FRI Natural language processing course 2021
Project report

University *of Ljubljana*
Faculty *of Computer and*
Information Science

Offensive language exploratory analysis

Tomaž Martinčič, Dimitar Stefanov and Žiga Trojer

Abstract

Offensive speech is often present on the Internet. Many times this is not desirable on the pages, or it is forbidden, so we want to detect it automatically. This task has attracted a lot of attention recently, and as a result, many language processing techniques and algorithms have been developed. In this report, we present the capabilities of a number of traditional and state-of-the-art methods. Furthermore, we try to produce meaningful visualizations of the peculiarities of the different types of hate speech, as well as construct an offensive language schema based on the drawn conclusions from the analyses.

Keywords
hate speech, offensive language, exploratory analysis

Advisors: prof. dr. Slavko Žitnik

Introduction

Annotated offensive language datasets are essential to perform any kind of offensive language exploratory analysis. They enable us to categorize texts according to their offensive content automatically. The dataset should be such that it contains as much different hate content as possible, as our main goal is to explain these classes. We found many data sets that were promising, but at the end we analyzed data from the most prominent social media platform, Twitter. Some examples of offensive content are hate speech, obscene, vulgar, cyberbullying and many more. In this paper, we explore many of the existing methods that enable us to analyze hate speech. We also try to highlight the similarities and differences of these methods, and in addition, we try to present the differences between various hate content with visualizations.

Offensive language categories of interest

In the rest of the work, if not explicitly said otherwise, we will be interested in exploring the main characteristics and interrelationships of the following types of hate speech:

- Abusive
- Hateful
- Discredit
- Cyberbullying
- Harassment
- Profane
- Slur
- Vulgar
- Obscene
- Hostile
- Racist
- Homophobe
- Offensive
- Threat
- Sexist
- Insult

Methods

Our first task was to find as many offensive speech data sets as possible. Out of the 65 datasets for which we were given descriptions and references, we decided to focus on the 25 English language datasets. We had a lot of problems with the protection of personal data as some data sets were not accessible. Many data sets consisted of Twitter messages, but only tweet IDs were given - scrape data is required to analyze these tweets. Another problem that we encountered was that we found some data sets and most of them had only binary class - hate speech or not.In addition, the datasets were obtained from different media, so formatting issues need to be taken into account as well in some cases.

We have documented our whole analysis of the quality and suitability of the datasets in a Github repository[1]. Additionally, on Kagle we have come across one useful Twitter dataset. This dataset is rather abundant, hence we have taken advantage of it on multiple occasions with both traditional and newer methods.

While searching for ideas, we found several techniques

[1] https://github.com/tm1897/nlp_offensive_lang

and methods that were already used for this kind of analysis. Here we provide a list of the models we considered for our project:

1. the traditional technique TF-IDF (Term Frequency - Inverse Document Frequency)
2. non-contextual pre-trained word embeddings:
   (a) Word2Vec model,
   (b) GloVe (Global Vectors for Words Representations),
   (c) fastText model,
3. contextual pre-trained word embeddings:
   (a) BERT (Bidirectional Encoder Representations from Transformers
   (b) ELMo (Embeddings from Language Models)

Firstly, we tried to generate insights from the chosen datasets by using the method TF-IDF. Once having exhausted the possibilities of TF-IDF, we proceeded with the more recent models mentioned in the list above.

In terms of the visualization part, we performed PCA (Principal Component Analysis), MDS (Multi-Dimensional Scaling) and t-SNE (t-distributed stochastic neighbor embedding) on the embeddings of all of the state-of-the-art approaches we considered. Additionally, for the pre-trained BERT model a dendrogram was obtained.

In the continuation, we delve deeper into the specifics of the applied models, elaborate on their pros and cons, as well as visualize these observations accordingly.

Twitter dataset

We already tested [2] some basic methods on one of the datasets. The dataset contains 149,823 tweets, which are categorized into 5 categories of offensive language. First, we had to clean the tweets by removing URLs, and user tags. After that, we encoded tweets by the TD-IDF method. So each document was represented with a vector. We then performed K-Means clustering with a different number of clusters (3, 5, 10, 15) and checked the most frequent words in each cluster. With even that simple methods we were able to notice that most of the clusters do make sense.

We noticed that the current stemmer and stop words list don't work well with this kind of text, as the tweets contain a lot of grammatically incorrect words and language variations, such as slang.

Reddit dataset

We also tested the TF-IDF method on a Reddit dataset containing 5000 conversations (each conversation has approximately 4 replies). We first preprocessed the conversations by removing tags and unnecessary characters. Then, we considered a corpus of English stopwords, so that we don't have to search for their TF-IDF value. Lastly, we performed the TF-IDF vectorization. By looking at the 50 most frequent words in this dataset, we can see that words representing offensive language are present there. Unfortunately, we came across the same problem as above: we still track a lot of common everyday words, short forms and slang which don't help in our analysis.

