Marie Sklodowska Curie,

Research and Innovation Staff Exchange (RISE)

European Commission

Horizon 2020
European Union funding
for Research & Innovation

ENhancing seCurity and privAcy in the Social wEb: a user-centered approach for the protection of minors

ENCASE
ENhancing seCurity and privAcy in the Social wEb

WP4 – User Profiling for Detection and Prediction of Malicious Online Behavior

Deliverable D4.1 "Development of Automated Techniques to Detect Early Indications of Malicious Behavior of Social Network Users"

Editor(s):	Ilias Leontiadis (TID)
Author(s):	Ilias Leontiadis, Nicolas Kourtelis (TID), Emiliano De Cristofaro, Gullermo Tangil, Jeremiah Onalapo, Lucky Onwuzurike, Gianluca Stringhini, Enrico Mariconti, Tristan Caufield (UCL), Savvas Zannetou, Michael Sirivianos, Costas Tziouvas, Samaras Riginos (CUT), Rig Das, Gabriel Emile Hine (ROMA3), Antigoni Maria Founta, Dimitris Spathis, Athena Vakali, Nick Vassiliades, Despoina Chatzakou (AUTH), Rafael Constantinou (CYRIC)
Dissemination Level:	Public
Nature:	Report
Version:	0.6

ENCASE Project Profile

Contract Number	691025
Acronym	ENCASE
Title	ENhancing seCurity and privacy in the Social wEb: a user-centered approach for the protection of minors
Start Date	Jan 1st, 2016
Duration	48 Months

Partners

	Cyprus University of Technology	Cyprus
	Telefonica Investigacion Y Desarrollo SA	Spain
	University College London	United Kingdom
	Cyprus Research and Innovation Center, Ltd	Cyprus
	SignalGenerix Ltd	Cyprus
	Aristotle University	Greece
	Innovators, AE	Greece
	Universita Degli Studi, Roma Tre	Italy
	LSTech	Spain

Document History

AUTHORS

(CUT)	Michael Sirivianos, Savvas Zannettou, Costas Tziouvas, Riginos Samaras
(ROMA3)	Rig Das, Gabriel Emile Hine
(CYRIC)	Rafael Constantinou
(AUTH)	Antigoni Maria Founta, Dimitris Spathis, Athena Vakali, Nick Vassiliades, Despoina Chatzakou
(UCL)	Emiliano De Cristofaro, Gullermo Tangil, Jeremiah Onalapo, Lucky Onwuzurike, Gianluca Stringhini, Enrico Mariconti, Tristan Caufield
(TID)	Ilias Leontiadis, Nicolas Kourtelis

VERSIONS

Version	Date	Author	Remarks
0.1	10.09.2017	TID	Initial Table of Contents (TOC)
0.2	13.09.2017	TID	Complete TOC
0.3	05.12.2017	All authors	Contributions from partners received
0.4	15.12.2017	CUT, TID	Revision and restructure
0.5	26.12.2017	CUT	Final editing and restructure
0.6	28.12.2017	CUT, TID	Final version

Executive Summary

One of the main objectives of the ENCASE project is to develop a browser add-on and its corresponding machine learning algorithms for the detection of malicious and problematic behavior (such as cyberbullying, sexual predators, distressed or aggressive behavior, etc.).

During this project, we attempt to identify and quantify the various types of online abuse and we research a number of methodologies to detect hateful content, raid, abuse and bullying. Furthermore, to train our algorithms, we are also collecting a large-scale crowdsourced dataset.

This deliverable describes in detail the significant progress towards automatically detecting malicious behavior in the context of tasks T4.1: “User profiling to detect and prevent malicious and criminal activities” and T4.2: “Sentiment and affective analysis on individual and collective basis”.

The purpose of T4.1 is to collect and analyse social network data, user profiles and other relevant data aiming to understand how users are exposed to predators, cyberbullying and other types of malicious behavior. Using the extracted features, we build classifiers that will be used later by the corresponding browser add-on to detect multiple types of malicious behavior.

In T4.2 we perform sentiment and affective analysis for more in-depth understanding of user behavior. From the acquired knowledge, we devise feature extraction techniques and we build classifiers to early indications of malicious behavior of social network users.

.

Table of Contents

List of Figures

List of Tables

1. Introduction

Cyberbullying, cyberaggression, and the lack of cyber safety are serious and widespread issues affecting increasingly more Internet users. Arguably, in today's hyper-connected society, bullying, once limited in particular places or times of the day (e.g., school hours), can instead occur anytime, anywhere, with just a few taps on a keyboard.

In this document we attempt to address this problem by developing automated techniques with the aim to detect early indications of malicious behavior of social network users. More specifically:

- In Section 2, we attempt to understand and quantify to which extend hateful information is spread within 4chan and how organized campaigns are orchestrated that can disrupt other communities (e.g., YouTube)
- In Section 3, we use machine learning to detect abusive behaviors on Twitter
- In Section 4, we study whether a dictionary can be used to detect hate speech and how this can be generalized with the help of a novel deep-learning methodology
- In Section 5, we build on these two works and we design a novel deep-learning methodology that can combine multiple inputs to detect abusive behavior online
- In Section 6, we tackle the problem of early-detecting raids on YouTube videos with the help of an ensemble classifier
- In Section 7, we setup a honeypot scheme to answer basic questions about social abuse such as what are the characteristics of messages that attract negative reactions from other users and how can we stop them
- In Section 8, we introduce a methodology to identify the behavior trails of both the predator and the victim in an online harassment conversation
- In Section 9, we attempt to build a large-scale crowdsourced dataset that will help us to further improve our detection mechanisms

2. A Measurement of 4chan’s Politically Incorrect Forum and Its Effects

2.1. Project description and motivation

The discussion-board site 4chan has been part of the Internet’s dark underbelly since its inception, and recent political events have put it increasingly in the spotlight. In particular, /pol/, the “Politically Incorrect” board, has been a central figure in the outlandish 2016 US election season, as it has often been linked to the alt-right movement and its rhetoric of hate and racism. However, 4chan remains relatively unstudied by the scientific community: little is known about its user base, the content it generates, and how it affects other parts of the Web. In our work, we started addressing this gap by analyzing /pol/ along several axes, using a dataset of over 8M posts we collected over two and a half months. First, we performed a general characterization, showing that /pol/ users are well distributed around the world and that 4chan’s unique features encourage fresh discussions. We also analyzed content, finding, for instance, that YouTube links and hate speech are predominant on /pol/. Overall, our analysis not only provides the first measurement study of /pol/, but also insight into online harassment and hate speech trends in social media.

2.2. Methodology

2.2.1.1. Data Collection

On June 30, 2016, we started crawling 4chan using its JSON API.4 We retrieve /pol/’s thread catalogue every 5 minutes and compare the threads that are currently live to those in the previously obtained catalogue. For each thread that has been purged, we retrieve a full copy from 4chan’s archive, which allows us to obtain the full/final contents of a thread. For each post in a thread, the API returns, among other things, the post’s number, its author (e.g., “Anonymous”), timestamp, and contents of the post (escaped HTML). Although our crawler does not save images, the API also includes image metadata, e.g., the name the image is uploaded with, dimensions (width and height), file size, and an MD5 hash of the image. On August 6, 2016 we also started crawling /sp/, 4chan’s sports board, and on August 10, 2016 /int/, the international board. We note that for about 6% of the threads, the crawler gets a 404 error: from a manual inspection, it seems that this is due to “janitors” (i.e., volunteer moderators) removing threads for violating rules. The analysis presented in this work considers data crawled until September 12, 2016, except for the raids analysis presented later on, where we considered threads and YouTube comments up to Sept. 25. We also use a set of 60,040,275 tweets from Sept. 18 to Oct. 5, 2016 for a brief comparison of hate speech usage.

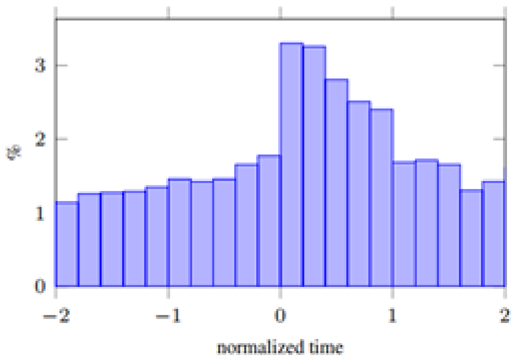


Figure 1. Distribution of the distance (in normalized thread lifetime) of the highest peak of activity in YouTube comments and the /pol/ thread they appear in. t = 0 denotes the time when video was first mentioned, and t = 1 the last related post in the thread

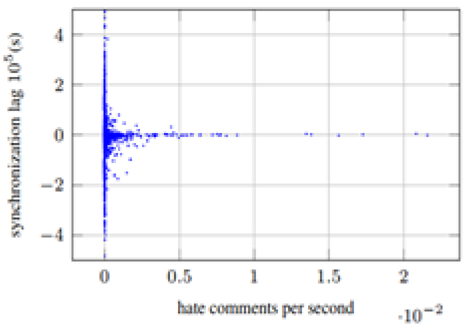


Figure 2. Hateful YouTube comments vs synchronization lag between /pol/ threads and corresponding YouTube comments. Each point is a /pol/ thread. The hateful comments count refers to just those within the thread lifetime ([0,+1])

2.2.1.2. Raid Analysis Methodology

One of our main goals was to highlight evidence of raids – i.e. attempt to disrupt another site, not from a network perspective (as in a DDoS attack), but from a content point of view. I.e., raids are not an attempt to directly attack a 3rd party service itself, but rather to disrupt the community that calls that service home. Raids on /pol/ are semi-organized: we anecdotally observe a number of calls for action consisting of a link to a target – e.g., a YouTube video or a Twitter hashtag – and the text “you know what to do,” prompting other 4chan users to start harassing the target.

We examined the comments from 19,568 YouTube videos linked to by 10,809 /pol/ threads to look for raiding behavior at scale. Therefore, rather than looking for a particular trigger on /pol/, we look for elevated activity in comments on YouTube videos linked from /pol/. In a nutshell, we expect raids to exhibit synchronized activity between comments in a /pol/ thread a YouTube link appears in and the number of comments it receives on YouTube. We also expect the rate of hateful comments to increase after a link is posted on /pol/.

To model synchronized activities, we use signal processing techniques. First, we introduce some notation: Let x be a /pol/ thread, and y the set of comments to a YouTube video linked from x. We denote with and, respectively, the set of timestamps of posts in x and y. Since the lifetime of /pol/ threads is quite dynamic, we shift and normalize the time axis for both and, so that t = 0 corresponds to when the video was first linked and t = 1 to the last post in the /pol/ thread: . In other words, we normalize to the duration of the /pol/ thread’s lifetime. We consider only /pol/ posts that occur after the YouTube mention, while, for computational complexity reasons, we consider only YouTube comments that occurred within the (normalized) [−10, +10] period, which accounts for 35% of YouTube comments in our dataset. From the list of YouTube comment timestamps, we compute the corresponding Probability Density Function (PDF) using the Kernel Density Estimator method, and estimate the position of the absolute maximum of the distribution. In Figure 1, we plot the distribution of the distance between the highest peak in YouTube commenting activity and the /pol/ post linking to the video. We observe that 14% of the YouTube videos experience a

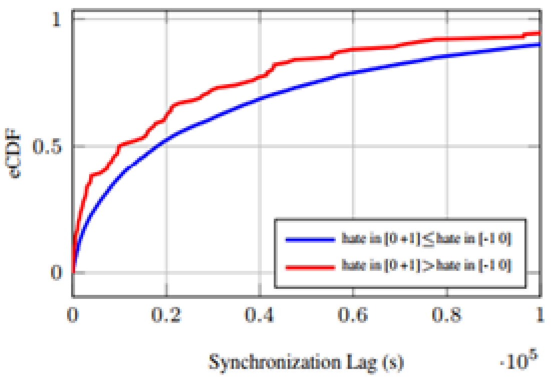


Figure 3. CDF of synchronization lag between /pol/ threads and YouTube comments, distinguishing between threads with YouTube videos containing higher hate comments percentage in the [0 +1] period or [-1 0]

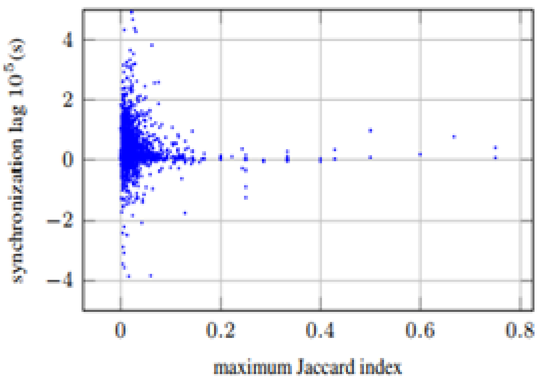


Figure 4. Maximum Jaccard Index of a YouTube video and all others vs synchronization lag between /pol/ threads and corresponding YouTube comments. Note the high correlation between overlap and synchronization lag

peak in activity during the period they are discussed on /pol/. In many cases, /pol/ seems to have a strong influence on YouTube activity, suggesting that the YouTube link posted on /pol/ might have a triggering behavior, even though this analysis does not necessarily provide evidence of a raid taking place. However, if a raid is taking place, then the comments on both /pol/ and YouTube are likely to be “synchronized.” Consider, for instance, the extreme case where some users that see the YouTube link on a /pol/ thread comment on both YouTube and and the /pol/ thread simultaneously: the two set of timestamps would be perfectly synchronized. In practice, we measure the synchronization, in terms of delay between activities, using cross-correlation to estimate the lag between two signals. In practice, cross-correlation slides one signal with respect to the other and calculates the dot product (i.e., the matching) between the two signals for each possible lag. The estimated lag is the one that maximizes the matching between the signals. We represent the sequences as signals (x(t) and y(t)), using Dirac delta distributions δ(·). Specifically, we expand x(t) and y(t) into trains of Dirac delta distributions:

$$x(t) = \sum_{i=1}^{N_x} \delta\left(t - t_x^i\right); \qquad y(t) = \sum_{j=1}^{N_y} \delta\left(t - t_y^j\right)$$

and we calculate c(t), the continuous time cross-correlation between the two series as:

$$c(t) = \int_{-\infty}^{+\infty} x(t+\tau) y(\tau) d\tau = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \delta\left(t - \left(t_y^j - t_x^i\right)\right)$$

The resulting cross-correlation is also a Dirac delta train, representing the set of all possible inter-arrival times between elements from the two sets. If y(t) is the version of x(t) shifted by ΔT (or at least contains a shifted version of x(t)), with each sample delayed with a slightly different time lag, c(t) will be characterized by a high concentration of pulses around ΔT. As in the peak activity detection, we can estimate the more likely lag by computing the associated PDF function by means of the Kernel Density Estimator method, and then compute the global maximum:

$$\tilde{c}(t) = \int_{-\infty}^{+\infty} c(t+\tau) k(\tau) d\tau; \quad \Delta T = \underset{t}{\arg max}\, \tilde{c}(t)$$

where k(t) is the kernel smoothing function (typically a zero mean Gaussian function).

2.3. Results

Building on the above insights, we provide large-scale evidence of raids. If a raid is taking place, we expect the estimated lag ΔT to be close to zero, and we can validate this by looking at the content of the YouTube comments. Figure 2 plots the relationship between the number of hateful comments on YouTube that occur within the /pol/ thread lifetime (i.e., containing at least one word from the hatebase dictionary) and the synchronization lag between the /pol/ thread and the YouTube comments. The trend is quite clear: as the rate of hateful comments on YouTube increases, the synchronization lag between /pol/ and YouTube comments decreases. This shows that almost all YouTube videos affected by (detected) hateful comments during the /pol/ thread lifetime are likely related to raids. Figure 3 plots the CDF of the absolute value of the synchronization lag between /pol/ threads and comments on the corresponding YouTube videos. We distinguish between comments with a higher percentage of comments containing hate words during the life of the thread from those with more before the thread. In other words, we compare threads where /pol/ appears to have a negative impact vs. those where they do not. From the plot, we observe that the YouTube comments with more hate speech during the /pol/ thread’s lifetime are significantly ($p < 0.01$ with a 2-sample Kolmogorov-Smirnov test) more synchronized with the /pol/ thread itself. Finally, to further show that /pol/ is raiding YouTube videos, we can look at the authors of YouTube comments. We argue that, unlike the anonymous venue of /pol/, raids on a service like YouTube will leave evidence via account usage, and that the same raiding YouTube accounts will likely be used by /pol/ users more than once. Indeed, while it is moderately easy to create a new YouTube account, there is still some effort involved. Troll accounts might also be cultivated for use over time, gaining some reputation as they go along. Perhaps more importantly, while less anonymous than /pol/, YouTube accounts are still only identified by a profile name and do not truly reveal the identity of the user. To measure this, we compute the overlap (Jaccard index) of commenters in each YouTube video. In Figure 4 we plot the synchronization lag as a function of the maximum overlap between a given video and all others. From the figure, we observe that if a YouTube video has relatively high overlap with at least one other YouTube video, it also highly synchronized with its corresponding /pol/ thread, indicative of a raid taking place

2.4. Conclusion

We have done first large-scale study of /pol/, 4chan's politically incorrect board, arguably the most controversial one owing to its links to the alt-right movement and its unconventional support to Donald Trump's 2016 presidential campaign. We studied "raiding" behavior by looking for evidence of /pol/'s hateful impact on YouTube comments. We used signal processing techniques to discover that peaks of commenting activity on YouTube tend to occur within the lifetime of the thread they were posted to on /pol/. Next, we used cross-correlation to estimate the synchronization lag between /pol/ threads and comments on linked YouTube videos. Here, we found that as the synchronization lag approaches zero, there is an increase in the rate of comments with hate words on the linked YouTube comments. Finally, we saw that if two YouTube videos' comments had many common authors they were likely to be highly synchronized, indicating potential raider accounts. This evidence suggests that, while not necessarily explicitly called for (and in fact, against /pol/'s rules), /pol/ users are performing raids in an attempt to disrupt the community of YouTube users. Overall, our analysis provides not only the first measurement study of /pol/, but also insight into the continued growth of hate and extremism trends on social media, and prompts a few interesting problems for future research. Naturally, however, our work is not without limitations. First, although the Hatebase dataset we used, is an invaluable resource for hate speech analysis, the usage of "hate" words may be context-dependent, and we leave it to future work to investigate how to distinguish context (e.g., by recognizing sarcasm or trolling).

