Hate Speech Detection on Social Networks with Application to TikTok

EURECOM

Emerson CARDOSO, Sameer HANS

Supervisor: Paolo Papotti, Gilbert Badaro

Email: {FirsName.LastName}@eurecom.fr

Project Description:

TikTok was launched to the international market in 2017 and thanks to its simple concept and GUI, it became extremely popular especially among teenagers. In effect, more than 2 billion downloads for the application were recorded in August 2020 [1]. Unfortunately, a lot of content on TikTok includes abusive language such as swear words, hate speech and racism, in addition to indecent behaviour. An in-depth analysis was conducted on 1030 videos, an equivalent of almost 8-hour of content, and multiple aspects of extremism and hatred speech were depicted [2]. By definition, a hate speech is an abusive or threatening speech or writing that expresses prejudice against a particular group, especially based on race, religion, or sexual orientation.

Introduction

The main objective of this project is to create an end-to-end approach to detect hate speech and offensive language in videos present in the social media platforms. The detection of the speech can be performed by analysing the audio content or the text present in the video. For that, we aim to create an API powered by Machine Learning model to detect hate speech in any selected video. For the start, we focus on two of the main social media platforms used to share video content; TikTok and YouTube. The focus of this project would be the sentiment analysis of the text that we obtain from the video.

The project is available here:

https://github.com/EmersonAlvaro/HateSpeach2022.git

Background

Hate Speech can be defined as a broad term that is used to identify a wide variety of expressions. In general, however, it refers to words or symbols that are offensive, intimidating, or harassing, and/or that incite violence, hatred, or discrimination on the basis of a person's race, religion, gender, sexual orientation, or another distinguishing status[1]. Hate speech is integrally linked to incitement to violence and daily infringements on human rights, often targeting the most vulnerable people and groups. Below we summarize the Hate Speech definition on two of the main video content share platforms.

- According to Tik Tok Policy "*hate speech or behaviour as content that attacks, threatens, incites violence against, or otherwise dehumanizes an individual or a group on the basis of the following protected attributes: race, ethnicity, national origin ,religion, caste, sexual orientation, sex, gender, gender identity, serious disease ,disability, immigration status*"[2].
- According to YouTube Policy "*hate speech when it incites hatred or violence against groups based on protected attributes such as age, gender, race, caste, religion, sexual orientation, or veteran status. This policy also includes common forms of online hate such as dehumanizing members of these groups; characterizing them as inherently inferior or ill; promoting hateful ideology like Naziism; promoting conspiracy theories about these groups; or denying that well-documented violent events took place, like a school shooting*[3]*"*

Summarize the definition give above we can say that what characterize hate speech is any speech that instigate violence or hate towards specific group.

The Internet has had a positive influence on society. For example, it helps us to communicate easily and to share knowledge on all kinds of important topics efficiently: from the treatment of disease to disaster relief. But the Internet has also broadened the potential for harm. Being able to communicate with a mass audience has meant that the way we engage with politics, public affairs and each other has also changed. Hateful messages and incitements to violence are distributed and amplified on social media in ways that were not previously possible.[4]

The business model of most social media companies is built on drawing attention, and given that offensive speech often attracts attention, it can become more audible on social media than it might on traditional mass media.

Hate speech online is not intrinsically different from hate speech offline. However, it differs in the nature of the interactions in which it takes place/occurs, as well as in the use and spread of specific words, accusations and conspiracy theories that can evolve, peak, and fade very quickly. Online hate speech can be produced and spread at low cost, experiencing vastly distinct levels of exposure depending on the popularity of the post and can be posted cross-nationally. Hate speech online can also be available for longer and go through waves of popularity, connect with new networks, or reappear, as well as be anonymous[5]

Part I: Software Engineering

In this part we will describe the API and the User interface, furthermore we describe how we improve the performance of our API by using Thread.

A. Introduction

This modules contains two main block, the application made for any user which take a link of o video (from TikTok or YouTube) then send a request to the API, the REST API which accept request, process and send back a response. The following image describe the general architecture implemented.