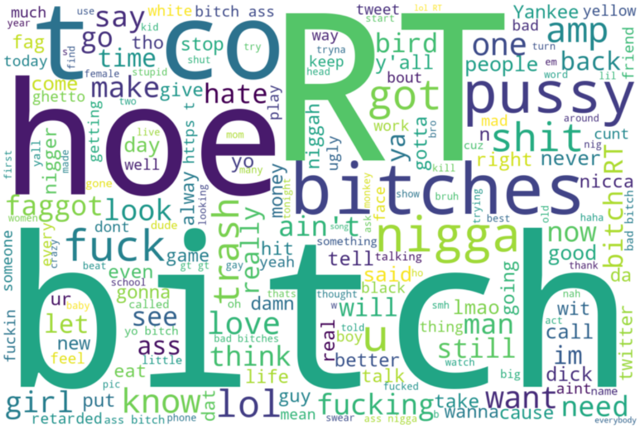

Figure 1. Word cloud from tweets with offensive language.

Word2Vec Model

We utilized the above described Twitter dataset to train our own *Word2Vec* model on it. The embedding size of the words we chose was 100, and during the training we considered a window of size 5. In terms of clustering, we tried to divide the words into 5 categories: homophobe, sexist, racist, religion and hateful, as these classes were also considered in the dataset itself. By having a dataset specific to hate speech recognition, we thought a reasonable Word2Vec model could be trained. Unfortunately, it turned out we needed additional text, additional tweets to capture deeper patterns. As a result, when analyzing the most similar words to our terms of interest, we notice words which are not necessarily highly related to the category they have been assigned. Still, as it can be observed in Figure 2, aside from the one mixed cluster, we obtain well defined clusters for each of the terms.

Pre-trained fastText model on the entire Urban Dictionary [3]

Pre-trained word embeddings represent a powerful resource for doing exploratory language analysis. And, there exist a great deal of them, however most of them have been trained on corpora which contain text written in the standard form of a language. On the other hand, the vast majority of hate speech conversations take place on social networks where slang is also present. Consequently, these pre-trained models might not always be able to catch some patterns peculiar to a particular type of hate speech. Therefore, we have decided to

[2] These initial analyses have been added to the *develop* branch of our repository.

[3] Urban Dictionary is a crowdsourced online dictionary for slang words and phrases. As of 2014, the dictionary had over seven million definitions. Their official webpage is: https://www.urbandictionary.com/.

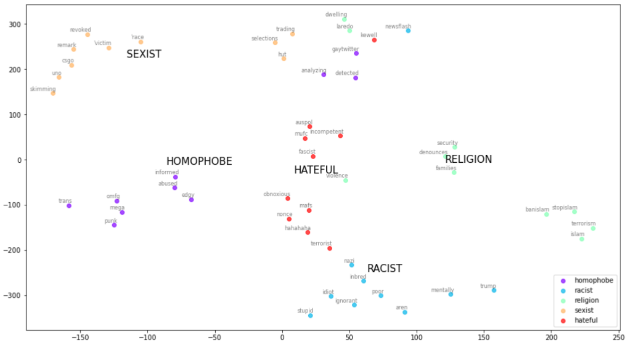

Figure 2. A Word2Vec model was trained on a hate speech dataset from Twitter. The obtained word clusters after utilizing t-SNE for dimensionality reduction are presented in this figure.

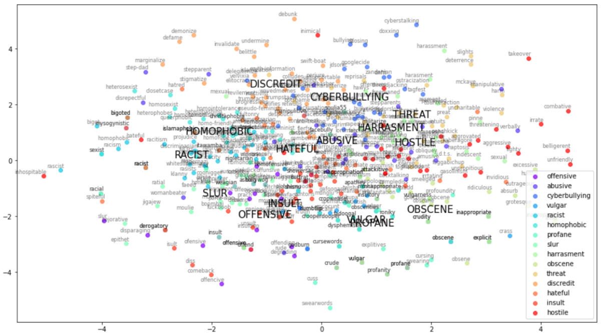

Figure 3. Multidimensional scaling performed on the embedding vectors in **UD-base**. Clusters tend to overlap, but in an explainable way.