3. Detecting Abusive Behaviors with Machine Learning

3.1. Project Description and Motivation

Cyberbullying and cyberaggression are serious and widespread issues affecting increasingly more Internet users. Arguably, in today's hyper-connected society, bullying, once limited in particular places or times of the day (e.g., school hours), can instead occur anytime, anywhere, with just a few taps on a keyboard. Cyberbullying and cyberaggression can take many forms and definitions [1, 2], however, the former typically denotes repeated and hostile behavior performed by a group or an individual and the latter intentional harm delivered via electronic means to a person or a group of people who perceive such acts as offensive, derogatory, harmful, or unwanted [1].

Only a few years ago, when Internet adoption was still limited, cyberbullying was not taken seriously: the typical advice was to ''just turn off the screen'' or ''disconnect'' [3]. However, as its proliferation and the extent of its consequences reach epidemic levels [4], this behavior can no longer be ignored: in 2014, about 50% of young social media users reported being bullied online in various forms [5]. Even more interesting and worrying (based on the aforementioned report) is the fact that 15% of high school students of 9-12 grades and 9% of students in grades 6-12 have experienced cyberbullying. So, the need to develop methods and tools for the early detection and prevention of such abusive cases is more than obligatory.

Over the past few years, several techniques have been proposed to measure and detect offensive or abusive content / behavior on platforms like Instagram [8], YouTube [9], Yahoo Finance [10], and Yahoo Answers [12]. Chen et al. [9] use both textual and structural features (e.g., ratio of imperative sentences, adjective and adverbs as offensive words) to predict a user's aptitude in producing offensive content in YouTube comments, while Djuric et al. [10] rely on word embeddings to distinguish abusive comments on Yahoo Finance. Nobara et al. [11] perform hate speech detection on Yahoo Finance and News data, using supervised learning classification. Kayes et al. [12] find that users tend to flag abusive content posted on Yahoo Answers in an overwhelmingly correct way (as confirmed by human annotators). Also, some users significantly deviate from community norms, posting a large amount of content that is flagged as abusive. Through careful feature extraction, they also show it is possible to use machine learning methods to predict which users will be suspended. Dinakar et al. [13] detect cyberbullying by decomposing it into detection of sensitive topics. They collect YouTube comments from controversial videos, use manual annotation to characterize them, and perform a bag-of-words driven text classification. Hee et al. [14] study linguistic characteristics in cyberbullying-related content extracted from Ask.fm, aiming to detect fine-grained types of cyberbullying, such as threats and insults. Besides the victim and harasser, they also identify bystander-defenders and bystander-assistants, who support, respectively, the victim or the harasser. Hosseinmardi et al. [8] study images posted on Instagram and their associated comments to detect and distinguish between cyberaggression and cyberbullying. Finally, authors in [15] present an approach for detecting bullying words in tweets, as well as demographics about bullies such as their age and gender.

In this work, the focus is on understanding and detecting abusive phenomena that take place on Twitter. Twitter is one of the most popular online social media sources, but despite the seriousness of the problem, there are very few successful efforts to detect abusive behavior, both from the research community and Twitter itself [6], due to several inherent obstacles. First, tweets are short and full of grammar and syntactic flaws, which makes it harder to use natural language processing tools to extract text-based attributes and characterize user interactions. Second, each tweet provides fairly limited context, therefore, taken on its own, an aggressive tweet may be disregarded as normal text, whereas, read along with other tweets, either from the same user or in the context of aggressive behavior from multiple users, the same tweet could be characterized as bullying. Third, despite extensive work on spam detection in social media, Twitter is still full of spam accounts [7] often using vulgar language and exhibiting behavior (repeated posts with similar content, mentions, or hashtags) that could also be considered as aggressive or bullying. Filtering out such accounts from actual abusive users may be a difficult task. Finally, aggression and bullying against an individual can be performed in several ways beyond just obviously abusive language -- e.g., via constant sarcasm, trolling, etc.

This work advances the state-of-art on cyberbullying and cyberaggression detection by proposing a scalable methodology for large-scale analysis and extraction of text, user, and network based features on Twitter, which has not been studied in this context before. The novel methodology analyzes users' tweets, individually and in groups, and extracts appropriate features connecting user behavior with a tendency of aggression or bullying. The importance of such attributes is thoroughly examined, and further advance the state-of-art by focusing on new network-related attributes that further distinguish Twitter-specific user behaviors.

3.2. Methodology

This section provides the methods followed to understand initially the behavioral patterns of abusive users and create then a method suitable for detecting specific instances of abusive behaviors, i.e., cyberbullying and cyberaggression.

3.2.1.1. Characteristics of Abusive Users

In order to study and understand the characteristics that are more indicative of abusive users' behavior a measurement study was conducted on Twitter by building upon the Gamergate controversy and studying the properties of the users involved in such a controversy, the content they post, and how they differ from random/typical Twitter users. The Gamergate controversy [16] is one example of a coordinated campaign of harassment in the online world. It started with a blog post by an ex-boyfriend of independent game developer Zoe Quinn, alleging sexual improprieties. It quickly devolved into a polarizing issue, involving sexism, feminism, and ''social justice,'' taking place on social media like Twitter. Although held up as a pseudo-political movement by its adherents, there is substantial evidence that Gamergate is more accurately described as an organized campaign of hate and harassment [17]. What started as ''mere'' denigration of women in the gaming industry, eventually evolved into directed threats of violence, rape, and murder [17]. With the advent of ubiquitous social media and a community born in the digital world, the Gamergate controversy provides us a unique point of view into online harassment campaigns.

Data collection. In this work, we explored a slice of the Gamergate controversy by analyzing 650k tweets of users who engaged in it. As a first attempt at quantifying this controversy, the focus was on how these users, and the content they post, differ from random (typical) Twitter users. To collect the necessary, we built upon the Twitter Streaming API [18] which gives access to 1% of all tweets. This returns a set of correlated information, either user-based, e.g., poster's username, followers and friends count, profile image, total number of posted/liked/favorite tweets, or text-based, e.g., the text itself, hashtags, URLs, if it is a retweeted or reply tweet, etc. The data collection process took place from June to August 2016. Initially, we obtained a 1% of the sample public tweets and parsed it to select all tweets containing the seed word #GamerGate, which are likely to involve the type of behavior, and the case study we were interested in. The #GamerGate served as a seed for a snowball sampling of other hashtags likely associated with cyberbullying and aggressive behavior. So, then we included tweets with hashtags appearing in the same tweets as #GamerGate. Overall, we reached 308 hashtags during the data collection period. A manual examination of these hashtags reveals that they do contain many hate words, such as #InternationalOffendAFeministDay, #IStandWithHateSpeech, and #KillAllNiggers. To complement the dataset with cases that are less likely to contain abusive content, we also crawled an 1M random sample of texts over the same time period which constituted our baseline.

Data processing. We performed preprocessing of the data collected to produce a ''clean'' dataset, free of noisy data, by both cleaning data and removing spam content. During the cleaning process we removed stop words, URLs, numbers, and punctuation marks. To eliminate spam content, we proceeded with a first-level spam removal process by considering two attributes which have already been used as filters [19] to remove spam incidents: (i) the number of hashtags per tweet (often used for boosting the visibility of the posted tweets), and (ii) posting of (almost) similar tweets. To find optimal cutoffs for these heuristics, we studied both the distribution of hashtags and the duplication of tweets.

Having collected and preprocessed the data, a measurement study took place (see Section 3.3.1) to compare the baseline and the Gamergate related datasets across various dimensions, including user attributes, posting activity, content semantics, and Twitter account status.

Abusive Detection Method

Our approach to detect aggressive and bullying behavior on Twitter, as summarized in Figure 5, involves the following steps: (1) data collection, (2) preprocessing of tweets, (3) sessionization, (4) ground truth building, (5) extracting user-, text-, and network-level features, (6) user modeling and characterization, and (7) classification.

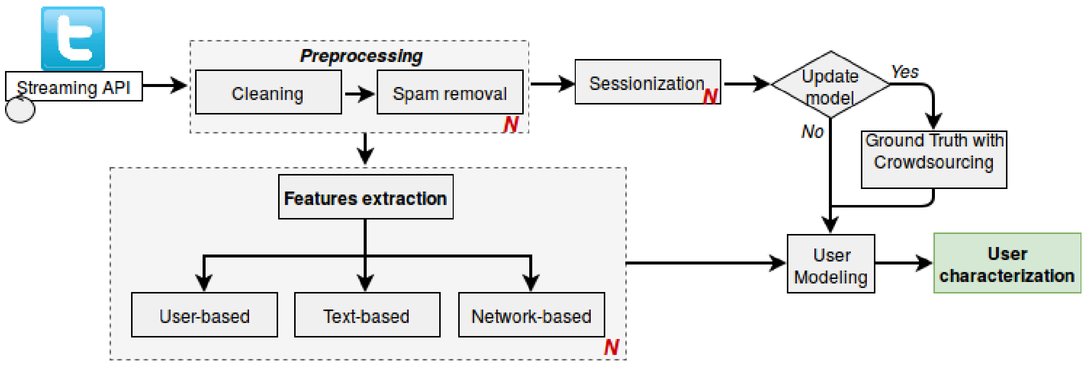


Figure 5. Overview of the methodology. N denotes the ability to parallelize a task on N processors.

Data Collection. For the data collection similar to Section 3.2.1 we relied again on Twitter's Streaming API.

Preprocessing. Next, we removed stop words, URLs, and punctuation marks from the tweet text and perform normalization -- i.e., we eliminated repetitive characters; e.g., the word ‘’yessss’’ is converted to ‘’yes.’’ This step also involved the removal of spam content, similar to the approach presented earlier.

Sessionization. Since analyzing single tweets does not provide enough context to discern if a user is behaving in an aggressive or bullying way, we grouped tweets from the same user, based on time clusters, into sessions and analyzed them instead of single tweets. More specifically, inspired by Hosseinmardi et al. [8] -- who consider a lower threshold of comments for media sessions extracted from Instagram to be presented in the annotation process -- we created, for each user, sets of time-sorted tweets (sessions) by grouping tweets posted close to each other in time. First, we removed users who were not significantly active, i.e., tweeting less than five times in the 3-month period. Then, we used a session-based model where, for each session S_i, the interarrival time between tweets did not exceed a predefined time threshold t_l. We experimented with various values of t_l to find an optimal session duration and arrived at a threshold of 8 hours. The minimum, median, and maximum length of the resulting sessions (in terms of the number of their included tweets) for the hate-related (i.e., Gamergate) dataset were, respectively, 12, 22, and 2.6k tweets. For the baseline set of tweets, they were 5, 44, and 1.6k tweets.

Next, we divided sessions in batches, as otherwise they would contain too much information to be carefully examined by a crowdworker within a reasonable period of time. To find the optimal size of a batch, i.e., the number of tweets per batch, we performed preliminary labeling runs on CrowdFlower, involving 100 workers each, using batches of exactly 5, 5-10, and 5-20 tweets. The intuition was that increasing the batch size provides more context to the worker to assess if a poster is acting in an aggressive or bullying behavior, however, too many tweets might confuse them. The best results with respect to labeling agreement -- i.e., the number of workers that provide the same label for a batch -- occurred with 5-10 tweets per batch. Therefore, we eliminated sessions with fewer than 5 tweets, and further split those with more than 10 tweets (preserving the chronological ordering of their posted time). In the end, we arrived at 1,500 batches. We maintained the same number of batches for both the hate-related and baseline tweets.

Ground Truth. We built ground truth (needed for machine learning classification, explained next) using human annotators. For this we used a crowdsourced approach, by recruiting workers who are provided with a set of tweets from a user, and were asked to classify them according to predefined labels. More specifically, the goal was to label each Twitter user -- not single tweets -- as normal, aggressive, bullying, or spammer by analyzing their batch(es) of tweets. Note that we also allowed for the possibility that a user is spamming and has passed our basic spam filtering. Workers were provided with the following definitions of aggressive, bullying, and spam behaviors:

aggressor: someone who posts at least one tweet or retweet with negative meaning, with the intent to harm or insult other users (e.g., the original poster of a tweet, a group of users, etc.);

bully: someone who posts multiple tweets or retweets (more than 2) with negative meaning for the same topic and in a repeated fashion, with the intent to harm or insult other users (e.g., the original poster of a tweet, a minor, a group of users, etc.) who may not be able to easily defend themselves during the postings;

spammer: someone who posts texts of advertising / marketing or other suspicious nature, such as to sell products of adult nature, and phishing attempts.

We redirected employed crowd workers to an online survey tool we developed and we asked them to label 10 batches.. We also provided them with the user profile description (if any) of the Twitter user they were labeling and the definition of aggressive, bullying, and spammer behaviors. Overall, we recruited 834 workers. They were allowed to participate only once to eliminate behavioral bias across tasks and discourage rushed tasks. Each batch was labeled by 5 different workers, and, similar to [8, 11], a majority vote was used to decide the final label. We received 1,303 annotated batches, comprising 9,484 tweets in total. 4.5% of users were labeled as bullies, 3.4% as aggressors, 31.8% as spammers, and 60.3% as normal. Overall, abusive users (i.e., bullies and aggressors) make up about 8% of our dataset, which mirrors observations from previous studies (e.g., in [12] 9% of the users in the examined dataset exhibits bad behavior, while in [20] 7% of users cheated). Thus, the ground truth dataset contained a representative sample of aggressive/abusive content. The inter-rater agreement was 0.54.

Feature Extraction. We extracted features from both tweets and user profiles. Features were user-, text-, or network-based; e.g., the number of followers, tweets, hashtags, etc. Table 1 summarizes the features considered for each category.

Type	Feature
User	avg. # posts, # days since account creation, verified account, # subscribed lists, posts' interarrival time, default profile image?, statistics on sessions: total number, avg., median, and STD. of their size
Text	avg. # hashtags, avg. # emoticons, avg. # upper cases, # URLs, avg. sentiment score, avg. emotional scores, hate score, avg. word embedding score, avg. curse score
Network	# friends, # followers, hubs, (d=#followers/#friends), authority, avg. power diff. with mentioned users, clustering coefficient, reciprocity, eigenvector centrality, closeness centrality, louvain modularity

Table 1. Features considered in the study

Classification. The final step was the classification process using the (extracted) features and the ground truth. Different machine learning techniques were considered in this task, including probabilistic classifiers (e.g., Naive Bayes), decision trees (e.g., J48), and ensembles (e.g., Random Forests). The best performance with respect to training time and performance, obtained with the Random Forest classifier.

Features selection. Most of the features presented in Table 1 were useful (stat. significant -- was estimated based on the two-sample Kolmogorov-Smirnov test, a nonparametric statistical test. As statistically significant were considered all cases with $p < 0.05$) in discriminating between the user classes. However, some were not useful and were excluded from the modeling analysis to avoid adding noise. Specifically, we excluded the following features from the analysis: user-based - verified account, default profile image, statistics on sessions, text-based - average emotional scores, hate score, average word embedding, average curse score, and network-based - closeness centrality and louvain modularity.

Evaluation. For evaluation purposes, we considered standard machine learning performance metrics: precision (prec), recall (rec), and weighted area under the ROC curve (AUC), at the class level and overall average across classes. Also, the overall kappa (compares an observed accuracy with an expected accuracy, i.e., random chance), the root mean squared error (RMSE), which measures the differences among the values predicted by the model and the actually observed values, and finally the accuracy values were estimated.

Experimentation phases. Two setups were tested to assess the feasibility of detecting user behavior: (i) 4-classes classification: bully, aggressive, spam, and normal users, (ii) 3-classes classification: bully, aggressive, and normal users. The second setup examines the case where we filter out spam with a more elaborate technique and attempt to detect the bullies and aggressors from normal users.

3.3. Results

This section presents the results of both the measurement study and the method developed for detecting specific instances of abusive behavior, namely cyberbullying and cyberaggression.

3.3.1.1. Results on the Measurement Study

Here, we present the results of the measurement-based characterization, comparing the baseline and the Gamergate related datasets across various dimensions, including user attributes, posting activity, content semantics, and Twitter account status.

How Active are Gamergaters?

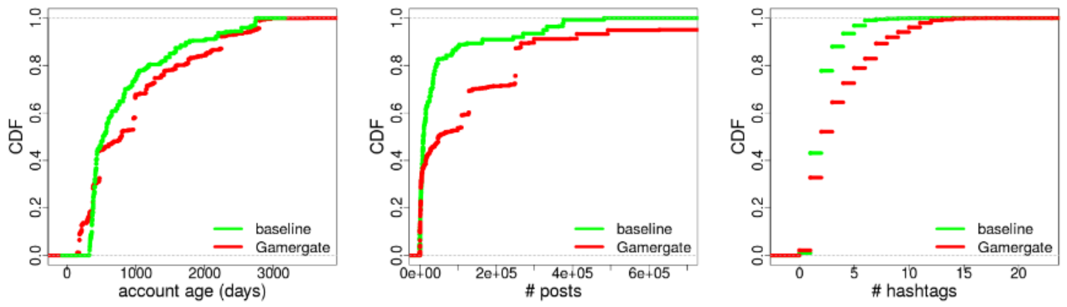


Figure 6. CDF of (a) Account age, (b) Number of posts, and (c) Hashtags.

Account age. An underlying question about the Gamergate controversy is what started first: participants tweeting about it or Twitter users participating in Gamergate? In other words, did Gamergate draw people to Twitter, or were Twitter users drawn to Gamergate? In Figure 6a, we plot the distribution of account age for users in the Gamergate dataset and baseline users. For the most part, Gamergate users tend to have older accounts (mean = 982.94 days, median = 788 days, STD = 772.49 days). The mean, median, and STD values for the random users are 834.39, 522, and 652.42 days, respectively. Based on a two-sample Kolmogorov-Smirnov test the two distributions are different with a test statistic D = 0.20142 and p < 0.01. Overall, the oldest account in our dataset belongs to a Gamergate user, while only 26.64% of baseline accounts are older than the mean value of the Gamergate users. Figure 6 indicates that Gamergaters were existing Twitter users drawn to the controversy. In fact, their familiarity with Twitter could be the reason that Gamergate exploded in the first place.

