graphs and traditional word embeddings.

Social network graphs are increasingly being used as a powerful tool for NLP applications (Mahata et al., 2018; Shah et al., 2016b), leading to substantial improvement in performance for tasks like text categorization, sentiment analysis, and author attribute identification ((Hovy, 2015); (Yang and Eisenstein, 2015); (Yang et al., 2016). The idea of using this type of information is best explained by the concept of homophily, i.e., the phenomenon that people, both in real life as well as on the Internet, tend to associate more with those who appear similar. Here, similarity can be defined based on various parameters like location, age, language, etc. The basic idea behind leveraging community interaction is that if we have information about members of a community defined by some similarity measure, then we can infer information about a person based on which community they belong to. For our study, knowing that members of a particular community are prone to proliferating religious hate speech content, and knowing that the user is connected to this community, we can use this information beyond linguistic cues and more accurately predict the use of hateful/non-hateful language. Our work seeks to address two main questions:

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pages 273–280
Florence, Italy, July 28 - August 2, 2019.

- Is one community more prone to spreading hateful content than the other?

- Can such information be effectively leveraged to improve the performance of the current state of the art in the detection of religious hate speech within Arabic speaking users?

In this paper, we do an in-depth analysis of how adding community features may enhance the performance of classification models that detect religious hate speech in Arabic.

2 Related Work

Hate speech research has been conducted extensively for the English language. Amongst the first ones to apply supervised learning to the task of hate speech detection were (Yin and Davison, 2009) who used a linear SVM classifier to identify posts containing harassment based on local, contextual and sentiment-based (e.g., presence of expletives) features. Their best results were with all of these features combined. Notably, (Waseem and Hovy, 2016) created a dataset for detection of Hate Speech on Twitter. They noted that character n-grams are better predictive features than word n-grams for recognizing racist and sexist tweets. Their n-gram-based classification model was outperformed using Gradient Boosted Decision Trees classifier trained on word embeddings learned using LSTMs (Waseem and Hovy, 2016). There has been limited literature on the problem of Hate Speech detection on Arabic social media. (Magdy et al., 2015) trained an SVM classifier to predict whether a user is more likely to be an ISIS supporter or opposer based on features of the users tweets.

Social Network graphs have been leveraged in several ways for a variety of purposes in NLP. Given the graph representing the social network, such methods create low-dimensional representations for each node, which are optimized to predict the nodes close to it in the network. Among those that implement this idea are (Yang et al., 2016), who used representations derived from a social graph to achieve better performance in entity linking tasks, and Chen and Ku (Yang and Eisenstein, 2015), who used them for stance classification. A considerable amount of literature has also been devoted to sentiment analysis with representations built from demographic factors ((Yang and Eisenstein, 2015); (Chen and Ku, 2016)). Other tasks that have benefited from social representations are sarcasm detection (Amir et al., 2016) and political opinion prediction (Tlmcel and Leon, 2017).

To our knowledge, so far there has been no substantial research on using social network graphs as features to analyze and categorize tweets in Arabic. Our work proposes a novel architecture that builds on the current state of the art and improves its performance using community graph features.

3 Data

We conduct our experiments with the dataset provided by (Albadi et al., 2018). The authors collected the tweets referring to different religious groups and labeled them using crowdsourced workers. In November 2017, using Twitters search API 2, the authors collected 6000 Arabic tweets, 1000 for each of the six religious groups. They used this collection of tweets as their training dataset. Due to the unavailability of a hate lexicon and to ensure unbiased data collection process; they included in their query only impartial terms that refer to a religion name or the people practicing that religion. In January 2018, they collected another set of 600 tweets, 100 for each of the six religious groups, for their testing dataset. After an inter-annotator agreement of 81% , 2526 tweets were labeled as *Hate*.

The dataset was released as a list of 5570 tweet IDs along with their corresponding annotations. Using the python Twarc library, we could only retrieve 3950 of the tweets since some of them have now been deleted or their visibility limited. Of the ones retrieved, 1,685 (42.6%) are labelled as *hate*, and the remaining 2,265 (57.4%) as *Non-Hate*; this distribution follows the original dataset very closely (45.1%, 54.9%).

3.1 Preprocessing

We followed some of the Arabic-specific normalization steps proposed in (Albadi et al., 2018) along with some other Twitter-specific preprocessing techniques.

- Normalization of Hamza with alef seat to bare alef.

- Normalization of dotless yeh (alef maksura) to yeh.

- Normalization of teh marbuta to heh.