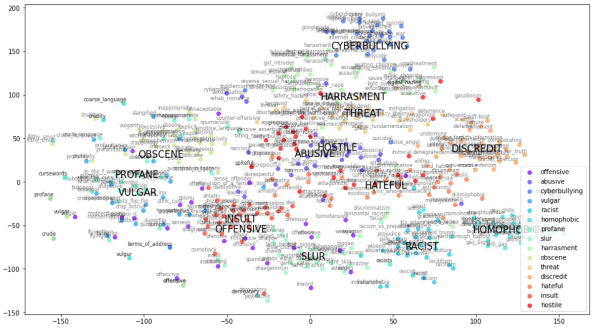

Figure 4. t-SNE was used to visualize selected word embeddings from **UD-phrase**. Created clusters were well described and appropriately grouped.

take advantage of a pre-trained fastText model on the entirety of Urban Dictionary (all terms, definitions, examples, and tags were treated as running text) presented in [1]. This dictionary has an abundance of slang words and phrases which we believed could help shed a different light on the interpretation of hate speech.

The authors in [1] trained two separate fastText models. For the first model called **UD-base**, they tokenized the entire corpus without any explicit guidance on how to join phrases during training time. Also, they allowed the model to learn representations for word n-grams with a length of up to five. In **UD-phrase**, the second group of word embeddings, they considered all phrases and their various occurences throughout Urban Dictionary.

We utilized these two models by finding the 30 most similar words to each one of our categories of interest. However, during this search, we excluded all words having the same lemma or stem as a term in our list. By this procedure, we hoped to obtain more insightful and fair visualizations of the different word clusters.

As in the case of the word2vec model we trained, we used dimensionality reduction techniques to try extract meaningful patterns from the embeddings of size 300. In Figures 3 and 4, we show two of the created visualizations. Additional visualizations can be viewed at [4].

In Figure 3, multidimensional scaling is performed on the embeddings from the model **UD-base**. At first glance, it might seem as if all the clusters are merged, however there exist pattterns. For instance, terms *threat, harrasment, hostile* are very close together, but that is not something we would not expect, because even humans would often classify a statement in each one of these hate speech types simultaneously. The same applies to *vulgar* and *profane* which have a significant overlap in our visualization.

On the other hand, in the t-SNE visualization in Figure 4, we can easily notice distinct clusters. In addition, the grouping of the clusters is as one would assume it should be. The cluster of every term is surrounded by clusters of terms representing similar kinds of hatred.

To conclude, it should be noted that most terms' list of most similar words according to the models **UD-base** and **UD-phrase** is more adequate than the one produced by word2vec models pre-trained on texts of standard English. We believe the reason for this lies in the fact that Urban Dictionary has a much larger record of the strong language used in hate speech expressed on social networks. Nevertheless, there are also categories which are better described by the pre-trained embeddings found in the widely used libraries.

GLoVe Model

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space [2].

The starting point for vector learning is the ratio of co-occurrence probabilities. Compared to probabilities, the ratio better distinguish relevant words from irrelevant ones and also is able to better discriminate between the two relevant words.

Authors of GloVe [2] found out that on the same corpus, vocabulary, window size and training time, `GloVe` consistently outperforms `word2vec`. It achieves better results faster, and also obtains the best results irrespective of speed.

[4] https://github.com/tm1897/nlp_offensive_lang/blob/develop/notebooks/pretrained_fasttext_model.ipynb

Pre-trained Glove model on the Twitter data [5]

First we converted GloVe file containing the word embeddings to *word2vec* format for convenience of use - we have used `glove2word2vec` function. Then we took 30 most similar words to each of the given terms and also excluded all words having the same lemma/stem as a given term. Then we created visualizations of different word clusters. From Figures 5, 6 and 7 we can observe clusters of those. We can see that clusters make sense and we also observe that if two clusters are close together using one of the methods (MDS, t-SNE or PCA), they are close using other two. The most interesting visualization seems the one on Figure 6, where we observe very clean distinction between some terms and some overlapping of terms in the middle. Those in the middle are the ones that a tweet may belong to more of the categories, i.e. *hostile* and *hateful*.

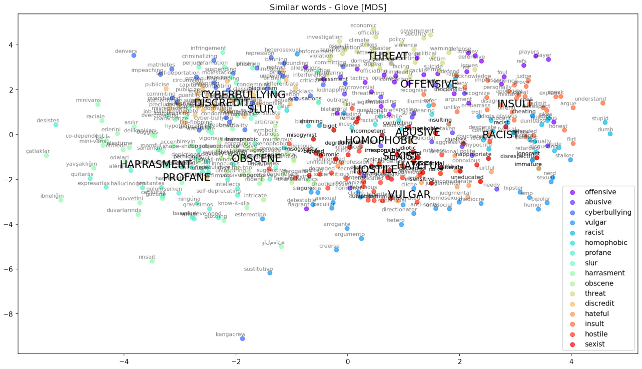

Figure 5. Similar words for GloVe model - MDS.