Tweets and Hashtags. In Figure 6b, we plot the distribution of the number of tweets made by Gamergate users and random users. Gamergaters are significantly more active than random Twitter users (D = 0.352, p < 0.01). The mean, median, and STD values for the Gamergate (random) users are 135,618 (49,342), 48,587 (9,429), and 185,997 (97,457) posts, respectively. Figure 6c reports the CDF of the number of hashtags found in users' tweets for both Gamergate and the random sample, finding that Gamergaters use significantly (D = 0.25681, p < 0.01) more hashtags than random Twitter users.

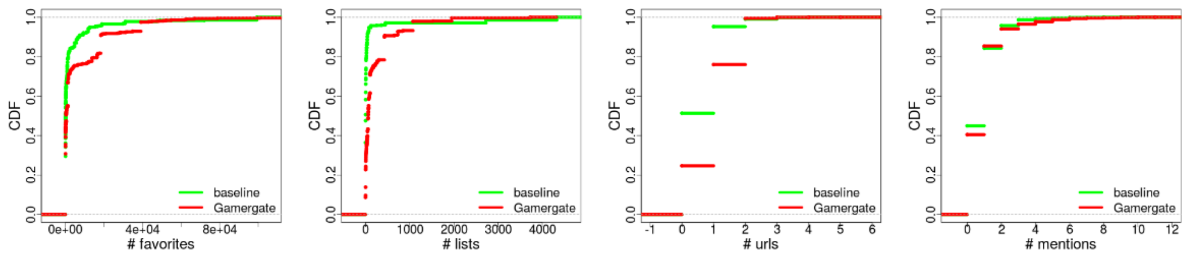


Figure 7. CDF of (a) Number of Favorites, (b) Lists, (c) URLs, (d) Mentions.

Other characteristics. Figures 7a and 7b show the CDFs of favorites and lists declared in the users' profiles. We note that in the median case, Gamergate users are similar to baseline users, but on the tail end (30% of users), Gamergate users have more favorites and topical lists declared than random users. Then, Figure 7c reports the CDF of the number of URLs found in tweets by both baseline and Gamergate users. The former post fewer URLs (the median indicates a difference of 1-2 URLs, D = 0.26659, p < 0.01), while the latter post more in an attempt to disseminate information about their ‘’cause,’’ somewhat using Twitter like a news service. Finally, Figure 7d shows that Gamergate users tend to make more mentions within their posts, which can be ascribed to the higher number of direct attacks compared to random users.

Overall, the behavior we observe is indicative of Gamergate users' ‘’mastery’’ of Twitter as a mechanism for broadcasting their ideals. They make use of more advanced features, e.g., lists, tend to favorite more tweets, and share more URLs and hashtags than random users. Using hashtags and mentions can draw attention to their message, thus Gamergate users likely use them to disseminate their ideas deeper in the Twitter network, possibly aiming to attack more users and topical groups.

How Social are Gamergaters?

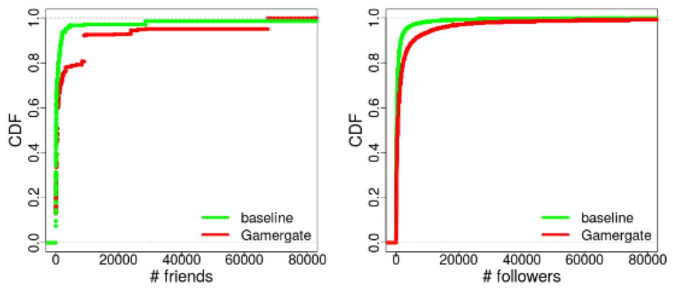


Figure 8. CDF of (a) Number of Friends, (b) Followers.

Gamergaters are involved in what we would typically think of as anti-social behavior. However, this is somewhat at odds with the fact that their activity takes place primarily on social media. Aiming to give an idea of how ‘’social’’ Gamergaters are, in Figures 8a and 8b, we plot the distribution of friends and followers for Gamergate vs baseline users. We observe that, perhaps surprisingly, Gamergate users tend to have more friends and followers (D=0.34 and 0.39, p < 0.01 for both). Although this might be somewhat counter-intuitive, the reality is that Gamergate was born on social media, and the controversy appears to be a clear ‘’us vs. them’’ situation. This leads to easy identification of in-group membership, thus heightening the likelihood of relationship formation.

The ease of in-group membership identification is somewhat different than polarizing issues in the real world where it may be difficult to know a person's views on a polarizing subject, without actually engaging them on the subject. In fact, people in real life might be unwilling to express their viewpoint because of social consequences. On the contrary, on social media platforms like Twitter, (pseudo-)anonymity often removes much of the inhibition people feel in the real world, and public timelines can often provide persistent and explicit expression of viewpoints.

How Different is Gamergater's Content?

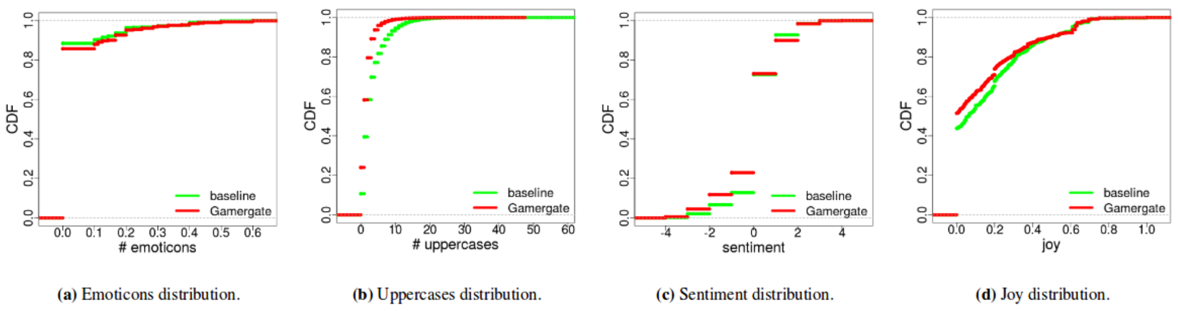


Figure 9. CDF of (a) Emoticons, (b) Uppercases, (c) Sentiment, (d) Joy.

Emoticons and Uppercase Tweets. Two common ways to express emotion in social media are emoticons and ‘’shouting’’ by using all capital letters. Based on the nature of Gamergate, we would expect a relatively low number of emoticon usage, but many tweets that would be shouting in all uppercase letters. However, as we can see in Figures 9a and 9b, which plot the CDF of emoticon usage and all uppercase tweets, respectively, this is not the case. Gamergate and random users tend to use emoticons at about the same rate (we are unable to reject the null hypothesis with D=0.028 and p = 0.96). However, Gamergate users tend to use all uppercase less often (D=0.212, $p < 0.01$). As mentioned, Gamergate users are savvy Twitter users, and generally speaking, shouting tends to be ignored. Thus, one explanation for this behavior is that Gamergate users avoid such a simple ‘’tell’’ as posting in all uppercase, to ensure their message is not so easily dismissed.

Sentiment. In Figure 9c, we plot the CDF of sentiment of tweets. In both cases (Gamergate and baseline) around 25% of tweets are positive. However, Gamergate users post tweets with a generally more negative sentiment (D=0.101, $p < 0.01$). In particular, around 25% of Gamergaters' tweets are negative compared to only around 15% for baseline users. This observation aligns with the fact that the Gamergate dataset contains a large proportion of offensive posts. Based on [22], we also extracted sentiment values for 6 different emotions: anger, disgust, fear, joy, sadness, and surprise. We note that of these, the 2-sample KS test is unable to reject the null hypothesis except for joy, as shown in Figure 5d (D=0.089, $p < 0.01$). This is particularly interesting because it contradicts the narrative that Gamergaters are posting virulent content out of anger. Instead, Gamergate users are less joyful, and this is a subtle but important difference: they are not necessarily angry, but they are apparently not happy.

3.3.1.2. Results based on the Abusive Detection Method

Here, we present the results obtained with the model created for detecting bullying and aggressive behaviors, using the features extracted and the labels provided by the crowdworkers as presented earlier (Section 3.2.2).

	Prec.	Rec.	AUC
bully	0.411	0.432	0.893
aggressive	0.295	0.118	0.793
spammer	0.686	0.561	0.808
normal	0.782	0.883	0.831
overall (avg.)	0.718	0.733	0.815

Table 2. 4-classes classification

Detecting offensive classes. Here, we examine whether it is possible to distinguish between bully, aggressive, spam, and normal users. Table 2 overviews the results obtained with the Random Forest classifier. In more detail, we observe that the classifier succeeds to detect 43.2% of the bully cases, which is fair enough considering the small number of bully cases identified to begin with (only 4.5% of the dataset). In the aggressive case, we observe that recall is quite low, 11.8%. The misclassified cases mostly fall in either the normal or bullying classes, which aligns with the human annotations gathered during the crowdsourcing phase. Overall, the average precision is 71.6%, and the recall is 73.32%, while the accuracy is 73.45%, with 0.4717 kappa and 0.3086 RMSE.

	Prec.	Rec.	AUC
bully	0.555	0.609	0.912
aggressive	0.304	0.114	0.812
normal	0.951	0.976	0.911
overall (avg.)	0.899	0.917	0.907

Table 3. 3-classes classification

Classifying after spam removal. In this experimental phase, we explore whether the distinction between bully/aggressive and normal users will be more evident after applying a more sophisticated spam removal process in the preprocessing step. To this end, we removed from our dataset all the cases identified by the annotators as spam, and re-run the Random Forest modeling and classification. Table 3 shows that, as expected, for bully cases there is an important increase in both the precision (14.4%) and recall (17.7%). For aggressors, the precision and recall values are almost the same, indicating that further examination of this behavior is warranted in the future. Overall, the average precision and recall of the Random Forest model is 89.9% and 91.7%, respectively, while the accuracy is 91.08% with 0.5284 kappa value and 0.2117 RMSE. Considering a 0.907 AUC, we believe that with a more sophisticated spam detection applied on the stream of tweets, the considered features and classification techniques can perform even better at detecting bullies and aggressors and distinguishing them from the typical Twitter users.

Experiment	Feature (preserving order)
4-classes	#friends (11.43%), reciprocity (11.10%), #followers (10.84%), #followers / #friends (9.82%), interarrival (9.35%), #lists (9.11%), hubs (9.07%), #URLs (7.69%), #hashtags (7.02%), authority (6.33%), account age (4.77%), clustering coefficient (3.47%)
3-classes	#followers / #friends (12.58%), #friends (12.56%), #followers (11.90%), interarrival (11.67%), reciprocity (11.01%), #lists (9.57%), hubs (8.41%), #hashtags (6.2%), #posts (6.01%), account age (4.13%), #URLs (3.73%), power difference (2.23%)

Table 4. Features evaluation

Features evaluation. Table 4 shows the top 12 features for each setup based on information gain. Overall, in both setups the most contributing features tend to be user- and network-based, which describe the activity and connectivity of a user in the network.

3.4. Conclusion

Although the digital revolution and the rise of social media enabled great advances in communication platforms and social interactions, wider proliferation of harmful behavior has also emerged. Unfortunately, effective tools for detecting harmful actions are scarce, as this type of behavior is often ambiguous in nature and/or exhibited via seemingly superficial comments and criticisms. Aiming to address this gap, initially we analyzed the behavioral patterns exhibited by abusive users and their differences from typical Twitter user categories. Specifically, we focused on a Twitter dataset revolving around the Gamergate controversy which led to many incidents of cyberbullying and cyberaggression on various gaming and social media platforms. We studied the properties of users tweeting about Gamergate, the content they post, and the differences in their behavior compared to typical Twitter users. Additionally, we developed a novel system geared towards automatically classifying two specific types of harmful online behavior, i.e., cyberbullying and cyberaggression. We relied on crowd-workers to label 1.5k users as normal, spammers, aggressive, or bullies, from a corpus of almost 10k tweets (distilled from a larger set of 1.6M tweets), using an efficient, streamlined labeling process. We investigated 30 features from 3 types of attributes (user, text, network based) characterizing such behavior. While prior work almost exclusively focused on user- and text-based features (e.g., linguistics, sentiment, membership duration), we performed a thorough analysis of network-based features, and found them to be very useful, as they actually are the most effective for classifying aggressive user behavior (half of the top-12 features in discriminatory power are network-based). Text-based features, somewhat surprisingly, do not contribute as much to the detection of aggression (with an exception of tweet characteristics, such as number of URLs, hashtags, and sentiment).

3.5. Section References

[1] Grigg, D. W. (2010). Cyber-aggression: Definition and concept of cyberbullying. *Journal of Psychologists and Counsellors in Schools*, *20*(2), 143-156.

[2] Tokunaga, R. S. (2010). Following you home from school: A critical review and synthesis of research on cyberbullying victimization. *Computers in human behavior*, *26*(3), 277-287.

[3] Meagan Miller, 4 Oct 2016, goo.gl/n1W6nt

[4] Cyberbullying Research Center, 26 Nov 2016, goo.gl/7kzSY0

[5] Facts About Bullying, 2014, https://www.stopbullying.gov/media/facts/index.html

[6] The Guardian, Twitter CEO: We suck at dealing with trolls and abuse, 2015, goo.gl/6CxnwP

[7] Chen, C., Zhang, J., Chen, X., Xiang, Y., & Zhou, W. (2015, June). 6 million spam tweets: A large ground truth for timely Twitter spam detection. In *Communications (ICC), 2015 IEEE International Conference on* (pp. 7065-7070). IEEE.

[8] Hosseinmardi, H., Mattson, S. A., Rafiq, R. I., Han, R., Lv, Q., & Mishra, S. (2015, December). Analyzing labeled cyberbullying incidents on the instagram social network. In *International Conference on Social Informatics* (pp. 49-66). Springer, Cham.

[9] Chen, Y., Zhou, Y., Zhu, S., & Xu, H. (2012, September). Detecting offensive language in social media to protect adolescent online safety. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)* (pp. 71-80). IEEE.

[10] Djuric, N., Zhou, J., Morris, R., Grbovic, M., Radosavljevic, V., & Bhamidipati, N. (2015, May). Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web* (pp. 29-30). ACM.

[11] Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., & Chang, Y. (2016, April). Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web* (pp. 145-153). International World Wide Web Conferences Steering Committee.

[12] Kayes, I., Kourtellis, N., Quercia, D., Iamnitchi, A., & Bonchi, F. (2015, May). The social world of content abusers in community question answering. In *Proceedings of the 24th International Conference on World Wide Web* (pp. 570-580). International World Wide Web Conferences Steering Committee.

[13] Dinakar, K., Reichart, R., & Lieberman, H. (2011). Modeling the detection of Textual Cyberbullying. *The Social Mobile Web*, *11*(02).

[14] Van Hee, C., Lefever, E., Verhoeven, B., Mennes, J., Desmet, B., De Pauw, G., ... & Hoste, V. (2015). Automatic detection and prevention of cyberbullying. In *International Conference on Human and Social Analytics (HUSO 2015)* (pp. 13-18). IARIA.

[15] Saravanaraj, A., Sheeba, J. I., & Devaneyan, S. P. (2016). Automatic Detection of Cyberbullying from Twitter.

[16] Massanari, A. (2017). # Gamergate and The Fappening: How Reddit’s algorithm, governance, and culture support toxic technocultures. *New Media & Society*, *19*(3), 329-346.

[17] Wingfield, N. (2014). Feminist critics of video games facing threats in ‘GamerGate’campaign. *The*

New York Times, *15*.
[18] Twitter Streaming API, https://dev.twitter.com/streaming/overview
[19] Wang, A. H. (2010, July). Don't follow me: Spam detection in twitter. In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on* (pp. 1-10). IEEE.
[20] Blackburn, J., Simha, R., Kourtellis, N., Zuo, X., Ripeanu, M., Skvoretz, J., & Iamnitchi, A. (2012, April). Branded with a scarlet C: cheaters in a gaming social network. In *Proceedings of the 21st international conference on World Wide Web* (pp. 81-90). ACM.
[21] Davis, J., & Goadrich, M. (2006, June). The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning* (pp. 233-240). ACM.
[22] Chatzakou, D., Vakali, A., & Kafetsios, K. (2017). Detecting variation of emotions in online activities. *Expert Systems with Applications*, *89*, 318-332.

4. Hate Speech Detection with out-of-vocabulary Words

4.1. Project Description and Motivation

Common approaches to text categorization essentially rely either on n-gram counts or on word embeddings. This presents important difficulties in highly dynamic or quickly-interacting environments, where the appearance of new words and/or varied misspellings is the norm. A paradigmatic example of this situation is abusive online behavior, with social networks and media platforms struggling to effectively combat uncommon or non-blacklisted hate words.

To better deal with the above issues in those fast-paced environments, we propose using the error signal of class-based language models as input to text classification algorithms. In particular, we train a next-character prediction model for any given class, and then exploit the error of such class-based models to inform a neural network classifier.

Our contribution is the adaptation of recurrent neural networks, that inherently tend to “remember” longer sequences than traditional sequence models like HMMs, into models that handle Out Of Vocabulary (OOV) words. Also, a custom loss function (objective) was designed in order to take into account the error-signals methodology described above.