Hate
اليوم الثلاثاء لعنة الله على اليهود الذين هم أبناء الشيطان وعلى مساعديه الذين هم وقود الجحيم TuesdayMorning curse of god on the jews who are the sons of the satan and on their helpers who are the fuel of hell
اللهم طهر الأرض من الشيعة المنافقين ومن يتبعهم Oh god purify the land from the rawafid hypocrite Shia and those who follow them
الله يلعن فيتفا لعنه اليهود والنصاري God cursed Vittafa cursed Jews and Christians
مؤله المسلم يقتل اخاه المسلم بالسلاح ويحاول قتل اليهودي بالدعاء تبا لعروبتكم حقيقه Muslim Muslim kills his Muslim brother and tries to kill the Jew by praying to the Arabs

Table 1: Examples for Hate Speech.

Non-Hate
مؤسسة أرشيف المغرب تتسلم وثائق عن ذاكرة اليهودگ المغاربة The Moroccan Archives Foundation receives documents on the memory of Moroccan Jews
يالله انزل المن والسلوي كما انزلته علي اليهود (المفضلون علي كل البشر) في التيه God sent down the Manna and the Salafi as it sent down on the Jews (the favored of all human beings) in Hell
مؤسسة أرشيف المغرب تتسلم وثائق عن ذاكرة اليهود المغاربة الخميس المقبلگ بالرباط Morocco's Shiv receives documents on the memory of Moroccan Jews next Thursday in Rabat
كلنا اولاد ادم مسلمين مسيحين يهود صهاينه بس لم نعد نحترم الانسانيه We are all Adam's children, Muslims, Christian Jews, Zionists, but we no longer respect humanity

Table 2: Examples for Non-Hate Speech.

- Normalizing links, user mentions, and numbers to somelink, someuser, and somenumber, respectively.
- Normalizing hashtags by deleting underscores and the # symbol.
- Removing diacritics (the harakat), tatweel (stretching character), punctuations, emojis, non-Arabic characters, and one-letter words.
- Repeated characters were removed if the repetition was of count three or more.
- We used the list of 356 stopwords created by (Albadi et al., 2018). This list did not have negation words as they usually represent important sentiments.
- Stemming: We used the ISRI Arabic Stemmer provided by NLTK to handle inflected words and reduce them to a common reduced form.

4 Methodology

4.1 Community and Social Interaction Network

To leverage information about community interaction, we create an undirected unlabeled social network graph wherein nodes are the authors and edges are the connections between them.

We use two social network graphs in our study :

- **Follower Graph** : This is an unweighted undirected graph G with nodes v representing authors, with edges e such that for each $e \in E$, there exists u, $v \in$ the set of authors such that u follows v or vice versa.
- **Retweet Graph** : This is an unweighted undirected graph G with nodes v representing authors, with edges e such that for each $e \in E$, there exists u, $v \in$ the set of authors such that u has retweeted v or vice versa.

From these social network graphs, we obtain a vector representation, i.e., an embedding that we refer to as an *Interaction*, for each author using the *Node2Vec* framework (Grover and Leskovec, 2016). *Node2Vec* uses a skip-gram model (Mikolov et al., 2013) on a graph to create a representation for each of its nodes based on their positions and their neighbors. Given a graph with nodes $V = v1, v2, ..., vn$, Node2Vec seeks to maximize the following log probability:

$$\sum_{v \in V} LogP_r(N_s(v)|v)$$

where $N_s(v)$denotes the network neighborhood of node v generated through sampling strategy s. The framework can learn low-dimensional embeddings for nodes in the graph. These embeddings can emphasize either their structural role or the local community they are a part of. This depends on the sampling strategies used to generate the neighborhood: if breadth-first sampling (BFS) is adopted, the model focuses on the immediate neighbors of a node; when depth-first sampling (DFS) is used, the model explores farther regions in the network, which results in embeddings that encode more information about structural role of a particular node . The balance between these two ways of sampling the neighbors is directly controlled by two node2vec parameters, namely p and q. The default value for these is 1, which ensures a node representation that gives equal weight to both structural and community-oriented information. In our work, we use the default value for both p and q. Additionally, since Node2Vec does not produce embeddings for single users without a community, these have been mapped to a single zero embedding. The dimensions of these embeddings were 64.

Figure 1 shows an example of a community. The nodes represent users and the edges represent an *Interaction* between them.

4.2 Classification

For every tweet $t_i \in D$, in the dataset, a binary valued value variable y_i is used, which can either be 0 or 1. The value 0 indicates that the text belongs to the *Non-Hate category* while 1 indicates *Hate Speech*.