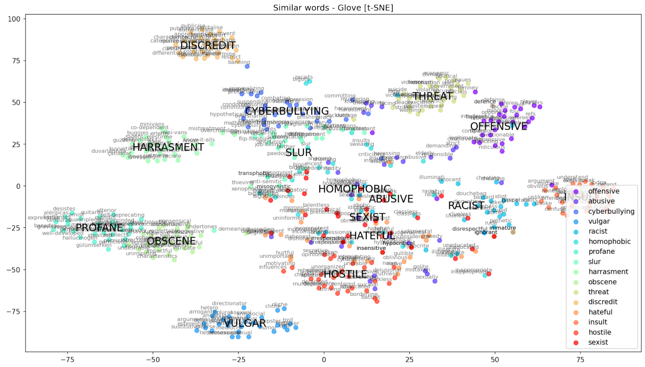

Figure 6. Similar words for GloVe model - t-SNE

BERT

We used BERT for contextualized word embedding. To extract information from tweets, we concatenated each tweet string with ”This is category”, where the category is a category of hate speech (racism, sexism, etc.). Then we extracted vector embedding of the category at the end of each tweet, which also contained the context from the tweet. The method for

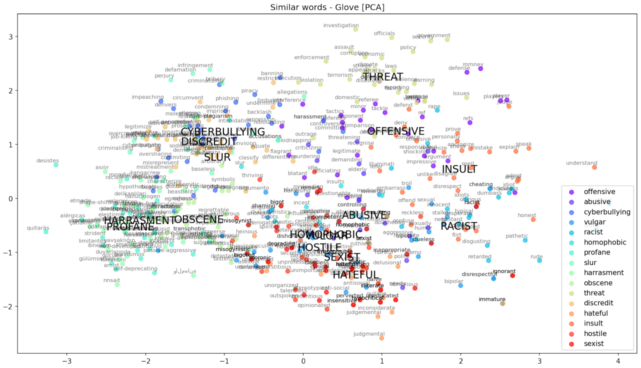

Figure 7. Similar words for GloVe model - PCA

word embedding with BERT is well explained in the Medium blog post [3]. For each category, we then computed the mean vector. Using cosine distance we plot the dendrogram which is shown in Figure 8. Racist and religious hate is the most similar to each other. Then homophobe and sexist. And in the last group, we have no hate and other hate tweets.

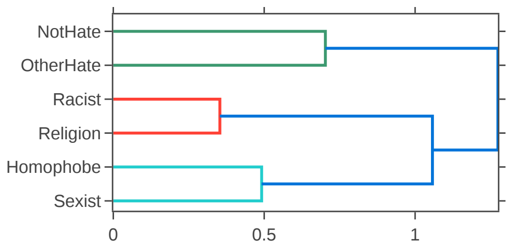

Figure 8. Dendrogram of hate speech categories average vectors by cosine distance.

We also extracted one tweet from each category, which is the most similar (by cosine similarity) to the average vector of its category:

- **NotHate** @dalehay there could not get you as a redneck what you were saying the other day https://t.co/aHcbZRSFJo
- **OtherHate** white men agreeing with each other on their retarded views https://t.co/59DIR3U6Ag
- **Racist** maybe if whoever this racist cunt is stanned seulgi! #weloveyouzach https://t.co/mRUiTDaysQ
- **Religion** Islam as a religion is garbage, it is mostly fanatical terrorism. My next posts are about Islamic art, not religion. https://t.co/LISimK7794
- **Homophobe** Why tf Tracy so cute???? No homo though cause I’m not on that faggot shit https://t.co/WNdLmxDlji
- **Sexist** @poet_blu Sex and not be a cunt? Can’t process that... https://t.co/ulWBtipDNn

[5] 2 Billion tweets, 27 Billion tokens, 1.2 Million vocab, 25 dimensional vectors, accessible on https://nlp.stanford.edu/projects/glove/.

Discussion

In the continuation of our work on this project, we would like to explore the following ideas:

- create a graph to illustrate the relations among different hate speech types, but also consider other new types of visualizations suitable for our task
- perform topic modelling
- try ConceptNet Numberbatch pre-computed embeddings [6], because they have showed promising results at competitions (SemEval 2017 [7])
- these embeddings stem from ConceptNet [8], which is a quite powerful tool, so we intend to familiarize further with this neural network and utilize some of the ideas behind it to formulate our knowledge base
- consider the contextual model ELMo
- generate a final offensive language schema.

References

[1] S. R. Wilson, W. Magdy, B. McGillivray, K. Garimella, and G Tyson. Urban dictionary embeddings for slang nlp applications. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, 05 2020.

[2] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[3] Andreas Pogiatzis. Nlp: Contextualized word embeddings from bert. https://towardsdatascience.com/nlp-extract-contextualized-word-embeddings-from-bert-keras-tf-67ef29f60a7b.

[6] https://github.com/commonsense/conceptnet-numberbatch

[7] https://alt.qcri.org/semeval2017/

[8] https://conceptnet.io/