4.2. Methodology

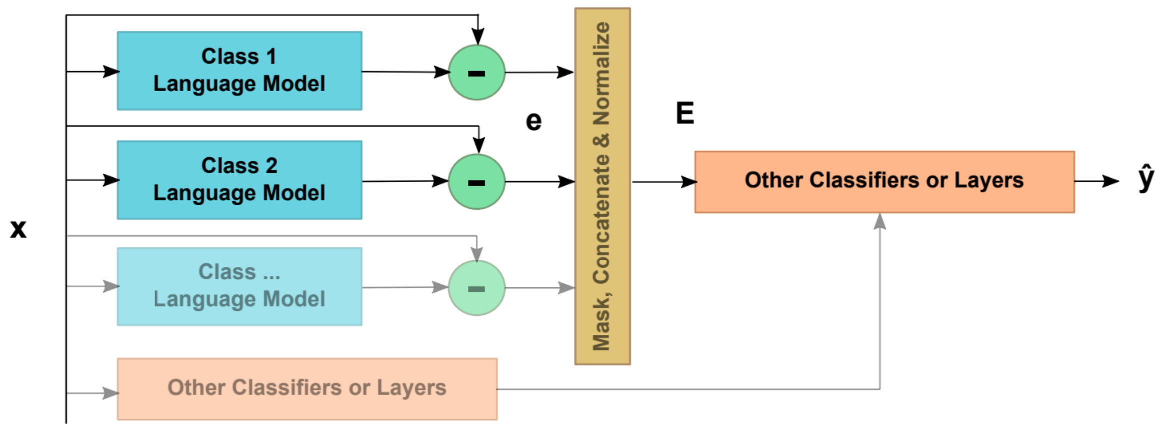

Class 1
Language Model
Class 2
Language Model
Class ...
Language Model
Other Classifiers or Layers
x
e
Mask, Concatenate & Normalize
E
Other Classifiers or Layers
ŷ

Figure 10. Text categorization following a two-step approach

We propose to perform text categorization following a two step approach (Figure 10). Firstly, a language model for next-character prediction is trained with the data corresponding to each single category. Secondly, we use a normalized, sequential measurement of the performance of those language models as input to a neural network classifier. In a sense, we aim to shift from ‘what is said’ (ability to describe) to ‘the way of saying’ (ability to predict). The idea is that the language model should be tailored to a given category, but without developing a strong dependence on specific characters that are frequent or representative for that category.

We train our models using Adam gradient descent until we do not see an improvement on the validation set for 4 epochs. For the language model, we use layer dimensionalities 250 (embedding) and 500 (GRU). For the classification model, we use a dropout of 0.5 and dimensionality 100. At classification time, the language model’s weights remain frozen. To implement our models we used Keras with Theano.

4.3. Results

Setup	Word NB	Char NB	Word NB-LR	Char NB-LR	Char RNN	Proposed
Easy	0.900 (0.091)	0.849 (0.052)	0.912 (0.113)	0.902 (0.102)	0.858 (0.141)	**0.951** (0.080)
Hard	0.634 (0.109)	0.663 (0.080)	0.580 (0.089)	0.679 (0.038)	0.619 (0.101)	**0.705** (0.059)

Table 5. AUC scores for the considered baselines (see text) and the proposed approach. A null randomized model yields 0.514 (0.082) and 0.503 (0.090) for the Easy and Hard setups, respectively

We compare the proposed approach with 5 of the most common and competitive baselines in abusive language detection (Djuric et al., 2015; Mehdad and Tetreault, 2016). The first two are based on a naiıve Bayes classifier using the TF-IDF of both word and character n-grams (NB). The next two are based on the approach proposed by Wang and Manning (2012): similarly, word and character n-grams are constructed, but NB ratios are calculated, and logistic regression is used as a classifier (NB-LR). The rationale behind this choice over other classifiers (e.g SVMs) was that Wang and Manning (2012) found that for short text sentiment tasks, NB actually does performs better than SVMs. For the previous non-neural count-based baselines, we use all combinations of {1,2,3}-grams with cutoff frequencies of 20 (NB) and 100 (NB-LR). These cutoff frequencies were chosen in-sample in order to maximize the performance of these baselines. The fifth baseline is a character-based RNN using a time-distributed embedding layer, followed by a PReLU activation, a GRU, a dense layer with PReLU activation, and a dense layer with sigmoid output. We try different values for the dimensionality of the aforementioned layers and finally use 200 (embedding), 400 (GRU), and 200 (dense).

The result of the comparison is reported in Table 5. We see that, among the 5 baselines, there is no clear winner for the two setups. Wordbased NB-LR performs best in the Easy setup and character-based NB-LR performs best in the Hard setup. Nonetheless, the proposed approach outperforms them by 4.2 and 3.8%, respectively. Compared to the average baseline performance, we observe a relative improvement of 7.5 and 11.0%. We also note that the standard deviation of the proposed approach (across runs) is comparatively low with respect to the baselines.

4.4. Conclusion

In this work, we deal with hate speech detection and with abusive OOV words. To this extent we propose to use the error signal of class-based language models as input to text classification algorithms. In particular, we train a next character prediction model for any given class, and then exploit the error of such class-based models to inform a neural network classifier. This way, we intend to shift from the ‘ability to describe seen documents to the ‘ability to predict unseen content. Experiments using OOV splits from abusive tweet data show promising results, outperforming competitive text categorization strategies by 4– 11%. We envision a number of potential extensions for the proposed approach: adding an ‘all-class’ predictor to the language models, improve (or learn) the error sequence normalization, studying the effect of adding further classifiers in parallel to the proposed classification model, ways of fusing those, play with class-based sentence or paragraph embeddings, etc. Generalizing the architecture to longer sequences is a main task for further research, perhaps considering recurrent, quasi-recurrent (Bradbury et al., 2017), or convolutional networks for the classification stage. A qualitative analysis of the output of above classifiers is planned as future work. Due to the fact that our data are weakly-labeled and come from dictionaries, we also plan to shift our focus to real world, curated data sets of abusive language, as well as evaluate our models on human-annotated crowd-sourced data.

5. A unified Deep Learning Architecture for Multi-facet Abuse Detection

5.1. Project Description and Motivation

In the past few years, the ubiquity of social media has raised concerns about abusive behavior. Unfortunately, this problem is difficult to deal with and multifaceted, covering things like hate speech, offensive, sexist and racist language, aggression, cyberbullying, harassment and trolling [23,26]. Each form of abusive behavior has its own characteristics, and manifests differently, depending on the type of social media, the users participating in it, and the particular platform it takes place on. Indeed, popular social media platforms like Twitter and Facebook are not immune to abusive behavior, even though they have substantial resources to deal with it [22].

This type of behavior is harmful both socially, reducing the proclivity of user interaction, as well as from a business perspective. Simply put, users suffer, and this in turn causes businesses bottom line to suffer. For example, concerns about racist and sexist attacks on Twitter may have impacted a potential sale of the company [3]. It is apparent that social media platforms have not adequately addressed this problem; instead, they have entered into something akin to an arms race with abusers who adapt to announced detection mechanisms as they are deployed.

The research community has also made attempts at detecting abusive behavior. For example, hate speech [4,11,25,27], cyberbullying [6,10,21], abusive [7,9,18] and offensive [8,17,28] language, racism [15,16], and sexism [13] have all been targets in the literature. However, these solutions are typically custom built and tuned for a specific platform and type of abusive behavior, as opposed to being a more general solution.

Ideally, we would like a more universal solution, one that takes into account some common properties of all types of abusive behavior. Thus, in this work, we tackle the problem using deep learning and design a unified architecture able to digest and combine multiple attributes to detect abusive behavior. We use: i) users' past activity, ii) user profiles, iii) social graphs, iv) content metadata, and v) content itself to classify a given post as abusive or not. The relevant values of these attributes are learned via a novel deep learning architecture combined with bleeding edge training methodology that captures the many facets of abusive behavior.

We show the effectiveness of our approach across four datasets, each of which captures a different facet of abusive behavior: i) cyberbullying, ii) sarcasm, iii) hate, and iv) offensive. Our results indicate that our architecture, in conjunction with the training methodology, substantially outperforms both baseline and state-of-the-art solutions.

More concretely, we make the following contributions:

- We are the first to study in depth the application of deep learning on the detection of abuse in multiple forms with a unified deep learning architecture.
- We introduce a novel, deep-learning-based classifier that combines multiple attributes and demonstrate that the combination of these inputs substantially boosts classification performance.
- We demonstrate that a naive training methodology is suboptimal, opting instead to use a technique that alternates training between distinct input category paths.
- We test our unified methodology on four datasets to detect a wide spectrum of abusive behavior \emph{without the need for any tuning or reconfiguring}.
- We demonstrate that our unified approach performs better than state-of-art techniques built specifically for each of the datasets we use.
- We provide our implementation to the community.

5.2.Methodology

Our overarching goal is to produce a classifier that can detect nuanced forms of abusive behavior. There is a host of literature that tackles this problem, and uses a variety of approaches to do so. A key takeaway from the majority of this work is that building a model based on text is outperformed by those that additionally take domain specific features into account. Unfortunately, this is a cumbersome process, with slightly different problems and data sources requiring specially constructed models using different architectures. Ideally, we would prefer to have a single model/architecture that incorporates domain specific features, as well as text content and is performant on a large number of abusive content detection tasks. To that end, we present a unified classification model for abusive behavior. Our approach is treating i) raw text, and ii) domain specific metadata, separately at first, and later combining together into a single model.

In the remainder of this section, we provide details on how our final network is built from its component parts, paying particular attention to the specifics of how to train such a multi-input model. Firstly, we present the two individual classifiers that we later fuse together: the text (Sec 5.2.1) and the metadata classifier (Sec 5.2.2). Later, we demonstrate how we can combine these two either as an ensemble or a single network (Sec 5.2.3). Finally, we present the different ways of training such a multipath network (Sec 5.2.4).

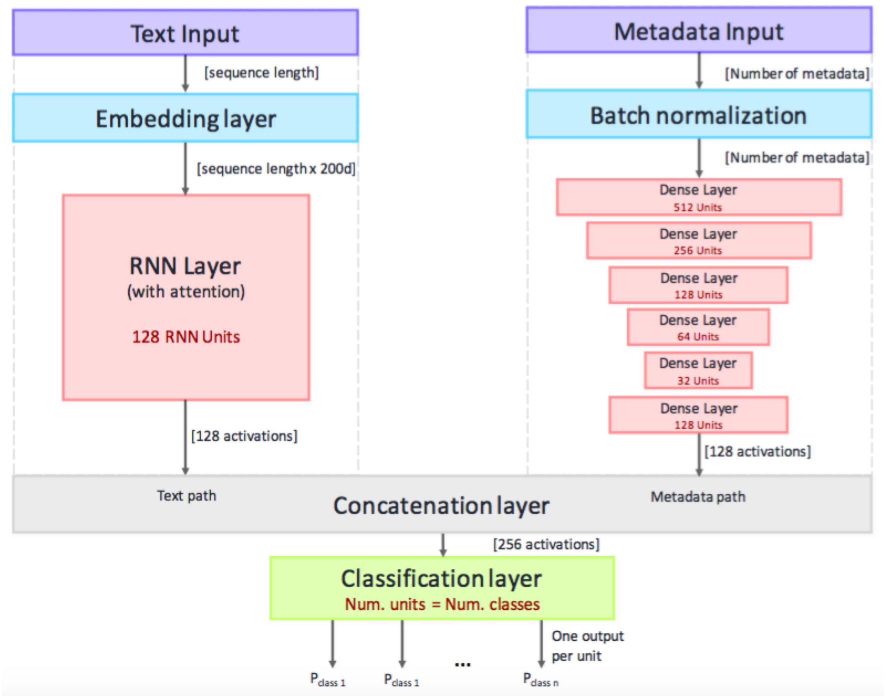


Figure 11. The combined classifier. The output of the two individual paths are concatenated and a classification layer is added over the merged data

5.2.1.1. Text Classification Network

This classifier only considers the raw text as input. There are several choices for the class of neural network to base our classifier on. We decided to use Recurrent Neural Networks (RNN) since they have proven successful at understanding sequences of words and interpreting their meaning. We experimented with both character-level and word-level RNNs and found the latter to be the most performant across all our datasets.

Text preprocessing: Before feeding any text to the network we need to transform each sample to a sequence of words. As neural networks are trained in mini-batches, every sample in a batch must have the same sequence length (number of words). Tweets containing more words than the sequence length will be trimmed, whereas tweets with less words will be left-padded with zeros (the model will learn they carry no information). Ideally, we want to setup a sequence length that is large enough to contain most text from the samples but avoid outliers as they waste resources (feeding zeros in the network). We thus take the 95th percentile of length (with respect to the number of words) in the input corpus as the optimal sequence length. For tweets, this results in sequences of 30 words (i.e., 5% of tweets that contain more than 30 words will be truncated). Sequences with less than this length are padded with zeros, which the model will learn carry no information.

We additionally remove any words that appear only once in the corpus, as they are most likely typos and can result in overfitting. Once preprocessed, the input text is fed to the network for learning.

Word embedding layer: The first layer of the network performs a word embedding, which maps each word to a high-dimensional (typically 25-300 dimensions) vector. Word embedding has proved to be a highly effective technique for text classification tasks, and additionally reduces the number of training samples required to reach good performance.

We settled on using pre-trained word embeddings from GloVe [19], which was constructed on more than 2 billion tweets. We choose the highest dimension embeddings (200) available, as these produce the best results across all our datasets. If a word is not found in the GloVe dataset, we initialize a vector of random weights, which the word embedding layer eventually learns from the input data.

Recurrent layer: The next layer is an RNN with 128 units (neurons). As mentioned previously, RNNs learn sequences of words by updating an internal state. After experimenting with several choices for the RNN architecture (Gated Recurrent Unit or GRUs, Long Short-Term Memory or LSTMs, and Bidirectional RNNs) we find that due to the rather small sequences of length in social media (typically less than 100 words per post, just 30 for twitter) simple GRUs were performing as well as more complex units. Additionally, to avoid overfitting we use a recurrent dropout with *p=0.5* (i.e., individual neurons were available for activation with probability 0.5), as it empirically provided the best results across our datasets. Finally, an attention layer [5] can be added as it provides a mechanism for the RNN to "focus"' on individual parts of the text that contain information that is related to the task. Attention is particularly useful to tackle texts that contain longer sequences of words (e.g., forum posts). Empirically, we find this only helps for texts that exceed 100 words and thus disable it for any classification task that involves tweets.

Classification layer: Finally, we use a fully connected output layer (a.k.a. Dense layer) with one neuron per class we want to predict and a softmax activation function to normalize output values between 0 and 1. The output of each neuron at this stage represents the probability of the sample belonging to each respective class. Note that this is the layer that will be sliced off when we fuse the text and metadata models into the final combined classifier.

5.2.1.2. Metadata Network

The metadata network considers non-sequential data. For example, on Twitter, it might evaluate the number of followers, the number of recent tweets, the location, account age, total number of tweets, etc., of a user.

Metadata Preprocessing: Before feeding the data into the neural network we need to transform any categorical data into numerical, either via enumeration or one-hot encoding, depending on the particulars of the feature. Once this has taken place, each sample is thus represented as a vector of numerical features.

Batch Normalization Layer: Neural network layers work best when the input data have zero mean and unit variance as it enables faster learning and higher overall accuracy.

We thus pass the data through a Batch Normalization layer that takes care of this transformation at each batch.

Dense layers: We use a simple network of several fully connected (dense) layers to learn the metadata. We design our network so that a bottleneck is formed. Such a bottleneck has been shown to result in automatic construction of high-level features [12,24]. In our implementation, we experimented with multiple architectures and we ended up using 5 layers of size 512, 245, 128, 64, 32, which provide good results across all our datasets. On top of this layer, we add an additional (6th) layer which ensures this network has the same dimensionality as the text-only network; this ends up enhancing performance when we fuse the two networks. Finally, we use *tanh* as our activation function since it works better with standardized numerical data.

Classification layer: As with the test only network, we use one neuron per class with softmax activation.

5.2.1.3. Combining the two classification paths

The two classifiers presented above can handle individually either the raw text or the metadata. To build a multi-input classifier we need to combine these two paths. There are two possible ways to perform such a task: i) just use the output of the two classifiers (probabilities of belonging to a given class) as input of a new classifier, or ii) combine the two classifiers on the previous layer that represents the automatically constructed features (as shown in Figure 11).

5.2.1.4. Combining as ensemble

For the ensemble, we train the text and metadata models separately. Each model provides a probability of a post belonging to a given class, when fed with its respective input. Once trained, we freeze these models (no additional training is possible) and combine their outputs (class probabilities) into a new classifier that resemble the last dense layer of the classifier (one unit per class with softmax activation). The weights of this extra layer are trained with a separate validation set.

5.2.1.5. Combined Classifier

Instead of combining an ensemble of pre-trained classifiers, neural networks allow us to create arbitrary combinations of layers and construct complex architectures that resemble graphs. Therefore, instead of training and then combining two separate classifiers, we can design from the beginning a single architecture that combines both paths before their inputs are squashed into classification probabilities (Figure 11). Therefore, we concatenate the text and metadata networks at their penultimate layer: i) the text path where sequences of raw text are input and 128 activations are produced (one for each RNN unit) and ii)~the metadata path where each input feature produces

128 activations. You can think of this architecture as merging together 128 automatically constructed features from each input and then attempting the final classification task based on this vector.

Contrary to traditional machine learning, this architecture allows us to mix a diverse set of data (sequences of text and discrete metadata features) without having to explicitly construct the text features (e.g., TF-IDF vectors). Furthermore, we utilize the power of word embedding that have been pre-trained on much larger datasets.

5.2.1.6. Training the Combined Network

While the combined architecture is straightforward, just training the whole network at once is not the most optimal way. In fact, there are several ways to train the combined network: below, we list some of the possible ways and in the Evaluation section we will compare their performance.

Training the entire network at once: The simplest approach is to train the entire network at once; i.e., to treat it as a single classification network with two inputs. However, the performance we achieve from this training technique is suboptimal: the two paths have different convergence rates (i.e., one of the paths might converge faster, and thus dominate any subsequent training epochs). Furthermore, standard backpropagation across the whole network can also induce unpredictable interactions as we allow the weights to be modified at both paths simultaneously.

Transfer learning: We can avoid this problem by **pre-training** the two paths separately and only afterwards join them together to construct the architecture of Figure 11.

This involves a number of steps:

1. Pre-train separately the text (Sec 5.2.1) and the metadata classifiers (Sec 5.2.2).
2. Afterwards, we **remove** the classification layer of each classifier, effectively exposing the activations of their penultimate layer. We treat these as the features that the two pre-trained networks have constructed based on their training.
3. We freeze the weights of both networks, so no further re-training of their weights is possible.
4. We add a concatenation layer and a classification layer, effectively transforming the separate models into the architecture of the combined model (Figure 11).
5. We train again the combined model. Only the final layer’s weights are trainable.