The following steps are executed for every tweet $t_i \in D$:

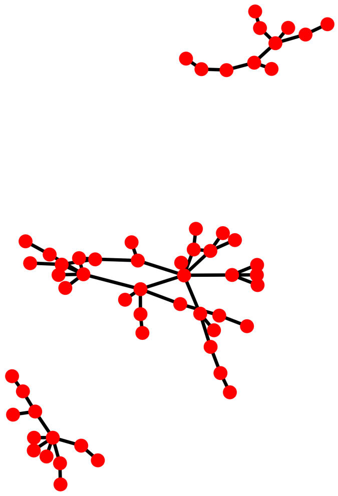

Figure 1: A community interaction snippet from $g_{retweet}$

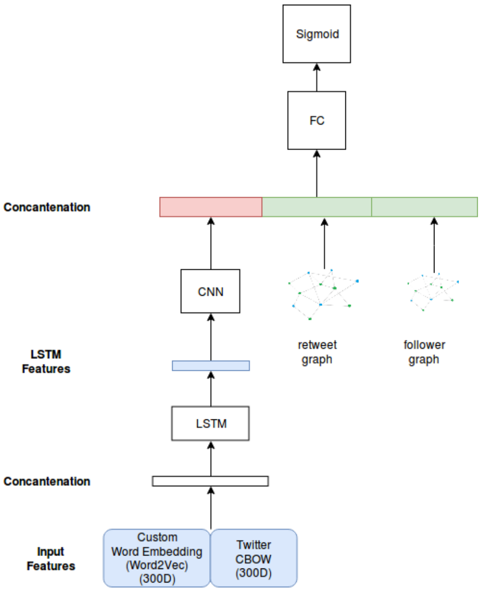

Figure 2: The ARHNet Architecture

1. *Word Embeddings*. All the words in our vocabulary are encoded to form 600-dimensional word embeddings obtained by concatenating Twitter-CBOW 300-dimensional embedding with our trained embedding.

2. *Sentence Representation*. This is obtained by passing the word embeddings through the corresponding deep learning model.

3. *Node Embeddings*. The node embedding for the author of t_i is concatenated with the sentence representation to get the final representation.

4. *Dense Layer*. The final representation is passed through a dense layer which outputs

Architecture	Accuracy	Precision	Recall	F1	AUROC
AraHate-LR	0.75	0.72	0.74	0.73	0.82
AraHate-SVM	0.75	0.72	0.72	0.72	0.81
AraHate-GRU	0.77	0.65	0.89	0.75	0.84
GRU + self-attention	0.78	0.71	0.78	0.74	0.83
GRU + CNN	0.79	0.69	0.86	0.77	0.86
LSTM	0.76	0.65	0.86	0.74	0.82
LSTM + self-attention	0.78	0.68	0.82	0.75	0.86
LSTM + CNN	0.80	0.71	0.83	0.77	0.86
Bidirectional GRU	0.79	0.70	0.85	0.77	0.85
Bidirectional GRU + self-attention	0.80	0.74	0.80	0.77	0.87
Bidirectional GRU + CNN	0.79	0.71	0.81	0.76	0.85
Bidirectional LSTM	0.80	0.73	0.79	0.76	0.86
Bidirectional LSTM + self-attention	0.77	0.66	0.86	0.75	0.87
Bidirectional LSTM + CNN	0.81	0.74	0.81	0.77	0.86

Table 3: Performance of various deep learning models.

Architecture	Accuracy	Precision	Recall	F1	AUROC
GRU + NODE2VEC	0.79	**0.74**	0.76	0.75	0.85
GRU + self-attention + NODE2VEC	0.78	0.67	0.87	0.75	0.84
GRU + CNN + NODE2VEC	0.80	0.68	0.87	0.77	0.85
LSTM + NODE2VEC	0.75	0.63	0.86	0.73	0.81
LSTM + self-attention + NODE2VEC	0.78	0.70	0.79	0.74	0.84
LSTM + CNN + NODE2VEC (ARHNet)	0.79	0.69	**0.89**	**0.78**	**0.86**
Bi-GRU + NODE2VEC	0.79	0.67	0.86	0.75	0.85
Bi-GRU + self-attention + NODE2VEC	0.79	0.70	0.82	0.76	0.86
Bi-GRU + CNN + NODE2VEC	**0.81**	0.72	0.84	0.77	0.86
Bi-LSTM + NODE2VEC	0.80	0.73	0.81	0.77	0.86
Bi-LSTM + self-attention + NODE2VEC	0.78	0.68	0.82	0.75	0.85
Bi-LSTM + CNN + NODE2VEC	0.80	0.73	0.81	0.77	0.86

Table 4: Performance of various deep learning models with community features.

a score that is converted to a probability distribution using a sigmoid activation.

