Detecting Hate Users on Twitter using Graph methods

Aldo Pioline Antony Charles
Northeastern University
antonycharles.a@northeastern.edu

ABSTRACT
Hate speech is a prevalent problem in social media. Detecting hate speech is a complex problem due to the existence of various dialects and contexts. Traditional methods focused on standalone content-level features to detect hate speech which is insufficient. The paper proposes a user-based approach to detect hate through methods based on network science and machine learning. The paper explores how such methods can be applied to a social media user network. Methods used in the paper include GloVe, Empath, boosted tree models, Graph Convolution networks, GraphSage, Node2Vec.

Keywords
Hate speech, Graph convolution networks, GraphSage, User Embedding, Node2vec, Word2vec, GloVe

1. INTRODUCTION
Social media was created for users to share information with their friends and to the world. With the rise in access to the internet, everyone uses some form of social media especially Instagram, Facebook and Twitter. Twitter specifically has been a powerful platform for people to share their views and get noticed in both good and bad ways and have paved way for many changes and revolutions in the real world. Social media is therefore a powerful weapon that should be carefully used by everyone. But, since Twitter is free to use for everyone, there are all kinds of users actively using the platform; Especially the users who use the platform for negative reasons such as spreading hate information. Social media admins have a hard time detecting these hate speeches and making right decisions due to the complexity of each language. For example, An African American user uses AAE dialect casually with no intention of spreading hate and the fellow African American users tend to understand that, but it could be offensive to a different demographic. This opens a whole discussion on what should be allowed and what should not. The social media platforms have employed pure content-based approaches traditionally even now to detect and ban the users using certain languages and there has been a surge of unnecessary bans overtime. Due to this nature, researchers have been trying to develop and employ different methods that truly identifies the hate users and help make the admin take the right decision. One such method is a user-focused approach using social media network graphs. It is an umbrella term containing various algorithms and methods with specific tasks that can be used for various contexts. These methods rather than only focusing on what a user posts, focusses on the user's social media characteristics and their content and then makes a judgement. We focus on few such methods here and show their effectiveness in identifying the hate users by their content and user-level features mainly and the paper discusses also how to extend it to real-world decision making.

2. RELATED WORK
Most of the approaches use language features to detect hate speech on social media. Some of the popular approaches include [1] uses a n-grams and skip-grams to detect hate speech which is purely content based. [2] uses regular expressions to match hate speech on Twitter and Whisper datasets in social media. [3] showcases a comprehensive survey of hate speech across various social media platforms and how various algorithmic methods that have been deployed to tackle the problem and proposes some good results and enhancements for detecting hate speech as well. Most of the other works produced in recent years revolve around the same ideas showcased above. There have been recent publications such as [4] uses Knowledge Graphs to detect hate speech on a social media network to build a classifier. [5] explores a similar approach to the current paper. The author uses Graph convolution networks to detect hate speech and provide detailed experiments to prove their strength.

3. METHODS
3.1 GloVe Embedding
GloVe embedding is a popular technique developed at Stanford to obtain vector representations of data. The method is trained on global word to word co-occurrence statistics which help identity related words and create embedding. In this paper, we run GloVe embedding on every User's tweet over time and create n-dimensional vector representations which is used as one of the primary features for the later downstream tasks.

3.2 Empath
Empath is a tool developed at Stanford University that can predict lexical categories from a given text. It uses relations between words and phrases using neural embeddings across 1.8 billion words from works of fiction. They also provide score for different predicted categories to show how important each category is to a document. Empath also provides an API for users to determine the lexical categories for a corpus. Here, we run empath on each user's historical tweets and determine scores across various categories and use them as features for the downstream tasks.

3.3 Gradient Boosted Trees
Gradient Boosted trees is an ensembling technique that combines multiple learners to make predictions. Here we use AdaBoost and Gradient Boost as our primary classifiers using the features obtained using GloVe and Empath and other network-based user features to predict if the user is hateful or not.

3.4 Graph Convolutional Networks
Graph convolutional networks (GCN) proposed by Thomas Kipf and Max Welling gained recent popularity due to their ability to learn non-Euclidian data in contrast to regular convolutional

neural networks that could only learn Euclidian representations. GCN uses a similar approach to CNN where they learn a graph node's embedding by surveying the features of its neighbors. They use graph propagation to use those features and a non-linear activation to represent non-linear features in the latent space. They are effective in learning graph data. In this paper, we use all the features discussed above and build a single layer graph network to predict a user to be hateful or not.

3.5 Node2Vec

Node2Vec is an embedding approach developed at SNAP, Stanford University to represent graph nodes. It uses Word2Vec to learn the similarity between user nodes and learn the representations. In this paper, we use Node2Vec to create user representation using their historical tweets and compare their usage against GloVe by combining with various classifiers.