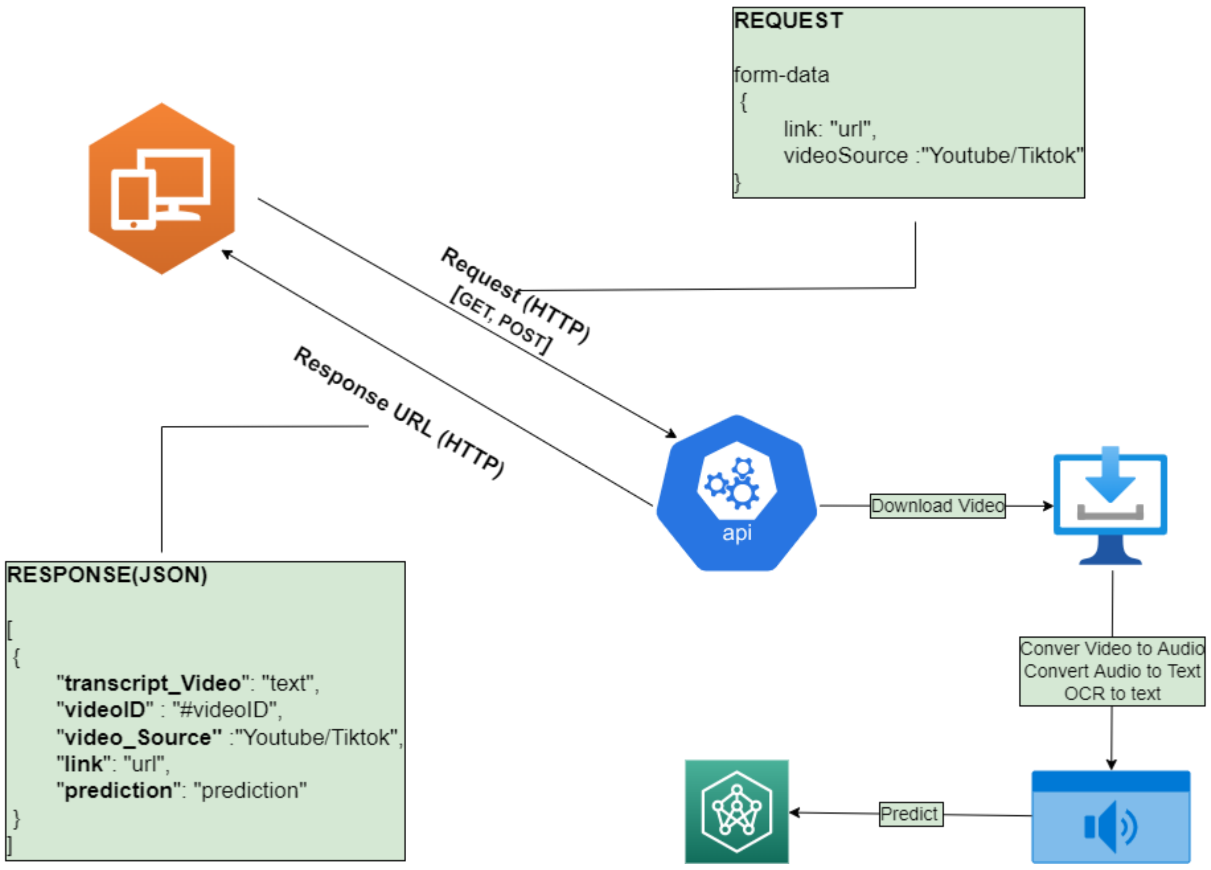


Figure 1: Architecture of our API

B. Interface Design and functionality

The user inference allows any user to verify if any TikTok or YouTube Video has Hate Speech by simples insert the link of the video. When the user inserts the link of the video and submit the interface make a http request to serve in order to get the prediction. To create the user interface we used Vanilla JavaScript, Html and CSS.