4.3 Baselines

An extensive comparison with state-of-the-art generic and specific models the case for our proposed methodology. To make a fair comparison between all the methodologies, the experiments are conducted concerning the baselines in (Albadi et al., 2018) have used a simple GRU model as their best performing model. Their GRU model uses 240 hidden features. They have also compared results with Logistic Regression and Support Vector Machine Models. The Logistic regression classifier was trained using character n-gram features (n =1-4) with L2 regularization. The SVM classifier was also trained using character n-gram features (n = 1-4) with linear kernel and L2 regularization, similar to (Albadi et al., 2018). For the GRU model, they have used the Twitter-CBOW 300-dimensional embedding model(Soliman et al., 2017) for obtaining word embeddings. The output of the embedding layer was fed into a dropout layer with probability 0.5. They used batches of size 32 and Adam as their optimizer. We refer the models trained by (Albadi et al., 2018) as the AraHate baselines. We conduct our experiments with LSTM (Liu et al., 2016) and CNN-LSTM (Zhou et al., 2015) models. LSTMs can capture long term dependencies better than RNNs and GRUs, and a CNN-LSTM network utilizes the ability of a CNN to extract higher-level phrase representations, which are fed into an LSTM. We did not increase the complexity of the baselines beyond this to not risk overfitting on a small dataset.

4.4 Models and Hyperparameters

First, we prepared the vocabulary by assigning integer indexes to unique words in our dataset. Tweets were then converted into sequences of integer indexes. These sequences were padded with zeros so that the tweets in each batch have the same length during training. They were then fed into an embedding layer which maps word indexes to word embeddings. We trained our word embeddings using GenSim [1]. We also used the Twitter-CBOW 300-dimension embedding model provided by AraVec (Soliman et al., 2017) which contains over 331k word vectors that have been trained on about 67M Arabic tweets. We concatenated our own trained embeddings with the AraVec embeddings to obtain 600-dimensional embeddings Similar to (Albadi et al., 2018), The output of the embedding layer was fed into a dropout layer with a rate of 0.5 to prevent overfitting.

For both LSTM and GRU, the word embeddings were passed to both unidirectional and bidirectional LSTM with 240 features each. In the GRU-CNN/LSTM-CNN models, we used 2 Convolutional Layers with a Kernel Size of 3 and Relu Activation in the middle. We obtained the final representation by taking the maximum along the temporal dimension. For self-attention, the output of the GRU/LSTM was passed to a self-attention layer. For the self-attention models, we used 240 features.

We compared each of these models with their counterparts obtained by concatenating Node2Vec embeddings to the representations obtained by the above deep learning models. The final representation was then passed into a Sigmoid Layer. We performed training in batches of size 32, and we used Adam as our optimizer for all experiments.

5 Results And Discussion

In our experiments, we have beaten the scores of (Albadi et al., 2018) in all 5 metrics. We obtained a highest f1-score of 0.78 as compared to 0.77 in (Albadi et al., 2018). This is achieved in our LSTM + CNN + CISNet model. The ARHNet model outperforms baselines in terms of Recall, F1 and AUROC metrics while GRU-NODE2VEC demonstrates the highest precision, and the Bi-GRU-CNN-NODE2VEC model achieves the highest accuracy. Our methodology effectively improves upon the current state of the art and is successful in demonstration of how community interaction can be leveraged to tackle downstream NLP tasks like detection of religious hate speech. Albadi et al. (2018) reached an 0.81 agreement score between annotators. Our methodology, therefore, matches human performance in terms of unambiguously categorizing texts that contain religious hate speech from texts that don't.

To summarize, our approach highlights the validity of using Community Interaction Graphs as features of classification in Arabic. Despite having a sparse representation of users, our proposed methodology has shown improvements on Accuracy and F1 over previously state of the art models on a reduced dataset.

