HATESPEECH DETECTION

Lucian Istrati[1], **Adrian-Stefan Miclaus**[1], **Eugen-Cristian Bleotiu**[1]

[1]Faculty of Mathematics and Computer Science, University of Bucharest

An Important Problem

Nowadays people interact more and more in the digital world and unfortunately this also gave a voice to some people that instigate towards violenece, hatred or discrimination. In order to tackle this issues many social media apps came up with engines that can predict whether a users spreads anything that falls under oen of these categories: mysogyny, racism and hate-speech.

Dataset description

In our case we had to classify if an author is spreading hate-speech or not based on a feed of 200 of his tweets. The task together with the dataset was taken from the PAN 2021 competition "Profiling Hate Speech Spreaders on Twitter"[3]. The dataset provided two separate sets of 100 English-speaking authors and 100 Spanish-speaking author(this paper treats just the English subdataset).

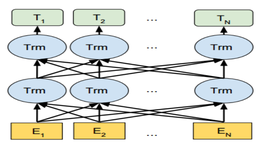
Method

The data preprocessing pipeline consists of the following steps:

- Emoji & Emoticon identification and translation;
- SpellChecking by correcting typos(either intended or unintended) as a way of normalizing data;
- Data Cleansing by removing urls, user mentions, hashtags etc;
- Stemming for certain algorithms, also as a way of normalizing data.

The dataset train-test split used was an 80-20 one with no validation set, due to the small size of the dataset. The best architecture that we worked with for the BERT Embedding for feature extraction that was given to a FCNN:

- fc1: 768 neurons fc2: 1024 neurons;
- optimizer: Adam(learnig rate = 1e-3);
- epochs: 10.



Experiments

- One of the first experiments that was tried came up as a necessity of augmenting the initial dataset as we had two different datasets(one in English and one in Spanish) and we wanted to enlarge the English dataset by translating the spanish one into english via different translation packages like translate, pytranslate, goslate, googletrans and vicevers, but this failed due to 403 Forbidden Error after too many requests were sent to the translation server.
- Another experiment that we tried was a transfer learning approach by training on a BERT-based model on a dataset mentioned by Davidson et al. in their work here [1]. This contains 25.000 tweets manually labelled with a score between 0 and 8 for a hateful tweet, score that is further binarized to a 0 and 1 problem in order to classify between hate and non-hate. After training the model we predicted every single tweet of each author and then we aggregated by averaging all of his tweets. The accuracy obtained was around 55%.
- We also experimented on trying different parameters and hyper-parameters (width/depth of the neural networks, number of epochs, learning rate or the optimizer), but definitely the experiment that lead us to the best results was the random sampling of the tweet. This idea standed on quite strong theoretical foothold because statistically if a user is spreading hate-speech or not, there is a good likelihood that just by sampling a subset of his tweets there are certain features that should be observed about his online behaviour. There is also an empirical reason that sustained this idea, because after reading tweets of some authors it was observed that if an author spreads hate-speech, just a minority(a quite consistent on) of his tweets are truly spreading hate speech, with the rest being normal tweets that could be posted by somebody else. The experiment consisted of trying to create n groups of tweets of an author, with n varying from 1 to 10, which translates to keeping all of his 200 tweets in a single groups and creating 10 groups of 20 tweets each, respectively(all the tweets that belong to a group are merged, thus forming a datapoint comprised of concatenated tweets and the label 0/1).

Results		
Model	Number of epochs	Accuracy
Bert	10	75%
Transformer	35	66%
Log. Regression (BoW)	-	65%
TextRNN	10	64%
Seq2Seq Attention	15	56%
CharCNN	10	56%
TextCNN	15	54%
FastText	30	53%

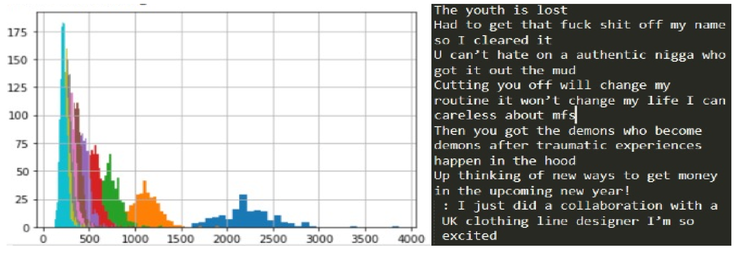


Fig. 3: Distribution of data by number of tokens when using k groups(k from 1 to 10) and a group of hateful tweets classified as hateful

K groups experiment

Results			
Num groups	Accuracy	Num groups	Accuracy
1	53%	6	58%
2	58%	7	65%
3	68%	8	61%
4	57%	9	62%
5	75%	10	64%

Conclusions

- Before summing-up some conclusions of this project we firstly wanted to point out that with an extremely small dataset also comes with a low upper bound on the performances that can be achieved by any machine learning model(unless the right way of data preprocessing is found for the best model with the appropriate hyper-parameters, in which case an accuracy closer to state-of-the-art results in competitions such as [4] or [2] can be met).
- However we still managed to achieve over-baseline performances. First part that is essential for these results stands for the way of preprocessing our data(especially with the translation of emojis and emoticons into meaningful words in English; but also with the text cleansing part by removing user mentions, urls, hashtags or any other irrelevant component that may be found in tweets or by using the SpellChecker in order to correct typos that occur quite frequently in web texts). The second part is, of course, the architecture that consists of a fully connected neural network that receives an embedding of the text via Google BERT. The performances of this model are similar to a simple model like Logistic Regression after fitting data using a Bag-of-Words approach, this is due to two reasons: a scarcity of datapoints and a space-limitation put on our word-embedding(as the feed of an author is on average consisting of 2140 tokens and the embedding provided by Google BERT is accommodating 512 tokens at most, thus losing the power to represent on average 75% of any data-sample, setting aside feeds as long as 3000 tokens where even a smaller percent of the data is represented).
- Another conclusion that we found after the failed translating experiment is the need to come up with different ways of translating a dataset that avoid timed-out requests from Google Translate server(possibly with translating that use lexicon-based approaches).

Acknowledgements

We wanted to thank our mentor for usefull discussions and advices.

References

[1] " Automated Hate Speech Detection and the Problem of Offensive Language". In: (2017).

[2] *Hate Eval Competition*. urlhttps://www.aclweb.org/anthology/S19-2007/. [Online; accessed 17-April-2021].

[3] *Profiling Hate Speech Spreaders on Twitter*. https://pan.webis.de/clef21/pan21-web/author-profiling.html. [Online; accessed 14-April-2021]. 2021.