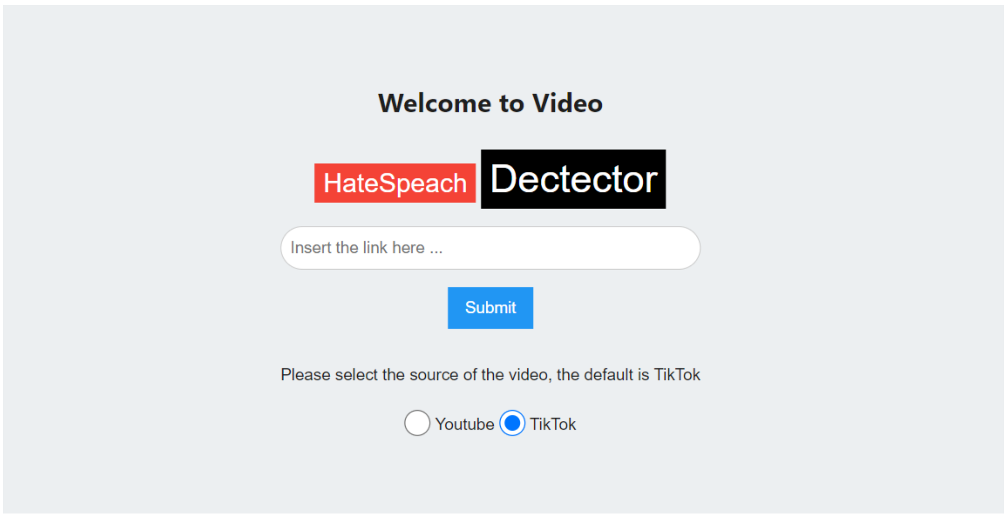
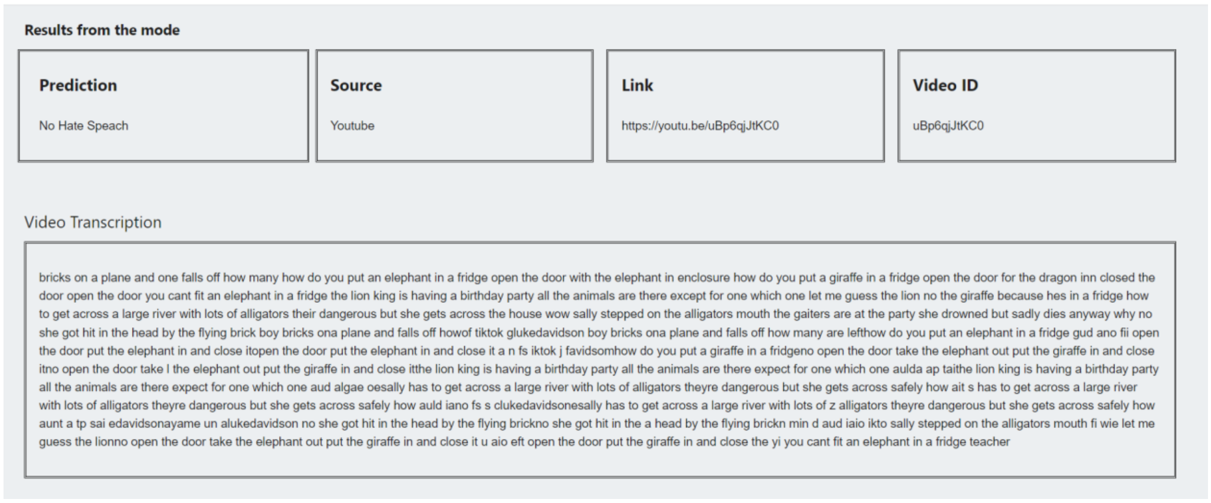


Figure 2: Homepage of User Interface

After the user make a request the following(figure below) show what kind of output the user get from our API.



Results from the mode

Prediction	Source	Link	Video ID
No Hate Speach	Youtube	https://youtu.be/uBp6qjJtKC0	uBp6qjJtKC0

Video Transcription

bricks on a plane and one falls off how many how do you put an elephant in a fridge open the door with the elephant in enclosure how do you put a giraffe in a fridge open the door for the dragon inn closed the door open the door you cant fit an elephant in a fridge the lion king is having a birthday party all the animals are there except for one which one let me guess the lion no the giraffe because hes in a fridge how to get across a large river with lots of alligators their dangerous but she gets across the house wow sally stepped on the alligators mouth the gaiters are at the party she drowned but sadly dies anyway why no she got hit in the head by the flying brick boy bricks ona plane and falls off howof tiktok glukedavidson boy bricks ona plane and falls off how many are lefthow do you put an elephant in a fridge gud ano fii open the door put the elephant in and close itopen the door put the elephant in and close it a n fs iktok j favidsomhow do you put a giraffe in a fridgeno open the door take the elephant out put the giraffe in and close itno open the door take l the elephant out put the giraffe in and close itthe lion king is having a birthday party all the animals are there expect for one which one aulda ap taithe lion king is having a birthday party all the animals are there expect for one which one aud algae oesally has to get across a large river with lots of alligators theyre dangerous but she gets across safely how ait s has to get across a large river with lots of alligators theyre dangerous but she gets across safely how auld iano fs s clukedavidsonesally has to get across a large river with lots of z alligators theyre dangerous but she gets across safely how aunt a tp sai edavidsonayame un alukedavidson no she got hit in the head by the flying brickno she got hit in the a head by the flying brickn min d aud iaio ikto sally stepped on the alligators mouth fi wie let me guess the lionno open the door take the elephant out put the giraffe in and close it u aio eft open the door put the giraffe in and close the yi you cant fit an elephant in a fridge teacher