6 Conclusion

In this paper, we explored the effectiveness of community-interaction information about authors for the purpose of categorizing religious hate speech in the Arabic Twittersphere and build upon existing work in the linguistic aspects of social media (Shah et al., 2016c,a; Mahata et al., 2015). Working with a dataset of 3950 tweets annotated for *Hate* and *Non-Hate*, we first comprehensively replicated three established and currently best-performing hate speech detection methods based on character n-grams and GRUs as our baselines. We then constructed a graph of all the authors of tweets in our dataset and extracted community-based information in the form of dense low-dimensional embeddings for each of them using Node2Vec. We showed that the inclusion of community graph embeddings significantly improves system performance over the baselines and advances the state of the art in this task. Users prone to proliferate hate do tend to form social groups online, and this stresses the importance of utilizing community-based information for automatic religious hate speech detection.

References

N. Albadi, M. Kurdi, and S. Mishra. 2018. Are they our brothers? analysis and detection of religious hate speech in the arabic twittersphere. In *2018 IEEE/ACM International Conference on Advances*

[1] radimrehurek.com/gensim/models/word2vec.html

in Social Networks Analysis and Mining (ASONAM), pages 69–76.

Silvio Amir, Byron C. Wallace, Hao Lyu, Paula Carvalho, and Mário J. Silva. 2016. Modelling context with user embeddings for sarcasm detection in social media. *CoRR*, abs/1607.00976.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. *CoRR*, abs/1706.00188.

Matteo Bonotti. 2017. Religion, hate speech and non-domination. *Ethnicities*, 17:259–274.

Wei-Fan Chen and Lun-Wei Ku. 2016. UTCNN: a deep learning model of stance classification on social media text. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1635–1645, Osaka, Japan. The COLING 2016 Organizing Committee.

Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *CoRR*, abs/1703.04009.

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, New York, NY, USA. ACM.

Teona Gelashvili. 2018. Hate speech on social media: Implications of private regulation and governance gaps. Student Paper.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. *CoRR*, abs/1607.00653.

Dirk Hovy. 2015. Demographic factors improve classification performance. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 752–762, Beijing, China. Association for Computational Linguistics.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *CoRR*, abs/1605.05101.

Walid Magdy, Kareem Darwish, and Ingmar Weber. 2015. #failedrevolutions: Using twitter to study the antecedents of ISIS support. *CoRR*, abs/1503.02401.

Debanjan Mahata, Jasper Friedrichs, Rajiv Ratn Shah, and Jing Jiang. 2018. Detecting personal intake of medicine from twitter. *IEEE Intelligent Systems*, 33(4):87–95.

Debanjan Mahata, John R Talburt, and Vivek Kumar Singh. 2015. From chirps to whistles: discovering event-specific informative content from twitter. In *Proceedings of the ACM web science conference*, page 17. ACM.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

Pushkar Mishra, Marco Del Tredici, Helen Yannakoudakis, and Ekaterina Shutova. 2018. Author profiling for abuse detection. In *COLING*.

Rajiv Ratn Shah, Anupam Samanta, Deepak Gupta, Yi Yu, Suhua Tang, and Roger Zimmermann. 2016a. Prompt: Personalized user tag recommendation for social media photos leveraging personal and social contexts. In *2016 IEEE International Symposium on Multimedia (ISM)*, pages 486–492. IEEE.

Rajiv Ratn Shah, Yi Yu, Suhua Tang, Shin'ichi Satoh, Akshay Verma, and Roger Zimmermann. 2016b. Concept-level multimodal ranking of flickr photo tags via recall based weighting. In *Proceedings of the 2016 ACM Workshop on Multimedia COMMONS*, pages 19–26. ACM.

Rajiv Ratn Shah, Yi Yu, Akshay Verma, Suhua Tang, Anwar Dilawar Shaikh, and Roger Zimmermann. 2016c. Leveraging multimodal information for event summarization and concept-level sentiment analysis. *Knowledge-Based Systems*, 108:102–109.

Abu Bakr Soliman, Kareem Eissa, and Samhaa R. El-Beltagy. 2017. Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science*, 117:256 – 265. Arabic Computational Linguistics.

C. Tlmcel and F. Leon. 2017. Predicting political opinions in social networks with user embeddings. In *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 213–219.

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.

Yi Yang, Ming-Wei Chang, and Jacob Eisenstein. 2016. Toward socially-infused information extraction: Embedding authors, mentions, and entities. *CoRR*, abs/1609.08084.

Yi Yang and Jacob Eisenstein. 2015. Putting things in context: Community-specific embedding projections for sentiment analysis. *CoRR*, abs/1511.06052.

Dawei Yin and Brian D. Davison. 2009. Detection of harassment on web 2.0.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. 2015. A C-LSTM neural network for text classification. *CoRR*, abs/1511.08630.