Notice that this model resembles an ensemble but with a key difference: the input is not the final class probabilities (num_{class} + num_{class} features) but the features learned by the previous layer of the pre-trained models (128 + 128 features).

Transfer learning with fine tuning: This approach is the same as above, except we do not freeze the weights on the original networks. The practical result of this is that our pre-training serves only to initialize the weights, which the fused network can later adapt when we merge the two paths.

Combined learning with interleaving: As discussed, standard training of the whole network at once may lead to poor performance due to the interaction of updating both data paths at once. The training approaches presented in the previous sections try to mitigate this problem by training the two paths separately and then concatenating the two pre-trained models together.

However, here we introduce a way that allows us to train the full network simultaneously while mitigating the aforementioned drawbacks. In fact, we demonstrate that this approach achieves the best results in all four datasets. To do this, we can design our training in a way that, at each mini-batch, data flow through the whole network but only one of the paths is updated. To do so, we train the two paths in an alternating fashion. For example, on even-numbered mini-batches the gradient descent only updates the text path whereas on odd-numbered batches the metadata ones. Finally, between epochs we also alternate the paths so both paths get a chance to observe the whole dataset. This results in a more optimal, balanced network as the gradient is only able to change one path at a time, thus avoiding unwanted interactions. At the same time, the loss function is calculated over the whole, combined, network (notice that the input does pass through the whole network).

This architecture was originally introduced by Hidasi et al [13], where they use parallel training on two recurrent neural networks for multimedia and textual features. In this case the architecture is used for video and product recommendations. However, our work is the first one to introduce it on text classification.

Dataset	Tweets	Classes	Users	TF	UF	NF
Cyberbullying [6]	6091	8.5% Bully 5.5% Aggressive 86% Normal	891	✓	✓	✓
Offensive [27]	16059	12% Racism 20% Sexism 68% None	1236	✓	✓	
Hate [9]	24783	6% Hate 77% Offensive 17% Neither		✓		
Sarcasm [20]	61075	10.5% Racism 89.5% Sexism	60255	✓	✓	

Table 6. A descriptive analysis of the datasets with information about the number of tweets and users, the distribution of the classes and whether or not we have the correspondent tweet- (TF), user- (UF) and network-based (NF) features

5.2.1.7. Experimental Setup

For our implementation we use Keras [1] with Theano [2] as back-end for the deep learning models implementation. We use the functional API to implement our multi-input single-output model. Finally, we run the experiments on a server that is equipped with three Tesla K40c GPUs.

In terms of training, we use *categorical cross-entropy* as loss function and *Adam* as the optimization function. A maximum of 100 epochs was allowed, but we also employed a separate validation set to perform *early stopping*: training was interrupted if the validation loss did not drop in 10 consecutive epochs and the weights of the best epoch were restored. All results shown here are based on 10-fold cross validation.

To implement the interleaved approach in Keras, we had to initialize two identical models *A* and *B* with the architecture shown in Figure 11. However, before compiling the models we introduce a single difference: the text-path of *A* is defined as non-trainable and, similarly, the metadata-path on *B* is also “frozen”. During training, at each mini-batch we alternate these models:

- If (*batch_number* + *epoch_number*) is even, then use model A (else, use B). Notice, the input will pass through both paths, however, the gradient will only update the weights of a single path.
- Copy the newly updated weights to the unused model. Now both models have equal weights.
- Repeat for next mini-batch.

At the end of this process we will have two identical models, each trained one-path-at-a-time. Our empirical results shown that mini-batches of 64 to 512 samples perform similar on all datasets and we chose 512 as it speeds up training.

Furthermore, it is important to notice that the same model (with the same architecture, number of layers, number of units and parameters) is used for all the datasets as we want to demonstrate the performance of this architecture across different tasks. The performance of the algorithm might increase even further if the parameters are tuned specifically for each task (e.g., using a larger network when there are more training samples). Overall, the model, excluding the pre-trained word embeddings, contains approximately 250,000 trainable parameters (i.e., weights).

Finally, except from each dataset's state-of-the-art, we also compare our results with a basic Naive Bayes model, using the TF-IDF weights for each tweet. For this baseline, we only use the raw text. First, we perform some basic preprocessing of the data; we convert all characters to lowercase and remove all stop words for 14 frequently spoken languages, as well as some twitter-specific stop words. Finally, we tokenize the tweet based on some Twitter-specific markers (hashtags, URLs, and mentions) and punctuation. Afterwards, we experiment with both Porter and Snowball stemmers, lemmatization, keeping the most frequent words and the combinations of all the previously mentioned. We find that the most efficient step is keeping only the most frequent words. We also experiment on the amount of frequent words we need to keep and find that the best results are yielded using the top *10k* words.

5.3. Results

In this section we present the classification performance of the proposed methodology on the four datasets. Next, we examine which are the inputs that contribute the most. Finally, we discuss about how the different training strategies affect the classification performance.

	AUC	**Accuracy**	**Precision**	**Recall**	**F1-score**
Cyberbullying Dataset (3 classes)					

Baseline Naive Bayes	0.73	0.88	0.88	0.88	0.88
Baseline Random Forest [6]	0.91	0.91	0.9	0.92	0.91
DL - Features Only	0.93	0.88	0.91	0.88	0.89
DL - Text Only	0.92	0.89	0.91	0.89	0.89
DL - Ensemble	0.85	0.8	0.89	0.8	0.83
DL - Text & Features (Feature Transfer)	0.95	0.9	0.92	0.9	0.9
DL - Text & Features (Feature Transfer FT)	0.95	0.9	0.91	0.9	0.91
DL - Text & Features (Naive Training)	0.94	0.89	0.9	0.9	0.9
DL - Text & Features (Interleaved)	0.96	0.92	0.93	0.92	0.93
Offensive Dataset					
Baseline Naive Bayes	0.79	0.81	0.81	0.81	0.81
Baseline Logistic Regression [27]	-	-	0.74	0.73	0.78
DL - Features Only	0.91	0.74	0.81	0.74	0.76
DL - Text Only	0.93	0.83	0.84	0.83	0.83
DL - Ensemble	0.93	0.85	0.86	0.85	0.85
DL - Text & Features (Feature Transfer)	0.95	0.85	0.86	0.85	0.85
DL - Text & Features (Feature Transfer FT)	0.95	0.86	0.87	0.86	0.86
DL - Text & Features (Naive Training)	0.93	0.85	0.86	0.86	0.86
DL - Text & Features (Interleaved)	0.96	0.87	0.88	0.87	0.87
Hate Dataset					
Baseline Naive Bayes	0.71	0.87	0.84	0.87	0.85
Baseline Logistic Regression [9]	0.87	0.89	0.91	0.9	0.9
DL - Features Only	0.75	0.61	0.8	0.61	0.66
DL - Text Only	0.91	0.87	0.89	0.87	0.88
DL - Ensemble	0.88	0.88	0.89	0.88	0.88

DL - Text & Features (Feature Transfer)	0.91	0.87	0.89	0.87	0.88
DL - Text & Features (Feature Transfer FT)	0.9	0.87	0.89	0.87	0.88
DL - Text & Features (Naive Training)	0.9	0.87	0.89	0.87	0.88
DL - Text & Features (Interleaved)	0.92	0.9	0.89	0.89	0.89
Sarcasm Dataset					
Baseline Naive Bayes	0.66	0.9	0.89	0.9	0.89
Baseline Logistic Regression [20]	0.7	0.93	-	-	-
DL - Features Only	0.96	0.92	0.94	0.92	0.92
DL - Text Only	0.81	0.89	0.89	0.89	0.89
DL - Ensemble	0.95	0.94	0.94	0.94	0.94
DL - Text & Features (Feature Transfer)	0.97	0.95	0.95	0.95	0.95
DL - Text & Features (Feature Transfer FT)	0.97	0.95	0.95	0.95	0.95
DL - Text & Features (Naive Training)	0.97	0.96	0.96	0.96	0.96
DL - Text & Features (Interleaved)	0.98	0.97	0.96	0.97	0.97

Table 7. Final results of the baselines and our experiments, for each one of the datasets

5.3.1.1. Classification Performance

As mentioned, we apply the same model over all 4 datasets and the results are summarized in Table 7. In all four models the proposed classifier (Interleaved) outperforms both the baseline and the state-of-the-art, as reported in recent publications. This happens for two reasons. First, the proposed approach that combines the raw text and the metadata achieves significantly higher performance when compared to the ones using a single attribute, as it takes advantage of the additional information from the users' profile and network. This is consistent across all four datasets. Second, the word embeddings allow us to transfer features that were constructed over a billion of tweets, and it enables us to model complex tasks with fewer samples (such as this one).

Looking at the four datasets, on the Cyberbullying dataset we observe that using a single attribute (text or metadata) we achieve better AUC (0.92 or 0.93, respectively) but worse accuracy (0.89 or 0.88, respectively) than the method proposed in the state-of-the-art (AUC 0.91, accuracy 0.91) [6]. However, the interleaved training of our model significantly outperforms the baseline (AUC 0.96, accuracy 0.92). Having said that, we need to mention here that the comparison with this dataset is not direct. The results reported on [6] are on user-level, while ours are on tweet-level. Therefore, while we can get an understanding on how well their data can be classified with our algorithm, we cannot parallelize the two cases. Nevertheless, the results we achieve on tweet-level are very high, which shows that our model distinguishes very well between the classes, regardless of the comparison.

On the offensive and sarcasm datasets, we significantly outperform the previous results as these works did not consider metadata. For example, in sarcasm we reach an AUC of 0.98 compared to 0.7 of the existing methodology, as in this case the text is not carrying significant information to detect if a tweet is sarcastic. However, the remaining features (user, network, sentiment, tweet-level metadata) do reveal such information. In the case of offensive dataset, there was no AUC information reported in the original publication, but the combination of text and metadata reached a precision and recall of 0.87-0.88, when compared to 0.73-0.74 of the baseline.

Finally, in the hate dataset the interleaved model is able to reach an AUC of 0.92. It is the only case where the precision and recall is similar to baseline, the one presented in [9], but our AUC and accuracy scores still outperform this baseline. This is due to the fact that we could not find any user or network metadata related to this dataset and, therefore, our classifier is only using the raw text and the text-based features.

5.3.1.2. Feature Importance

As mentioned earlier, we use a number of features extracted from the tweets and their authors, namely tweet-based, user-based, and network-based. These features play an important role on the improvement of the performance. When combined with the raw text they significantly increase all the metrics. However, not all of them have the same impact on the performance (some are more essential than others).

In order to determine how each one of these features affects our model, we experiment with the cyberbullying dataset and calculate their importance. The results on the AUC are presented in Table 3. We chose this dataset as we have all groups of features available (user, tweet, network, text) and, therefore, it is possible to examine how each of them contributes to the model. The results for other datasets are following similar trends and are omitted due to space limitations. Firstly, by examining individual features we observe that models which are built with individual metadata result in the poorest performance. For example, network-level features are the least descriptive (AUC of just 0.64) whereas tweet- and user-based are slightly better (AUC ~0.8). By far, raw text is the best feature for this task, as it can lead to a model with significantly higher AUC of 0.91.

Features	AUC
Network Only	0.641
Tweet Only	0.799
User Only	0.806
User & Tweet	0.887
Network & Tweet	0.908

Text Only	0.915
User & Network	0.915
All-metadata Only	0.923
Text & Tweet	0.93
Text & Network	0.931
Text & User & Tweet	0.933
Text & Network & Tweet	0.936
Text & User	0.938
Text & User & Network	0.955
All	0.961

Table 8. Feature Importance

Furthermore, combining two or more metadata classes together (user, network, tweet) does increase performance. This indicates that the information provided is not overlapping and it all adds to better performance. When all the metadata are combined, we reach an AUC of 0.923 which is higher than any other metadata combination. Moreover, using all metadata does result in better classification when compared to just using the text, showing that these features do carry as much predictive power as text does. Nevertheless, the strongest models were built when text is combined with metadata showing how much the raw text contributes in this classification task. For example, just combining text with user-level metadata is enough to reach an impressive performance (0.94 AUC). Adding network data bumps the performance to 0.955.

Finally, the best performance is reached when all attributes are used. This also demonstrates the fact that the metadata information does not overlap the information that can be extracted from raw text and this is why this model can be quite powerful and outperforms the state-of-art models for these tasks.

5.3.1.3. Training Methodology

Here, we compare the training methods discussed in the previous section, while Table 8 depicts the corresponding results. Firstly, we observe that training with the whole network at once (naive training) results to suboptimal performance (e.g., AUC of 0.94 in the cyberbullying dataset). The reason is that allowing the gradient descent to update both paths simultaneously might result in unwanted interactions between the two. For example, one path might converge faster than the other, dominating in the decisions. In fact, when we examine the standalone classifiers, we observe that the text classifier requires 25-40 epochs to converge whereas the meta-data classifier only requires 7-12. By training the whole network together the metadata side can start overfitting.

A step towards the right direction is to train each path separately, as individual classifiers, and then transfer the constructed features to a new classifier. This methodology slightly improves the results as it reduces the interactions between the two paths. Notice that, due to the fact that most of the network has been already trained, the additional layer converges after just 3-5 epochs. Finally, by employing alternate training we further improve the predictive power of the resulting model reaching 0.96 AUC in the cyberbullying. This shows that multi-input models such as this can benefit from alternate training. The reason is that this methodology avoids any interactions that might result when weights are updated simultaneously in both paths.

5.4. Conclusion

In this work, we demonstrate how a unique deep learning architecture can be trained using the right set of features, to achieve great classification results. Here we present some of the main insights we achieve from this work. First of all, we confirm that each of the attributes (text, user, network, tweet) do contribute in each task. Our proposed architecture can seamlessly combine this input into a single classification model. Also, we show that training a multi-input network is not straightforward. We introduce a methodology that alternates training between the two input paths to further increase performance in all the datasets that we have tried. We compared the proposed training paradigm with various other possible training methodologies (ensemble, feature transfer, concurrent training) and we show that it can significantly outperform them.

Additionally, we prove that a unified classifier can be possible. In our evaluation, we applied the exact same model architecture in all four datasets and demonstrated that it can efficiently handle each task. While fine-tuning the classifier parameters for each dataset can squeeze some more performance, the proposed methodology does beat the current state of the art. Furthermore, our proposed architecture seems to be flexible to other data. In this work, we demonstrate the ability of our approach to combine two different paths: text and metadata. However, one can simply concatenate more input paths to the architecture. For example, in an image classification problem, a CNN-based network can be used to extract image features and it could be joined with text information (tags and user comments) and image metadata (time and location taken, how many pictures the user has taken, the uploader's social network etc.). Similarly, in an audio classification task, an audio path can be merged with text and metadata. We leave this exploration as future work.

Finally, we conduct some preliminary research on whether this work can be generalized to other platforms. In this work we focused on Twitter to demonstrate how our unified approach works with the same set of features. However, the same methodology can be applied to other domains **with no modifications**. To demonstrate this, we run the same architecture over a dataset from a completely different domain, more specifically a gaming dataset from League of Legends. While we expected reasonable performance, we achieved an accuracy of 0.93 and an AUC of 0.89, beating the performance of even the easiest task of state-of-the-art, presented in [14]. These results provide a strong indication that our architecture is suitable for finding abusive behavior in a wide variety of domains.

5.5. Section References

[1] Keras: The python deep learning library. https://keras.io/.
[2] Theano. http://deeplearning.net/software/theano/.

[3] Did trolls cost Twitter 3.5bn and its sale? goo.gl/PlIL66, 2016.
[4] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma. Deep learning for hate speech detection in tweets. In Proceedings of the 26th International Conference on World Wide Web Companion, pages 759–760. International World Wide Web Conferences Steering Committee, 2017.
[5] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014.
[6] D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, and A. Vakali. Mean birds: Detecting aggression and bullying on twitter. arXiv preprint arXiv:1702.06877, 2017.
[7] Y. Chen, Y. Zhou, S. Zhu, and H. Xu. Detecting offensive language in social media to protect adolescent online safety. In Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Conference on Social Computing (SocialCom), pages 71–80. IEEE, 2012.
[8] I. Clarke and J. Grieve. Dimensions of abusive language on twitter. In Proceedings of the First Workshop on Abusive Language Online, pages 1–10, 2017.
[9] T. Davidson, D. Warmsley, M. Macy, and I. Weber. Automated hate speech detection and the problem of offensive language. arXiv preprint arXiv:1703.04009, 2017.
[10] K. Dinakar, R. Reichart, and H. Lieberman. Modeling the detection of textual cyberbullying. The Social Mobile Web, 11(02), 2011.
[11] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati. Hate speech detection with comment embeddings. In Proceedings of the 24th International Conference on World Wide Web, pages 29–30. ACM, 2015.
[12] B.Hidasi,M.Quadrana,A.Karatzoglou, and D.Tikk. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In Proceedings of the 10th ACM Conference on Recommender Systems, pages 241–248. ACM, 2016.
[13] A. Jha and R. Mamidi. When does a compliment become sexist? analysis and classification of ambivalent sexism using twitter data. In Proceedings of the Second Workshop on NLP and Computational Social Science, pages 7–16, 2017.
[14] H. Kwak, J. Blackburn, and S. Han. Exploring cyberbullying and other toxic behavior in team competition online games. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI ’15, pages 3739–3748, New York, NY, USA, 2015. ACM.
[15] I. Kwok and Y. Wang. Locate the hate: Detecting tweets against blacks. 2013.
[16] E. Lozano, J. Cedeño, G. Castillo, F. Layedra, H. Lasso, and C. Vaca. Requiem for online harassers: Identifying racism from political tweets. In eDemocracy & eGovernment (ICEDEG), 2017 Fourth International Conference on, pages 154–160. IEEE, 2017.
[17] Y. Mehdad and J. R. Tetreault. Do characters abuse more than words? In SIGDIAL Conference, pages 299–303, 2016.
[18] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang. Abusive language detection in online user content. In Proceedings of the 25th International Conference on World Wide Web, pages 145–153. International World Wide Web Conferences Steering Committee, 2016.
[19] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, 2014.
[20] A. Rajadesingan, R. Zafarani, and H. Liu. Sarcasm detection on twitter: A behavioral modeling approach. In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, pages 97–106. ACM, 2015.