Figure 3: Result from the API request

C. Rest API

We create lightweight API using bottle framework which is distributed as a single file module and has no dependencies other than the Python[6]. The API them has five main functionality which are:

i. **Download Video** – This module is responsible for downloading the video from source and save it temporally on the disk.

ii. **Convert Video to Audio** – Read the video file and convert the video into WAV audio file.

iii. **Convert Audio to Text** – ready the WAV file and create small chunk of the audio and convert the audio file into audio.

iv. **Video OCR** – this module is responsible for reading the video file and detect any sentence that can appear on the video.

v. **Model Prediction** – After the above steps are completed on txt file is generate with all the text in the video (Audio + OCR text) which them is feed in a Machine Learning model to predict if the text is Hate Speech, Offensive Language or Neutral.

D. Performance

In order to increase the performance of our API and reduce the time that a user have to wait to get a response from the API, we redesign some of our module (Convert Audio and Video OCR) to use threads in order to speed up thing, we chose these two modules because they can be done concurrently. The following experiment was made in a computer with the following specs:

- Processor: Intel(R) Core (TM) i5-10210U CPU @ 1.60GHz (8 CPUs), ~2.1GHz
- Memory: 8192MB RAM
- Available OS Memory: 8026MB RAM
- Page File: 20406MB used, 5027MB available

We have the following results after redesigning our modules to use threads:

Table 1: Performance comparison

	Time in Seconds					
	Test1		**Test2**		**Test3**	
	Normal	Threads	Normal	Threads	Normal	Threads
Download Video	-	-	-	-	-	-
Convert Video to Audio	-	-	-	-	-	-
Convert Audio to Text	22.69	11.29	21.32	12.55	24.65	13.06
Convert OCR to Text	47.27	8.6	45.12	10.35	52.18	7.15
Prediction	-	-	-	-	-	-
Total	69.96	19.89	66.44	22.9	76.83	22.36

The uses of thread in these two main modules led to an increase of at least 50% of performance

Part II: Machine Learning

Machine Learning models can be a powerful tool to rapidly detect Hate Speech on social media, and consequently prevent the spread of hate online. In this section we describe about the Machine Learning Approach used to detect hate Speech based in three categories: Hate Speech, Not Hate Speech and Offensive Language.

A. Related Work

Many Studies has been conducted to detect offensive language, below we will describe 3 of research done to detect hate speech.

a) On research conducted by Gibert[7] named "*Hate Speech Dataset from a White Supremacy Forum*" they use a dataset consists of 10k sentences labelled as conveying hate speech or no, the
b) In the research conduct by Malmasi [8]named "*Detecting Hate Speech in Social Media*" the dataset was split in three main categories: Hate, No Hate and Others. They create a model by applying a standard lexical feature an a linear SVM classifier.
c) The approach [9] named "*Detecting Hate Speech on the World Wide Web*" by Hirschberg use seven categories as antisemitic, antiblack, anti-Asian, anti-woman, anti-Muslimism, antiimmigrant or other hate.

B. Dataset

We tried using and merging various datasets at first, like the Davidson and Conan, but some of them were not compatible with our model. For example, when we put a video with hate speech into our model, the model detected as it was not a hate speech. Hence, we had to experiment with a lot of datasets and finally found that to be safe (to avoid the false positives), we should choose a dataset that contained at least three classes; hate speech, offensive and not hate or offensive.