3.6 GraphSage

GraphSage is a recent technique that is highly effective for graph learning. They use an inductive approach to learn graphs (i.e.) they do not require the entire graph to learn the representations, but they have the power to generalize to unseen nodes when they are added to the graph. Here, we see GraphSage to be highly accurate in the context of Hate User detection because of their aggregation technique. [6] implementation of GraphSage was used for experimentation purposes. [7]

4. EXPERIMENTS

4.1 Data

The paper uses the Hateful Users on Twitter Dataset from Kaggle [8]. The dataset contains files about user graph neighborhood, each users' glove embedding, users' twitter features and an annotated csv file containing all the above details with a labelled column if the user is hateful, not hateful (normal) or unknown (other). The data was collected using Twitter API over the period of October 2017 – January 2018. They collect each user's 200 most recent tweets. The dataset in total consists of 100,386 unique users and 2,286,592 edges. They collect the tweets using 23-word lexicon containing words related to hate collected from hatebase.org. The authors of the dataset then annotated each user through the help of Crowdflower, a crowdsourcing service and they managed to annotate 4972 users. The dataset contains 544 hateful users and 4427 normal users. The dataset also contains empath values for different topics for each user. They also contain network details such as betweenness, hateful neighbor count, normal neighbor count and twitter related features such as followers, following, etc.

4.2 Evaluation and Results

As an initial experiment to test the features, gradient boosted methods were used. AdaBoost and Gradient Boost were chosen for this case. K fold cross validation was used to split and fit the data. The data and labels were then fit on both the models. AdaBoost used the following hyperparameters: n_estimators=75, learning_rate=0.01, and Gradient Boost used the following: max_depth=5, n_estimators=100, learning_rate=0.01. First the models were run on only the GloVe vectors calculated from each users' tweets and then the rest of the features were added along with GloVe and re-run.

Secondly, in order to test the effectiveness of the network features using a graphical method, Graph Convolutional Networks (GCN) is used. We use a vanilla GCN with a single Droupout layer, a Graph Convolution layer with the following settings: 32 units, activation = 'relu', use_bias=True; A dense layer using Softmax to predict the probability of each of the three classes is used as output. The model uses Adam to optimize the cross-entropy loss function. The model is trained for 100 epochs without shuffling because shuffling confuses the network connection. The labels are encoded using Sklearn's label_encoder. The adjacency matrix is constructed using Stellargraph which is Normalized according to [9]. We obtain train, test, validation splits in the form of indices and feed them into the network during training. Here, we experiment using all the User features available in the dataset.

We proceed with testing a network-based embedding for each User as an alternative to GloVe called Node2Vec [12]. We use Karateclub's BiasedRandomwalker [10] to obtain the embedding for each user node and due to memory limitation, we only used a sub-sampled graph from the original graph containing 10,000 user nodes. The BiasedRandomwalker was run using the following parameters: walk_length=10, walk_number=5. Once the node neighborhood embedding is obtained, we use gensim's Word2vec to embed the information for each user node using hierarchical softmax and skip-gram. The obtained embedding is mapped to the respective users. We then obtain the adjacency matrix for the sub-graph and concatenate with the embeddings and feed it into the vanilla GCN model with the same settings used previously. The labels are encoded, and train/test/validation split are done the same as before.

We use GraphSage as a final benchmark model with two different experimental setups. First, we use only the GloVe embedding with GraphSage which helps evaluate GraphSage's aggregation method which is like Node2Vec as it creates an embedding based on the neighboring users for each user node. Next, we use all the features from the original dataset with GraphSage. For the experiment we utilize GraphSage's implementation in [7] which uses mean aggregator to learn neighborhood features and uses Relu for activation. The approach uses Stochastic Gradient Descent to optimize the loss function.

We calculate Precision, Recall and F-1 score for all the methods and compare them.

Method	Precision	Recall	F-1
AdaBoost	0.24	0.84	0.37
GradBoost	0.38	0.75	0.5
GCN + GloVe	0.73	0.63	0.67
GCN + Node2Vec	0.4	0.45	0.42
GraphSage + GloVe	0.51	0.64	0.64
GraphSage + All	0.55	0.68	0.69

Table 1. Performance Metrics of all the methods

Table 1 showcases the Precision, Recall and F-1 score of all the experiments. The gradient boosting methods AdaBoost and

GradBoost show higher values of recall, but the overall precision and F-1 score are lower compared to graph-based methods. Looking at the performance of the graph-based methods, the vanilla GCN has one of the best performance metrics when used with GloVe embedding. The GCN variant with Node2Vec has very low metrics because it was only trained on a subsampled graph of 10,000 nodes due to hardware limitations as Node2Vec produces vectors that require more memory and computation. GraphSage when used with only the GloVe embedding has lower performance values compared to GCN. But when all the User's information such as Empath, Network details, Twitter features are added, it produces the best metrics than all the methods. This shows that node related information is crucial in addition to the network features to perform network-based classification.