[21] I. Riadi et al. Detection of cyberbullying on social media using data mining techniques. International Journal of Computer Science and Information Security, 15(3):244, 2017.
[22] Twitter trolls are now abusing the company’s bottom line. goo.gl/ JV9ZMH, 2016.
[23] H. Sanchez and S. Kumar. Twitter bullying detection. 2011.
[24] N. Tishby and N. Zaslavsky. Deep learning and the information bottleneck principle. In Information Theory Workshop (ITW), 2015 IEEE, pages 1–5. IEEE, 2015.
[25] W. Warner and J. Hirschberg. Detecting hate speech on the world wide web. In Proceedings of the Second Workshop on Language in Social Media, pages 19–26. Association for Computational Linguistics, 2012.
[26] Z. Waseem, T. Davidson, D. Warmsley, and I. Weber. Understanding abuse: A typology of abusive language detection subtasks. arXiv preprint arXiv:1705.09899, 2017.
[27] Z. Waseem and D. Hovy. Hateful symbols or hateful people? Predictive features for hate speech detection on twitter. In SRW@ HLT-NAACL, pages 88–93, 2016.
[28] G. Xiang, B. Fan, L. Wang, J. Hong, and C. Rose. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In Proceedings of the 21st ACM international conference on Information and knowledge management, pages 1980–1984. ACM, 2012.

6. Proactive Detection of YouTube Videos Targeted by Hate Attacks

6.1. Project Description and Motivation

As people move significant portions of their lives to online interactions, online aggression phenomena have become a significant problem [27]. Online aggression can happen on a one-to-one basis, in which the victim is repeatedly abused [13], or take the form of a coordinated harassment campaign, where the perpetrators loosely coordinate to deliver harmful content in a repetitive fashion [12, 14]. Such coordinated aggression campaigns take the name of raids [28]. Recent research showed that raids are often organized by fringe Web communities such as 4chan with the goal of attacking and undermining users on other platforms (e.g., YouTube, Twitter) who advocate for issues or policies that they do not agree with [28].

Despite the increasing relevance of online aggression, social networks lack adequate countermeasures to mitigate this problem. In general, abusive activity is generated by humans and not by automated programs, therefore automated systems that have been proposed to detect bots and fake accounts [9, 44] are unsuitable in this scenario. For these reasons, most platforms make use of reactive systems: users can report abusive accounts to the social network, which will then verify the claim and potentially suspend oenders [34]. The efficacy of this modus operandi has been brought to question both by researchers [14], and by the social network operators themselves, e.g., Twitter's transparency in admitting that the company could do a much better job in dealing with abuse [46].

In this project, we focus on identifying YouTube videos that are likely to be raided. We focus on YouTube because it is one of the top visited video platforms, with more than 1 billion users and 1 billion hours of videos watched every day [4]. With such large amount of activity, it is only expected to attract a lot of hate speech, something that prompted YouTube to announce new measures to crackdown such abusive behavior [11].

Also, previous work identified YouTube as the most heavily targeted platform by fringe communities such as 4chan [28]. This work analyzed the behavior of 4chan users active on the /pol/ board. The authors observed a particular action, among others: some users were posting YouTube links with sentences like you know what to do; suddenly, some of those videos were commented on by people interested in spreading hateful comments. In some cases, the behavior was attacking the video and the people commenting it positively. In other cases, the 4chan users supported the message of the video and attacked other commenters detracting from it. In either case, this action has been defined by the authors as a raid. In fact, the variety of topics covered by YouTube uploaders, in conjunction with its comment-oriented interface, make it an attractive platform for abusers. We obtained a ground truth dataset of raided YouTube videos from the authors of [28], and compared such videos with random YouTube uploads that were not targeted by raiders.

Previous work [28] showed how to detect whether a raid was occurring by examining the degree of synchronous commenting behavior between 4chan and YouTube, validating it in terms of the rate of hate comments in YouTube, as well as commenter account overlap. In this work package, instead, we take an orthogonal approach, looking raided videos alone, with the goal of gaining an understanding of which videos are likely to be raided in the future. To this end, we look at multiple features of YouTube videos, such as their title, category, what is displayed in the thumbnail image that is shown as a preview, and what the video is talking about (i.e., the audio transcript of the video). This information allows us to gain an understanding of what content attracts raids, in other words why these videos are raided.

Based on the insights gained from our analysis, we build a system that is able to tell at upload time whether a video is likely to be raided in the future. Our system leverages an ensemble of classifiers each looking at a different element of the video (metadata, thumbnails, and audio transcripts). Our approach allows video streaming platforms to assess whether a video that has been uploaded is at risk of being raided. Our ensemble detection algorithm performs very well, with 0:9 precision and 0:85 recall. Moreover, this approach could enable platforms to take countermeasures, by studying the comments posted on such videos and rate limit them, or temporarily blocking the comments to risky videos when a link to them is posted on fringe Web communities.

The contributions of the present work are as follows:

- We study the relatively new problem of raiding videos on YouTube from fringe communities like 4chan on a large set of 2.8k videos.
- We extract and use different properties of these videos in our analysis, including the transcripts of the videos, their metadata, as well as their thumbnails.
- We apply a novel, ensemble classier based on deep-learning methods which performs better than state-of-art methods and baselines.
- We oer suggestions on how current video platforms can apply our methodology to detect such attacks and mitigate their impact.

6.2. Methodology

We now introduce our approach to provide a *proactive* detection tool of videos targeted by hate attacks on online services, and on YouTube in particular. Our goal is to systematize this task and, for this, we use supervised machine learning. In particular, we build a set of machine learning classifiers, each of which focuses on a different set of features extracted from online videos. We next describe our proposed set of features, which is strongly motivated by the findings reported above. As not all videos contain the same information, we then present three independent classifiers that are triggered accordingly, and we finally show how to ensemble and properly balance the different predictions to provide one single decision.

6.2.1.1. General Overview

A high-level description of our detection system is depicted in Figure 12. The system is first trained using a dataset of *raided* and *non-raided* videos from different sources. The purpose of this phase is to obtain the following key elements that will later be used to predict whether a video could be targeted by hate attacks or not:

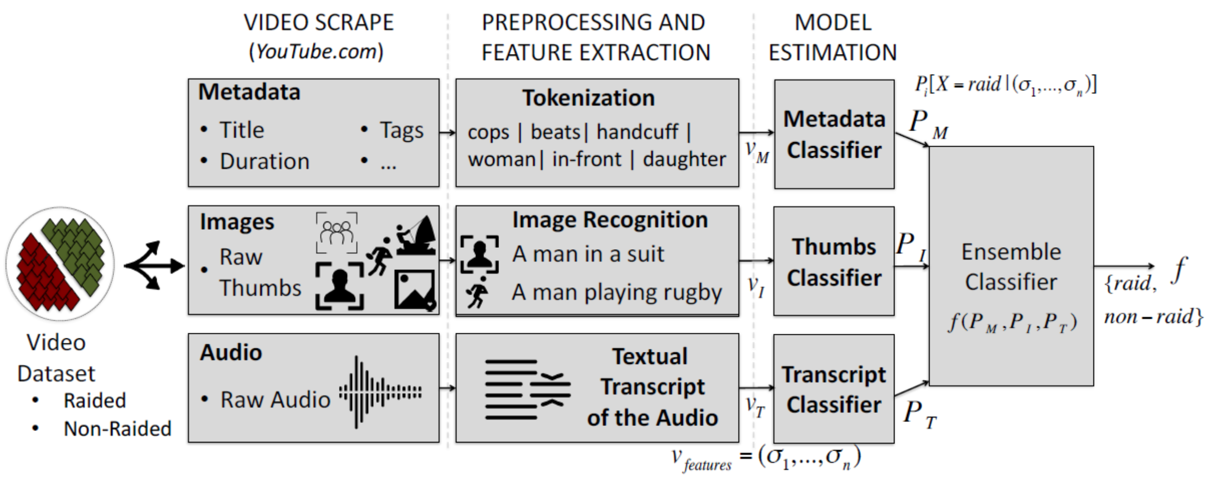


Figure 12. Architecture of our proactive detection system

(i) A set of prediction models that output the probability
$C_i(\sigma_1,\dots,\sigma_n) = P_i[Y = \mathtt{raid} \mid (\sigma_1,\dots,\sigma_n)]$

of each video *Y* being raid given a feature vector obtained from different elements *i* of the video. Each of this prediction models are referred as *individual classifiers*.

(ii) A weighted model $f(C) = \sum w_i \cdot C_i$

that combines all predictions in *C*, where each of the classifiers is weighted based on the performance obtained on a validation set. This set is different from the training set (used to build the individual probabilities), as well as different from the testing set (used to measure the efficiency of the classifiers). The process of weighting the individual predictors also serves as a way to calibrate the output of the probabilities. The final classifier will then output a decision based on a votation such that $f = \begin{cases} \mathtt{raid} & \text{if } f(C) < \epsilon \\ \mathtt{non\text{-}raid} & \text{otherwise,} \end{cases}$ where ε is a threshold typically set to $\left\lfloor \frac{\sum w_i}{2} \right\rfloor$.

For the sake of simplicity, we refer to the model presented in (ii) as *weighted-vote*. One can simplify the model by giving equal weight to all (typically) and obtaining a nominal value for before voting. In other words, applying a threshold for each (e.g., 0.5) and creating an equal vote among participants. We refer to this non-weighted voting system as *majority-vote*. One can further simplify the scheme by combining each individual prediction using the arithmetic mean of the output the probabilities. We refer to this as *average-prediction* system. Note the parameters used in both *majority-vote* and *average-prediction* are fixed and they do not require calibration. Thus, the validation set is not used in these two modes.

6.2.1.2. Feature Engineering

One of the key elements to build a robust classification system is to consider a diverse set of features. Our work mainly extracts features from three different sources: (i) structured attributes of the *metadata* of the video, (ii) features extracted from raw *audio* from the video, and (iii) features extracted from raw *images* (thumbnails) from the video. Based on a preprocessing, we transform non-textual elements of a video (i.e., audio and images) into text representation. Other textual elements such as the title of the video and the tags are kept as text.

These textual representations are then transformed into a fixed-size input space vector of categorical features. This is done by tokenizing the input text to obtain a nominal discrete representation of the words described on it. Thus, feature vectors will have a limited number of possible values given by the bag of words representing the corpus in the training set. When extracting features from the text, we count the number of occurrences a word appears in the text.

As in large text corpus certain words—such as articles—can appear rather frequently (without carrying meaningful information), we transform occurrences into a score based on two relative infrequency known as *term-frequency* and *inverse document-frequency* (namely, *tf-idfs*). Intuitively, the term frequency represents how "popular" a word is in a text (in the feature vector), and the inverse document-frequency represents how "popular" a word appear in the text, provided that it does not appear very frequently in other in the corpus (the feature space)

As for the thumbnails, after extracting the most representative descriptions per image, we removed the least informative elements and retained only entities (nouns), actions (verbs), and modifiers (adverbs and adjectives). Each element in the caption is processed to a common base to reduce inflectional forms and derived forms (known as stemming). We further abstracted the descriptions obtained from the images using *topic-modeling*.

In our current implementation, we extract features from only one image of the video (i.e., the thumbnail). As commented before, this is mainly because the thumbnails are typically purposely selected by the user as a video core and encapsulate semantically relevant context. However, we emphasize that our architecture supports the extraction of features from every frame in the video.

6.2.1.3. Prediction Models

We now present three independent classifiers to estimate the likelihood of a video being targeted by hate attacks. These classifiers are built to operate independently and at the moment a video is uploaded. In particular, each classier is designed to model traits from different aspects of the video as described above. Available decisions are later combined to provide one unanimous output.

The motivation behind the use of three different classifiers that are used in an ensemble is as follows. Features obtained from different parts of a video are inherently incomplete since some fields are optional and others might not report meaningful features. For instance, a music video might not report a lengthy transcript or a thumbnail might not contain distinguishable context. Thus, any reliable decision system should be able to exibly deal with incomplete video sections. Ensemble methods are suitable for this. Another reason behind it is that ensembles often perform better than a single classier [18].

Metadata and thumbnail cassiers. We build a prediction model such that Pi(X = raid) based on the features extracted from the metadata (PM) and from those extracted from the image thumbnails (P_I). The architecture of these two predictors is exible and accepts a range of classifiers. Our current implementation supports Random Forests, Extra Randomized Trees, and Support Vector Machines (SVM), both radial and linear. For the purpose of this work, we selected Random Forest (RF) as the base classier for P_T and SVM with linear kernel for P_M. Both SVM and RF have been successfully applied to different aspects of security in the past (e.g., fraud detection [10]) and have been shown to outperform other classifiers (when compared to 180 classifiers in real-world problems [22]).

Audio-transcript classifier Audio-transcript classier Before feeding the transcripts into the classier it's necessary to perform data cleaning. Firstly, we remove from the transcripts any words that have a transcription confidence ptrans < 0:5 as these are mostly wrong and unrelated to the video context. Including these terms will only confuse the classier. This only removes 9.2% of the words (91% of words have a condence of 0.5 or more). Furthermore, the transcripts contain a lot of repeated terms that are mostly exclamations such as uh uh, or hm hm, Finally, notice that the transcripts contain tags for non-verbal communication such as noise, laughter, etc. These were included in the text as they do carry predictive power.

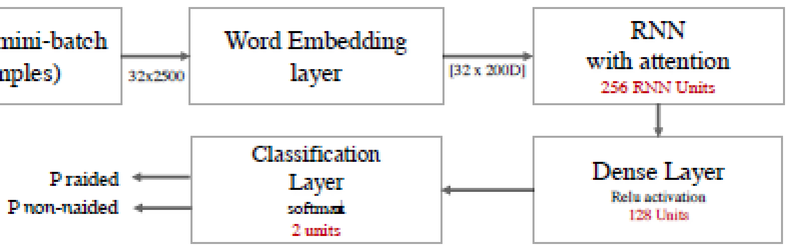


Figure 13. Architecture of the transcript classier. An Recurrent neural network with attention and pre-trained word embeddings was used.

There are several choices in terms of selecting a classier for long sequences of text. We experimented with transitional TF-IDF based approaches, with Convolutional Networks and with Recurrent Neural Networks. In the end we decided to use Recurrent Neural Networks (RNN) since they achieved the best performance as they are quite ecient at understanding sequences of words and interpret the overall context. Furthermore, we also use an attention mechanism [8] as they can really help the RNN to focus its attention to sequence of words that might indicate raidable videos. An illustration of the overall architecture is shown in Figure 13.

Before feeding any text to the network we need to transform each transcript to a sequence of words. Because neural networks process data in mini-batches, all transcripts within a mini-batch must have the same length (number of words). Transcripts with words that are larger than the sequence length will be trimmed whereas samples with less words are left-padded with zeros (the model will learn they carry no information). Ideally, we want to setup a sequence length that is large enough to contain the text from all samples in a mini-batch but not too long to waste resources (feeding zeros in the network). We thus take the 95th percentile of length (with respect to the number of words) in the input corpus as the optimal sequence length (i.e., 5% of samples will be truncated). This results in establishing a sequence length of 2500 words.

The rst layer of the network performs a word embedding, which maps each word to a high-dimensional vector. Word embedding has proved to be a highly effective technique for text classification tasks especially for tasks for which we have few training samples. We use pre-trained word embeddings from GloVe, which was constructed on more than 2 billion tweets that map each word into a 200-D vector. If a word is not found in the GloVe dataset, we initialize a vector of random weights, which the word embedding layer eventually learns from the input data.

Next, after experimenting with several choices for the RNN architecture we use a layer of 256 GRU units. To reduce over-tting we use a recurrent dropout with p = 0.5 as it empirically provided the best results across our datasets. On top of the recurrent layer, we add an attention layer as we are working with large sequence lengths (2500 words). The network at this stage outputs one activation at the end of the whole sequence (the whole transcript).

Connected to the recurrent part, we add a fully-connected (Dense) layer to mix the recurrent layer outputs and to gradually bring the dimensionality down to 128 units. Finally, the output layer is another Dense layer with one neuron per class. We use softmax as the activation function to normalize output values between 0 and 1.

In terms of training, we use mini-batches of 32 transcripts (i.e., the input is of shape 32x2500). We use categorical cross-entropy as loss function and Adam as the optimization function. A maximum of 100 epochs was allowed but we also employed a separate validation set to perform early stopping: training was interrupted if the validation loss did not drop in 10 consecutive epochs and the weights of the best epoch were restored.

Finally, for our implementation we use Keras [2] with Theano [3] as a back-end.

Ensemble classier. The objective of this ensemble method is to aggregate the predictions of the different base estimators. Each classier individually model the notion of videos that can potentially be targeted by hate attacks based on the set of features. The idea is that the decisions are then combined together so that the ensemble is able to make a more informed prediction. This not only allows to improve the robustness of the predictions (in terms of confidence), but also can result on a more accurate prediction. We design our ensemble method to take a weighted vote of the available predictions. In order to compute the best-performing set of weights, we estimate a function f that takes as input each of the individual probabilities and outputs the aggregated prediction. During training this function learns from a independent validation set, and it will be used during testing to weight each prediction model P_i.

For the decision function f we use a logistic distribution function that models how an expected probability in the validation set is affected by individual decisions P_i in a multiple regression. This can be interpreted as the sum of the weights w times the probability score p_i 2 P_i given by the individual classifiers in the weighted voting system.

6.3. Results

In this section we describe in detail our experimental setup and results while testing the performance of our method on the given dataset.

6.3.1.1. Experimental Setup

Ultimately, we want to show that our system can distinguish between raided and non-raided videos. That said, there are several subtasks that we also evaluate, each of which let us better understand the problem and how our classier performs.

First, we note that a video posted on /pol/ is likely quite different than a random video uploaded to YouTube. Since /pol/ is a rather focused community, and further its focus is quite fringe, we want to ensure that our classier is able to distinguish between random YouTube videos and all videos posted on 4chan. The outcome of this experiment gives us insight into our classifier's ability to discriminate between videos potentially raided vs those that are not really at risk at all. We refer to this setting as Experiment 1.