Dataset using Twitter data, is used to research hate-speech detection. The text is classified as: hate-speech, offensive language, and neither. Due to the nature of the study, it's important to note that this dataset contains text that can be considered racist, sexist, homophobic, or offensive. Our dataset has the labels distributed in the following way:

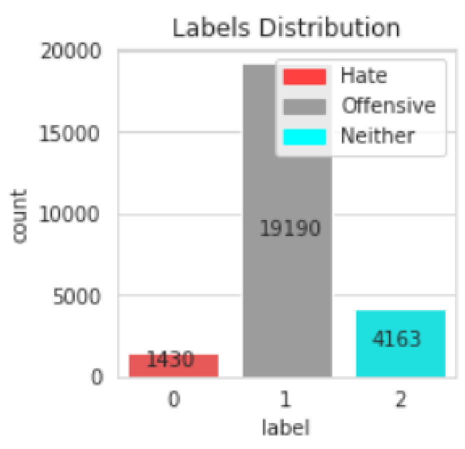


Figure 4: Dataset distribution

1. Pre-processing

We did the basic pre-processing steps for NLP on the “text” column. We did data cleaning starting from removing garbage noise such as punctuations, symbols, RT, mentions and emails. Then removed the stop words using nltk library. We performed tokenization (Tokenization is a method of breaking down a piece of text (the tweet) into smaller pieces (words). It converts a string of text into a list of strings, each string representing one of the tweet's words.) Then we did some chat words replacement (after obtaining the tokens); such as “ppl” to “people”.

Finally, we did Lemmatization. Lemmatization removes the grammar tense and restores the original form of each word. Lemmatization is the process of converting a word into its lemma form. For example, if we lemmatize the word "books", we get the term "book".

C. Models

1. Models Considered:

Considering the different models, the SVM and Decision Tree models are trained well but while it predicts, it overfits. This may be because of the hyperparameters which could have been better tuned. Then we tried more advanced models like LSTM and Bert Classifier from Transformers. The LSTM did not perform well. We were expecting it to perform better but we think that because of our pre-processing there is not much necessity to remember things and then classify. If the scenario would have been remembering the sentences without the pre-processing we used, then maybe the LSTM would have performed better. The Bert Classifier from the transformer gave us the best results as it learns bidirectionally which was indeed efficient.

2. Model Selected:

Considering the accuracy and the auc of the different models, we selected the Bert Classifier from Transformers. We started with making a pytorch model of Bert transformer. The pytorch model showed several integration problems without our API. Then we switched to the Tensorflow version and rewrote the whole notebook to work with tensorflow instead of pytorch.

We experimented with many pre-trained models of BERT (cased and uncased). The uncased model worked better than the cased models. After performing various experiments, the following selections were done:

BERT model selected: https://tfhub.dev/tensorflow/small_bert/bert_en_uncased_L-4_H-512_A-8/1

Pre-process model auto selected: https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/1.

3. Feature Extraction:

BERT stands for Bidirectional Encoder Representations from Transformers. It is a pre-trained model which is trained bidirectionally from left to right and vice versa. It is basically used to extract high quality language features from the text data.

We have chosen the Bert Tokenizer as a pre-trained model as it is trained on a large corpus of unlabelled text including the entire Wikipedia (that is 2,500 million words!) and Book Corpus (800 million words) which allows us to get appropriate pre-trained weights. Using these pre-trained weights is easier to fine tune the model.

4. Transformer (+Bert):

A transformer is a deep learning model that adopts the mechanism of attention, weighing the influence of different parts of the input data. For our scenario we have used the Bert Classifier model with a pretrained Bert Tokenizer. We choose Bert models to pre-train the model as well as the classifier because it is a simple and empirically powerful model. Meanwhile, it is a "deep bidirectional" which means it learns information from both the left and the right side of a token's context during the training phase which is more efficient.

D. Evaluation and Performance

To evaluate our model, we use the concept of epochs and early stopping. We select 100 epochs to train the Bert transformer on our dataset. However, we use early stopping, that is, we stop the execution of training the classifier as soon as the validation accuracy starts to decrease after reaching a maximum value. We got the accuracy of 0.99 and 0.90 for the training and validation datasets respectively for the given scenario. We use different metrics such as Loss, accuracy. Performance comparison:

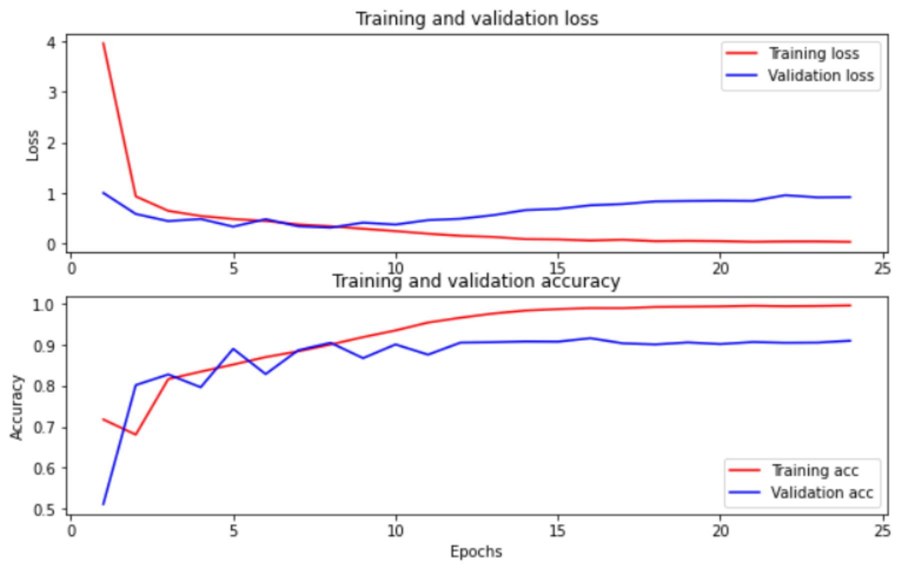


Figure 5: Model history during the training phase

We stop training our model after 25 epochs because the model started to overfit. We used to concept of early stopping to stop the epochs as soon as the validation accuracy started to deteriorate. For the hyperparameters, after doing many experiments with them, we considered the learning rate to be 3e-5 and the optimizer as adamw as it had an improved implementation of weight decay. On the test dataset we got the following result:

Table 2: Bert performance on test dataset

	Loss	**Accuracy**
Bert Model	0.84	0.9

Conclusion and Future Works

To conclude, through this project we discovered different kinds of embedding techniques such as bag-of-words and Bert and implemented several models like LSTM, Bert. We learned what kind of basic pre-processing one can do on typescripts data. And by comparing different models based on the accuracy and the auc we chose the transformer with the Bert model. Natural language is a large domain and we saw how difficult it was to tune models and pre-process the data in order to reach good auc and accuracy. We implemented python script to extract the textual data from a video to feed it into our model.

The main goal of our project was mostly satisfied, to classify a video as hate speech by performing some processing on the video. However, many tasks remain that need to address and described below.

Future works:

- Add more data and solve the overfitting problem.
- Model Bert for multi-Language.
- After the server detect any hate speech or offensive language, send this to Annotator website.
- Deploy the application.

References

[1] “Hate Speech | Encyclopedia.com.” https://www.encyclopedia.com/international/encyclopedias-almanacs-transcripts-and-maps/hate-speech (accessed Jun. 23, 2022).

[2] “TikTok.” https://www.tiktok.com/community-guidelines?lang=en (accessed Jun. 23, 2022).

[3] “YouTube Community Guidelines enforcement – Hate Speech.” https://transparencyreport.google.com/youtube-policy/featured-policies/hate-speech?hl=en (accessed Jun. 23, 2022).

[4] C. O&rsquo;Regan, “Hate Speech Online: an (Intractable) Contemporary Challenge?,” *Current Legal Problems*, vol. 71, no. 1, pp. 403–429, Dec. 2018, doi: 10.1093/clp/cuy012.

[5] “Time to take action against hate speech | The Guardian Nigeria News - Nigeria and World News — Features — The Guardian Nigeria News – Nigeria and World News.” https://guardian.ng/features/media/time-to-take-action-against-hate-speech/ (accessed Jun. 23, 2022).

[6] “Bottle: Python Web Framework — Bottle 0.13-dev documentation.” https://bottlepy.org/docs/dev/ (accessed Jun. 23, 2022).

[7] O. de Gibert, N. Perez, A. García-Pablos, and M. Cuadros, “Hate Speech Dataset from a White Supremacy Forum,” Sep. 2018, [Online]. Available: http://arxiv.org/abs/1809.04444

[8] S. Malmasi and M. Zampieri, “Detecting Hate Speech in Social Media,” Dec. 2017, [Online]. Available: http://arxiv.org/abs/1712.06427

[9] W. Warner and J. Hirschberg, “Detecting Hate Speech on the World Wide Web,” 2012. [Online]. Available: http://info.yahoo.com/legal/us/yahoo/utos/utos-173.html