5. CONCLUSIONS AND FUTURE WORK

The paper focused on showcasing the effectiveness of graph-based methods such as graph Convolutions, Graph embeddings for graph data in comparison to traditional Machine learning methods that reduce multidimensional graph data into Euclidian features and perform computation on them. This is clearly showcased from the metrics as Graph based methods perform almost twice as better than the non-graph counterparts.

Secondly, in terms of the features to be used by the graph-based methods, we noticed how graph-based methods require not only the network related details but additional features pertaining to the individual node (i.e.) User's topics of interest, Account information, etc. and it surely boosted the precision, recall and F-1 score to a notable amount. GraphSage's embedding technique which takes into consideration of neighboring nodes' features and propagates them proved to be effective in determining hate which inherently propagates through a real-world network and not just to the immediate neighbors which is what vanilla GCN computes as it is only limited to the adjacent nodes' features. While we tried including Node2Vec to overcome GCN's limitation of propagating further nodes, due to constraint of resources, we couldn't explore its full extent, but GraphSage does the same job more efficiently. But still, training GCN with Node2Vec on a subsampled graph with 90% less nodes have better performance than the non-graph-based Machine Learning methods. So, in this work, we were able to effectively utilize graph-based method for a more compelling topic which requires such approach, and we also showed their effectiveness and can also be extended to problems like this.

The main limitation of Graph based methods is the problem of inductive learning where the entire graph is used to train the model and when a new node is added, the whole graph needs to be retrained. GCN uses this approach and when used with an embedding method like Node2Vec which requires more computation capabilities, it is tedious to retrain the model every time. Although GraphSage is a transductive method that does not require retraining, more complex graphs require advanced architectures like attention-based graph methods that are computationally intensive. Also, the data imbalance is high as there was only 4% labelled nodes and with more labelled data, the graph methods can learn more information. Also, classifying hate as a binary classification problem is limiting and must be enhanced into a hate spectrum when used in real-world problems and better decisions can be made.

For further enhancements, testing advanced graph methods based on attention which is a better way to learn neighboring nodes in a network is an interesting track. I would also like to use the raw tweets and determine different embeddings and features from the raw text to determine hate. Also, with access to more computational power, GCN with Node2Vec can be fully tested and a certain baseline can be obtained. Since GraphSage models are transferrable to a completely different problem, using the feature weights of current model into a similar problem and testing it will be an interesting area of research.

6. REFERENCES

[1] Detecting Hate Speech in Social Media - Shervin Malmasi, Marcos Zampieri

https://doi.org/10.26615/978-954-452-049-6_062

[2] Analyzing the targets of hate in online social media.

Silva, L. A.; Mondal, M.; Correa, D.; Benevenuto, F.; and Weber, I. 2016. arXiv:1603.07709 **[cs.SI]**

[3] A Survey on Automatic Detection of Hate Speech in Text Paula Fortuna and Sérgio Nunes. 2018. A Survey on Automatic Detection of Hate Speech in Text. ACM Comput. Surv. 51, 4, Article 85 (July 2019), 30 pages. DOI:https://doi.org/10.1145/3232676

[4] Using Knowledge Graphs to improve Hate Speech Detection Poojitha Maheshappa, Binny Mathew, and Punyajoy Saha. 2021. Using Knowledge Graphs to improve Hate Speech Detection. In 8th ACM IKDD CODS and 26th COMAD (CODS COMAD 2021). Association for Computing Machinery, New York, NY, USA, 430. DOI:https://doi.org/10.1145/3430984.3431072

[5] M. Beatty, "Graph-Based Methods to Detect Hate Speech Diffusion on Twitter," *2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2020, pp. 502-506, doi: 10.1109/ASONAM49781.2020.9381473.

[6] Characterizing and Detecting Hateful Users on Twitter Manoel Horta Ribeiro, Pedro H. Calais, Yuri A. Santos, Virg´ılio A. F. Almeida, Wagner Meira Jr.)

[7] https://github.com/manoelhortaribeiro/GraphSageHatefulUsers

[8] https://www.kaggle.com/manoelribeiro/hateful-users-on-twitter

[9] Semi-Supervised Classification with Graph Convolutional Networks.

Thomas Kipf, Max Welling arXiv:1609.02907v4 [cs.LG]

[10] https://karateclub.readthedocs.io/en/latest/

[11] http://snap.stanford.edu/graphsage/

[12] https://snap.stanford.edu/node2vec/

[13] Empath: Understanding Topic Signals in Large-Scale Text
Ethan Fast, Binbin Chen, Michael Bernstein

arXiv:1602.06979 **[cs.CL]**

[14] Characterizing and Detecting Hateful Users on Twitter

Ribeiro, M., Calais, P., Santos, Y., Almeida, V. and Meira Jr., W. 2018. Characterizing and Detecting Hateful Users on Twitter. *Proceedings of the International AAAI Conference on Web and Social Media*. 12, 1 (Jun. 2018).