Second, we check how well our classier performs when working on random videos posted to YouTube and non-raided videos posted to 4chan. Here we can learn whether or not the classier has the potential to differentiate between videos 4chan is interested, but do not possess whatever element induces raids. We refer to this setting as Experiment 2.

Third, we experiment on distinguishing between videos that will be raided and random YouTube videos. Similar to the above, this evaluation lets us understand how well the classier performs with respect to learning what makes a video raided. We refer to this setting as Experiment 3.

Fourth, we look at only videos posted on 4chan and determine which are raided and which are not. This task is important because it ensures that the classier is not just able to predict whether a video was posted on 4chan, but whether or not said video will be raided. We refer to this setting as Experiment 4.

Finally, we evaluate our ultimate goal: whether or not the classier can discriminate between any non-raided video (i.e., regardless of whether it is a random YouTube video or one posted on 4chan) and videos that will be raided. We refer to this setting as Experiment 5.

Train, Validate, and Test Splits We split our datasets into three chunks, one for training, one for validation, and one for testing. Because we are dealing with highly disproportional classes (there are multiple order of magnitude more videos posted to YouTube than those posted to 4chan, let alone those that are raided), we balance each split. Leaving the training split unbalanced would make it difficult for our models to properly learn the differences between the different classes. The total number of videos in each split is proportionally sampled, assigning splits of 60%, 20% and 20% to the training, validation and testing sets. Table 9 summarizes all settings in our experiments together with the number of samples used.

ID	Description	Training	Test	Validation
Exp. 1	Random YouTube vs. all on 4chan	640+640	282+282	282+282
Exp. 2	Random YouTube vs. non-raided on 4chan	320+320	107+107	107+107
Exp. 3	Random YouTube vs. raided on 4chan	320+320	107+107	107+107
Exp. 4	Non-raided on 4chan vs. raided on 4chan	320+320	107+107	107+107
Exp. 5	All non-raided vs. raided on 4chan	320+320	107+107	107+107

Table 9. Number of samples used in our experiments. The sets are balanced as there is the same amount of samples per each class (class 1 samples+class 2 samples)

Evaluation Metrics For evaluating our approach we use a number statistical metrics that measure how efficient our prediction tasks perform across different angles. In particular, we use the accuracy, the precision, the recall, and F1-measure. In brief, precision measures the performance of our algorithm only for the values of the class of interest, while recall measures the proportion of positives that are correctly identified as such. Accuracy further reports the proportion of correct predictions made in both classes and the F1-measure represents the harmonic mean of precision and recall. These metrics are a good summary of the performance of an classier in terms of True Negatives (TN), False Negatives (FN), False Positives (FP), and True Positives (TP). However, these metrics are not suitable for comparing results across different experiments. For this, we plot the Area Under the Curve (AUC) which reports the TP-rate (recall) against the FP-rate (1 - recall).

6.3.1.2. Experimental Results

We next report the experimental results for the settings introduced above. Due to space restrictions, we only report details for each of the individual classifiers and the weighted-vote ensemble method where weights are t using Linear Regression as described in 3.3. For the cases of majority-vote and average-prediction, we found that they both underperformed when compared with weighted-vote.

Experiment 1. Here we study whether we can predict that a video will be posted in 4chan. Table 10 summarizes our results. Our results show that overall we can correctly identify 91% of the videos, while maintaining a high precision and F1-measure of 0.91. Judging by the results obtained by the individual classifiers, we can attribute these figures primarily to the metadata with a F1-measure of 0.89although both the audio-transcript and the thumbnail classifiers score highly as well. While the best performance is given by the metadata classier, the thumbnail reports the highest recall. Figure 4a depicts the ROC curve for all four classifiers. The individual AUCs are 0.62, 0.80, 0.96, 0.97, for the Thumbnails, Transcripts, Metadata, and Ensemble classifiers, respectively. The ensemble classier is the most effective at discriminating between random videos and those posted on 4chan, also evidenced by the maximum performance across the metrics in Table 10.

	Experiment 1				Experiment 2			
Classifier	**ACC**	**PRE**	**REC**	**F1**	**ACC**	**PRE**	**REC**	**F1**
Transcripts	0.73	0.72	0.82	0.77	*0.75*	*0.75*	0.61	0.72
Metadata	*0.89*	*0.89*	0.89	*0.89*	0.69	0.71	***0.94***	*0.81*
Thumbnails	0.65	0.69	*0.89*	0.78	0.69	0.71	***0.94***	*0.81*
Ensemble	**0.91**	**0.91**	**0.90**	**0.90**	**0.90**	**0.90**	0.90	**0.90**

Table 10. Results for Experiment 1 (videos posted on 4chan and random YouTube samples); and for Experiment 2 (non-raided videos posted on 4chan and random YouTube samples). ACC stands for accuracy, PRE for precision, and REC for recall.

Experiment 2. When trying to model videos that are posted on 4chan but have not been raided, we observed that the individual classifiers can correctly identify between 69% and 89% of the videos as shown in Table 3. Similar to the previous case, the best performance is given by the metadata classier except in terms of false negatives, where the thumbnails classier performs best. This setting also reports remarkably high F1-measures (i.e., 0.9) when combining all classifiers into our ensemble. Interestingly, in this setting the thumbnails outperform the rest of the classifiers in terms of recall. However, in terms of ROC curves (see Figure 4b, where the thumbnails classier achieves an AUC of 0.59, the transcripts classier hits 0.79, the metadata classier hits 0.96, and the ensemble reaches 0.97), both the transcripts and the thumbnails describe a lower area than for Experiment 1.

With these experiments we show that we can characterize the traits that make a video special enough to be posted to 4chan, but not attract raiding behavior. This is of interest to service operators because, even though a video might not get raided, the fact that it has attracted attention of a potentially abusive community is valuable information. For example, while it might not be necessary to take immediate action on the video, the operator could monitor the account and the users that viewed the video for future abuse (either as perpetrators or victims).

Experiment 3. Table 4 shows the results for the case where we model the differences between random videos on YouTube and raided videos that appear on 4chan. One can see that the metadata classier is the most effective individual classier with a highest precision than even the ensemble. That said, the thumbnails classier does a better job of finding more raided videos (i.e., it has the highest recall overall). When this is combined into the ensemble, we achieve a false positive rate of less than 10%, while detecting nearly 86% of raided videos. The high efficiency of the ensemble classier having low levels of false positives is shown also in the ROC curve depicted in Figure 4c. For this experiment, we see AUCs of 0.62, 0.86, 0.94, and 0.96 for the thumbnails, transcripts, metadata, and ensemble classifiers.

Results show that we are can understand what are the attributes that might be influencing the decision to raid a YouTube video.

	Experiment 3				**Experiment 4**			
Classifier	**ACC**	**PRE**	**REC**	**F1**	**ACC**	**PRE**	**REC**	**F1**
Transcripts	0.74	0.87	0.61	0.71	*0.70*	*0.73*	0.56	0.63
Metadata	*0.89*	***0.91***	0.83	*0.87*	*0.70*	0.68	***0.63***	*0.66*
Thumbnails	0.66	0.71	***0.86***	0.78	0.50	0.44	0.38	0.41
Ensemble	**0.89**	0.91	0.86	**0.88**	**0.72**	**0.75**	0.59	**0.66**

Table 11. Results for Experiment 3 (raided videos from random YouTube samples); and for Experiment 4 (raided videos and non-raided videos posted on 4chan)

Experiment 4. We then evaluate how well our models discriminate between raided videos posted to 4chan and non-raided videos also posted to 4chan. Results show that this is indeed the most challenging task. Intuitively, these videos are much more similar to each other than those found randomly on YouTube. This is because the 4chan community share common interests regardless of whether the video is later chosen to be raided. As shown in Table 11, about 72% of the videos are correctly classified in our best setting, where the F1-measure is at 0.65. The ROC curve depicted in Figure 4d shows how the ensemble is penalized by the low performances reported by the thumbnails and the ensemble is, at points, drown under the curve of the metadata and the transcripts. The individual classifiers reach AUCs of 0.55, 0.74, 0.75, and 0.78 for the thumbnails, transcripts, metadata, and ensemble classifiers, respectively.

Experiment 5. Finally, we report in Figure 4e results on how we perform at classifying raided videos and non-raided (regardless of whether the latter are a random YouTube video or a non-raided one posted on 4chan). Our results show that can correctly classify about 71% of the videos, with an the F1-measure leaning towards 0.7. The ROC curve displayed in Figure 4e), where we observe a similar situation with the previous experiment the metadata classier line is in some operational settings higher than the ensemble. Here, individual classifiers have AUCs of 0.52 for thumbnails, 0.64 for transcripts, 0.76 for metadata, with the ensemble reaching 0.77.

Classifier	Accuracy	Precision	Recall	F1-measure
Transcripts	0.64	*0.67*	0.45	0.54
Metadata	*0.69*	0.66	*0.68*	*0.67*
Thumbnails	0.53	0.54	0.60	0.57
Ensemble	**0.72**	**0.68**	0.70	**0.69**

Table 12. Results for Experiment 5, where we try to classify raided videos from non-raided YouTube and 4chan videos.

6.4. Conclusion

In this work, we introduce a system that aims to flag at upload-time whether a video is likely to be raided. Our results indicate that even single-input classifiers that use metadata, thumbnails or audio transcripts can be effective. However, an ensemble of these classifiers can significantly improve the detection performance and result in deployable early-warning systems. Furthermore, we demonstrate that our methodology can also associate videos with the fringe communities that are likely to raid them (e.g., videos that are typically attacked by the 4chan community) and, therefore, warn video providers about the possible source of a raid. In terms of future, work we plan to try more elaborate deep-learning methods that will allow us to combine directly audio, video and metadata into a single classifier. Furthermore, we plan to look into raids from other communities such as reddit, twitter, etc.

6.5. Section References

[1] [n. d.]. CMU Pronouncing Dictionary. http://www.speech.cs.cmu.edu/cgi-bin/cmudict. ([n. d.]). Accessed: 2017-10-30.

[2] [n. d.]. Keras: The Python Deep Learning library. https://keras.io/. ([n. d.]).

[3] [n. d.]. Theano. http://deeplearning.net/software/theano/. ([n. d.]).

[4] 2017. YouTube for the press. https://www.youtube.com/yt/about/press/. (2017).

[5] Swati Agarwal and Ashish Sureka. 2014. A Focused Crawler for Mining Hate and Extremism Promoting Videos on YouTube. In Proceedings of the 25th ACM Conference on Hypertext and Social Media (HT '14). ACM, 294296.

[6] Nisha Aggarwal, Swati Agrawal, and Ashish Sureka. 2014. Mining YouTube metadata for detecting privacy invading harassment and misdemeanor videos. In 12th Annual Conference on Privacy, Security and Trust, PST. 8493.

[7] Saleem Alhabash, Jong hwan Baek, Carie Cunningham, and Amy Hagerstrom. 2015. To comment or not to comment?: How virality, arousal level, and commenting behavior on YouTube videos aect civic behavioral intentions. Computers in Human Behavior 51, Part A (2015), 520 531.

[8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014).

[9]Fabricio Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. 2010. Detecting spammers on twitter. In CEAS, Vol. 6.

[10] Siddhartha Bhattacharyya, Sanjeev Jha, Kurian Tharakunnel, and J Christopher Westland. 2011. Data mining for credit card fraud: A comparative study. Decision Support Systems 50, 3 (2011), 602613.

[11] Ali Breland. 2017. YouTube cracking down on hate speech. http://thehill.com/policy/technology/347868-google-launches-initiative-to-reduce-hateful-content-on-youtube. (2017).

[12] Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. 2017. Hate is not Binary: Studying Abusive Behavior of #GamerGate on Twitter. In ACM Hypertext.

[13] Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. 2017. Mean Birds: Detecting Aggression and Bullying on Twitter. In International ACM Web Science Conference.

[14] Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, and Athena Vakali. 2017. Measuring #GamerGate: A Tale of Hate, Sexism, and Bullying. In WWW CyberSafety Workshop.

[15] Maura Conway and Lisa McInerney. 2008. Jihadi Video and Auto-radicalisation: Evidence from an Exploratory YouTube Study. Springer Berlin Heidelberg, 108118.

[16] YouTube CreatorAcademy. [n. d.]. https://creatoracademy.youtube.com/page/lesson/thumbnails\#yt-creators-strategies-5. ([n. d.]). Online; accessed October 2017.

[17] M. Dadvar, Rudolf Berend Trieschnigg, and Franciska M.G. de Jong. 2014. Experts and Machines against Bullies: A Hybrid Approach to Detect Cyberbullies. Springer Verlag, 275281.

[18] Thomas G. Dietterich. 2000. Ensemble Methods in Machine Learning. In Proceedings of the First International Workshop on Multiple Classier Systems.

[19] Ekaterina Egorova and Jordi Luque Serrano. 2016. Semi-Supervised Training of Language Model on Spanish Conversational Telephone Speech Data. Procedia Computer Science 81, Supplement C (2016), 114 120. https://doi.org/10.1016/j.procs.2016.04.038 SLTU-2016 5th Workshop on Spoken Language Technologies for Under-resourced languages 09-12 May 2016 Yogyakarta, Indonesia.

[20] Daniel Ballcells Eichenberger. 2016. Speech activity detection: Application-specic tuning and context-based neural approaches. Bachelor Thesis. Universitat Politcnica de Catalunya, UPC.

[21] Mattias Ekman. 2014. The dark side of online activism: Swedish right-wing extremist video activism on YouTube. MedieKultur: Journal of media and communication research 30, 56 (2014), 21.

[22] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. 2014. Do we need hundreds of classiers to solve real world classication problems? The Journal of Machine Learning Research (JMLR) 15, 1 (Jan. 2014), 31333181.

[23] John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. SWITCHBOARD: Telephone Speech Corpus for Research and Development. In Proceedings of the 1992 IEEE International Conference on Acoustics, Speech and Signal Processing - Volume 1 (ICASSP'92). IEEE Computer Society, Washington, DC, USA, 517520. http://dl.acm.org/citation.cfm?id=1895550.1895693

[24] V. Goel, S. Kumar, and W. Byrne. 2004. Segmental Minimum Bayes-Risk Decoding for Automatic Speech Recognition. IEEE Transactions on Speech and Audio Processing 12, 3 (2004), 234249.
[25] Ramesh A Gopinath. 1998. Maximum likelihood modeling with Gaussian distributions for classification. In Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on, Vol. 2. IEEE, 661664.
[26] Michael Green, Ania Bobrowicz, and Chee Siang Ang. 2015. The lesbian, gay, bisexual and transgender community online: discussions of bullying and self-disclosure in YouTube videos. Behaviour & Information Technology (February 2015), 19.
[27] Dorothy Wunmi Grigg. 2010. Cyber-aggression: Definition and concept of cyberbullying. Australian Journal of Guidance and Counselling 20, 02 (2010).
[28] Gabriel Emile Hine, Jeremiah Onaolapo, Emiliano De Cristofaro, Nicolas Kourtellis, Ilias Leontiadis, Riginos Samaras, Gianluca Stringhini, and Jeremy Blackburn. 2017. Kek, Cucks, and God Emperor Trump: A Measurement Study of 4chan's Politically Incorrect Forum and Its Effects on the Web. In ICWSM.
[29] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. Neural Comput. 9, 8 (Nov. 1997), 17351780. https://doi.org/10.1162/neco.1997.9.8.1735
[30] Bo-June Paul Hsu and James R Glass. 2008. Iterative language model estimation: efficient data structure & algorithms.. In in Proc. Interspeech. 841844.
[31] Mallory Hussin, Savannah Frazier, and J. Kevin Thompson. 2011. Fat stigmatization on YouTube: A content analysis. Body Image 8, 1 (2011), 90 92.
[32] L.A Jnson. 2013. Flaming motivation in YouTube users as a function of the traits Disinhibition seeking, Assertiveness and Anxiety? Technical Report.
[33] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 31283137.
[34] Imrul Kayes, Nicolas Kourtellis, Daniele Quercia, Adriana Iamnitchi, and Francesco Bonchi. 2015. The Social World of Content Abusers in Community Question Answering. In WWW.
[35] Kyounghee Kwon and Anatoliy Gruzd. 2017. Is Aggression Contagious Online? A Case of Swearing on Donald Trump's Campaign Videos on YouTube. 50 (01 2017), 2165.
[36] K. Hazel Kwon and Anatoliy Gruzd. 2017. Is offensive commenting contagious online? Examining public vs interpersonal swearing in response to Donald Trump's YouTube campaign videos. Internet Research 27, 4 (2017), 9911010.
[37] Patricia G. Lange. 2014. Commenting on YouTube rants: Perceptions of inappropriateness or civic engagement? Journal of Pragmatics 73, Supplement C (2014), 53 65. The Pragmatics of Textual Participation in the Social Media.
[38] Jordi Luque, Carlos Segura, Ariadna Snchez, Mart Umbert, and Luis Angel Galindo. 2017. The Role of Linguistic and Prosodic Cues on the Prediction of Self-Reported Satisfaction in Contact Centre Phone Calls. In Proc. Interspeech 2017. 23462350. https://doi.org/10.21437/Interspeech.2017-424
[39] Shivraj Marathe and Kavita P. Shirsat. 2015. Approaches for Mining YouTube Videos Metadata in Cyber bullying Detection. 4 (05 2015), 680 684.
[40] Peter J. Moor, Ard Heuvelman, and Ria Verleur. 2010. Flaming on YouTube. Computers in Human Behavior 26, 6 (November 2010), 15361546.

[41] A. Oksanen, D. Garcia, A. Sirola, M. Nsi, M. Kaakinen, T. Keipi, and P Rsnen. 2015. Pro-Anorexia and Anti-Pro-Anorexia Videos on YouTube: Sentiment Analysis of User Responses. Journal of Medical Internet Research 17, 11e256 (October 2015).
[42] Jaimie Yejean Park, Jiyeon Jang, Alejandro Jaimes, Chin-Wan Chung, and Sung-Hyon Myaeng. 2014. Exploring the User-generated Content (UGC) Uploading Behavior on Youtube. In WWW Companion. 529534.
[43] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. 2011. The Kaldi Speech Recognition Toolkit. In IEEE 2011 Workshop on Automatic Speech Recognition and Understanding. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB.
[44] Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. 2010. Detecting spammers on social networks. In ACSAC.
[45] Ashish Sureka, Ponnurangam Kumaraguru, Atul Goyal, and Sidharth Chhabra. 2010. Mining YouTube to Discover Extremist Videos, Users and Hidden Communities. Springer Berlin Heidelberg, Berlin,Heidelberg, 1324.
[46] The Guardian. 2015. Twitter CEO: We suck at dealing with trolls and abuse. goo.gl/6CxnwP. (2015).
[47] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2016. Show and tell: Lessons learned from the 2015 MSCOCO image captioning challenge. IEEE Transactions on Pattern Analysis and Machine Intelligence (2016).
[48] Andrew Weaver, Asta Zelenkauskaite, and Lelia Samson. 2012. The (Non)Violent World of YouTube: Content Trends in Web Video. Journal of Communication 62, 6 (December 2012), 1065?1083.

7. Honeypots (Ongoing Work)

This is an ongoing word that commenced as part for D4.1.

7.1. Project Description and Motivation

The rise in popularity of online social networks (OSN) has provided avenue for increase in global connectedness of the social graphs of individuals and businesses alike. For example, Facebook pages give businesses a platform to interact with their customers while a tweet on a particular topic may attract the attention of others to connect to the owner of the Twitter handle. However, since OSN represents a digital version of the real world/physical social graph, it also exhibits the challenges observed in the real world. One of such challenges is bullying which is referred to as cyberbullying in OSN.

There have been several research studies that examine cyberbullying in different OSN platforms. A review of previous literature on cyberbullying show that there is limited research on how cyberbullying begins and there are no experiments of cyberbullying predator and victim behavior. Similarly, cyberbullying may not be everywhere as different OSN platforms might encourage or inhibit cyberbullying for different reasons. Therefore, in our study, we aim to answer the following research questions that help us bridge observed gaps in our understanding of cyberbullying. We select Facebook and Twitter as the OSN platform in our study because they are two of the most popular OSN platforms with API for accessing publicly available data.

RQ1: What are the characteristics of messages that attract negative reactions from other users?

RQ2: Do accounts with differing privacy settings (i.e., public profile but private wall and private profile and wall) get different reactions when they post similar contents with the same potential to offend?

RQ3: How does Facebook compare to Twitter i.e., do the same/different messages get the same/different reactions? Are characteristics of negative responders similar etc.?

7.2. Preliminary Methodology

To answer the research questions of this study, we have designed a set of experiments which we perform on two OSN platforms i.e., Facebook and Twitter.

First, we create 120 personas with equal number of gender type (i.e., male or female), age group (teenager or adult), privacy settings (public or private), and randomly generated traits. Using the personas created, we create “honey” accounts on both platforms. Second, using our knowledge from detecting controversial contents (c.f Section 9), we label 1% of tweets we obtain from Twitter that these accounts propagate using public API. Third, we monitor the interactions with the contents posted by these accounts, and finally, using different statistical metrics, we analyze these interactions.

7.3. Conclusion

In this study, we aim to shed more light on the understanding of cyberbullying. In particular, we investigate characteristics of messages that are more likely to attract cyberbullies. We also investigate whether the privacy settings of a user effects whether they get bullied or not. Finally, we also compare the prevalence of cyberbullying between Facebook and Twitter.

8. Behavioral Pattern Identification (Ongoing work)

A research effort was set-up and performed targeting the online and real-time classification of Cyberbullying and Online Sexual Harassment observed in online conversations such as online chat and comment discussions. The project researches the idea of identifying the behavior trails of both the predator and the victim in an online harassment conversation. As a behavior trails (patterns or trends) indicate and try to capture the switching process of behavior of a victim and a predator, as this is depicted by analyzing sentiment and affect in their online conversation.

Towards this end the project sets two main hypotheses:

1. Predators in online harassment activities usually follow specific patterns towards convincing their victim. These patterns can be revealed through the analysis of their chat messages.
2. Victims of online harassment experience different affects during this process. These affects can be revealed by the analysis of their chat messages.

The project follows a collective approach for analyzing chat conversations. Instead of analyzing individual chat lines we will attempt to organize chat lines from the same person (predator or victim) into common behavior blocks (sessions). These blocks allow the identification of specific patterns of behavior and how these affect the behavior or reaction of the chat responder.

This is an on-going project. During the secondment a thorough study of the current approaches regarding sentiment and affective analysis was performed, including their implementation using Natural Language Processing (NLP) and machine learning approaches. Additionally, methods for identifying behavioral patterns where examined as well as methods for the identification of sexual predators. This process was very helpful towards the secondee to familiarize himself with research methodologies as well as the tools and algorithms towards implementing the project.

In this section we briefly describe the work performed towards behavioral pattern identification. Focusing on the data collection, processing and preparation, detailing the experimental procedure and presenting the initial outcomes. Finally, we discuss future steps to be taken towards the completion of the research effort.

8.1. Dataset Collection and Pre-processing

Our project aims at the identification of sexual harassment predators through their online chat conversations with possible victims. Towards this end we needed a dataset of chat conversations coming from real predators.

Perverted Justice is an American NGO (Non-Governmental Organization) whose purpose according to their statement is to “*create a **chilling effect** in regional chat rooms and other easy targets of opportunity online such as social networking websites… **to poison** the well of these rooms and places by covering enough of them that even if you’re looking for underage females, an extra bit of paranoia will cross your mind*”[1]. In other words, the purpose is to awaken parents and adolescents of the importance of identifying that not always the person who speaks on the internet is the one he/she claims to be.

Perverted Justice site aims to prevent the above statements by turning the website into a conviction machine to raise awareness and help with this way the parents. A full list of current convictions (by the time this report is written is reaches ***622 convicted cases***) is available here[2]. For any conviction case the organization provides the entire conversation the predator had with his victim through its website. This conversation includes mail exchanges, online chats, text message chats and more, in a single timed listing including also commentary from the NGO representatives.

We consider the Perverted Justice dataset to be a good match for the goals of our project since it provides actual ground truth of predator behavior in his conversations with the victim. A limitation of the dataset is that all victims’ exchanged dialogs cannot be considered valid since are victims are impersonated by volunteers and are not valid teenagers. This does not stop the purposed methodology to consider these dialogs since it is an extra indication even though these dialogs are not entirely valid. Additionally, this dataset does not have non-harassment conversations that can be

[1] **Information about Perverted-Justice.com:** http://www.perverted-justice.com/index.php?pg=faq
[2] **Perverted Justice Convicted Cases:** http://www.perverted-justice.com/?con=full

used to differentiate the identified patters. Addressing these limitations is left as future work, where we plan to validate our approach with additional online conversations.

Since the entire Perverted Justice dataset is not available for download significant effort was taken to retrieve the data directly from the site. Via the use of ***curl*** command, we managed to extract all 622 dialogs directly from site. Out of the 622 reported cases only 581 where containing data. The rest of the cases led to dead links and not well-maintained data. Thus, our final dataset consists of these 581 convicted sexual predator cases.

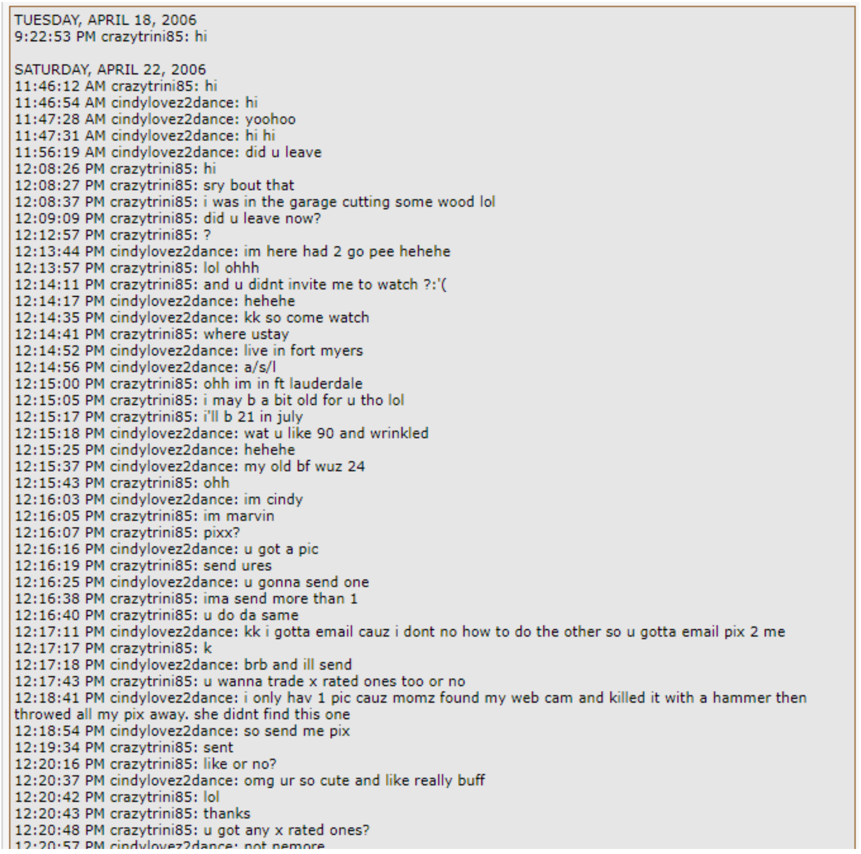
Various difficulties arose regarding this dataset since there is no standard dialog format, i.e. date format is not the same in all dialogs, or the name of the victim/predator is not always in the beginning. To overcome these difficulties a pre-processing step was introduced to standardize the data into a common format that will allow and assist the automated analysis. The following paragraphs describe this pre-processing procedure.

8.1.1.1. Data Pre-processing

Error! Reference source not found.4 presents part of a sample dialog as this is available from the Perverted Justice web page. Beside retrieving the chat information from the specified html div in the web page source code the following processing steps were also taken to annotate each of the chat messages:

1. **Datetime Annotation**: Identify the date and time entries. As it can be seen in **Error! Reference source not found.**4 while chat messages are associated with a timestamp, date information is globally defined for a single day. This was true for a large number of the conversations. In other cases, both time and date were included in the chat message. In the first prepressing step HTML manipulation was performed to associate each chat message with the time and date of its event.
2. **Conversation session identification**: In the data log continuous conversations were separated by a single space in the HTML (See **Error! Reference source not found.**4 Red box). During the analysis process we identify these continuous conversations as a different communication session between the predator and the victim. New sessions mainly arise when predator and victim change medium of communication or when it is observed a time difference in their exchanged posts. In the data extraction process every chat message was assigned the specific session id it was belong to.
3. **Multiline chat message handling**: In some cases, a chat message spanned over a multiple number of lines (See **Error! Reference source not found.** Blue box) without any specified HTML elements denoting this behavior. Special care was taken during the pre-processing phase to identify and combine these multiline chat message into a single message with proper annotation.
4. **Predator/Victim identification**: The Perverted Justice dialog does not explicitly state which is the predator and which is the victim in each chat message, instead it reports that information in the dataset description. Additionally, each dialog contains conversation that happen in different media. It is a common practice for the predator to start the conversation into one medium (i.e. an online chat or forum) and then continue the conversation to a different medium (i.e. email, SMS). The issue that arises in this case is that the nicknames of the conversation parties may change when moving from one medium to the other. To

overcome these limitations extra processing was performed to the dataset to add an indication whether the talking party is the predator (P) or the victim (V) of the conversation.

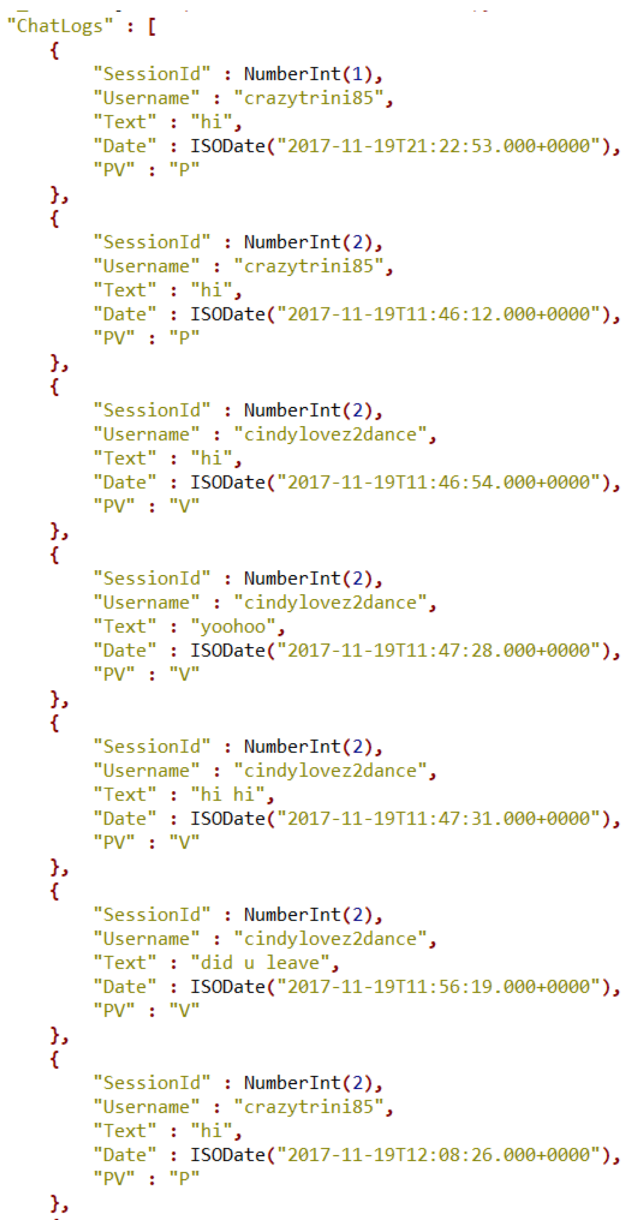


```
TUESDAY, APRIL 18, 2006
9:22:53 PM crazytrini85: hi

SATURDAY, APRIL 22, 2006
11:46:12 AM crazytrini85: hi
11:46:54 AM cindylovez2dance: hi
11:47:28 AM cindylovez2dance: yoohoo
11:47:31 AM cindylovez2dance: hi hi
11:56:19 AM cindylovez2dance: did u leave
12:08:26 PM crazytrini85: hi
12:08:27 PM crazytrini85: sry bout that
12:08:37 PM crazytrini85: i was in the garage cutting some wood lol
12:09:09 PM crazytrini85: did u leave now?
12:12:57 PM crazytrini85: ?
12:13:44 PM cindylovez2dance: im here had 2 go pee hehehe
12:13:57 PM crazytrini85: lol ohhh
12:14:11 PM crazytrini85: and u didnt invite me to watch ?:'(
12:14:17 PM cindylovez2dance: hehehe
12:14:35 PM cindylovez2dance: kk so come watch
12:14:41 PM crazytrini85: where ustay
12:14:52 PM cindylovez2dance: live in fort myers
12:14:56 PM cindylovez2dance: a/s/l
12:15:00 PM crazytrini85: ohh im in ft lauderdale
12:15:05 PM crazytrini85: i may b a bit old for u tho lol
12:15:17 PM crazytrini85: i'll b 21 in july
12:15:18 PM cindylovez2dance: wat u like 90 and wrinkled
12:15:25 PM cindylovez2dance: hehehe
12:15:37 PM cindylovez2dance: my old bf wuz 24
12:15:43 PM crazytrini85: ohh
12:16:03 PM cindylovez2dance: im cindy
12:16:05 PM crazytrini85: im marvin
12:16:07 PM crazytrini85: pixx?
12:16:16 PM cindylovez2dance: u got a pic
12:16:19 PM crazytrini85: send ures
12:16:25 PM cindylovez2dance: u gonna send one
12:16:38 PM crazytrini85: ima send more than 1
12:16:40 PM crazytrini85: u do da same
12:17:11 PM cindylovez2dance: kk i gotta email cauz i dont no how to do the other so u gotta email pix 2 me
12:17:17 PM crazytrini85: k
12:17:18 PM cindylovez2dance: brb and ill send
12:17:43 PM crazytrini85: u wanna trade x rated ones too or no
12:18:41 PM cindylovez2dance: i only hav 1 pic cauz momz found my web cam and killed it with a hammer then
throwed all my pix away. she didnt find this one
12:18:54 PM cindylovez2dance: so send me pix
12:19:34 PM crazytrini85: sent
12:20:16 PM crazytrini85: like or no?
12:20:37 PM cindylovez2dance: omg ur so cute and like really buff
12:20:42 PM crazytrini85: lol
12:20:43 PM crazytrini85: thanks
12:20:48 PM crazytrini85: u got any x rated ones?
12:20:57 PM cindylovez2dance: not nemore
```

Figure 14. Sample exchanged Dialog for case: CrazyTrini85

After the pre-processing steps listed above the annotated dataset is stored into a MongoDB using the following format (15 shows a sample of the annotated data):

a. **CaseId** denoting the whole dialog between the victim and the predator
b. **SessionId** is an index that separates a session from another in terms of changed medium of communication and/or specific time interval
c. **Username:** denoting the Predator/Victim Name
d. **DateTime** which is either full (containing both date and time) or it is just time
e. **PV** is the indication for predator or victim [P, V]



```
"ChatLogs" : [
    {
        "SessionId" : NumberInt(1),
        "Username" : "crazytrini85",
        "Text" : "hi",
        "Date" : ISODate("2017-11-19T21:22:53.000+0000"),
        "PV" : "P"
    },
    {
        "SessionId" : NumberInt(2),
        "Username" : "crazytrini85",
        "Text" : "hi",
        "Date" : ISODate("2017-11-19T11:46:12.000+0000"),
        "PV" : "P"
    },
    {
        "SessionId" : NumberInt(2),
        "Username" : "cindylovez2dance",
        "Text" : "hi",
        "Date" : ISODate("2017-11-19T11:46:54.000+0000"),
        "PV" : "V"
    },
    {
        "SessionId" : NumberInt(2),
        "Username" : "cindylovez2dance",
        "Text" : "yoohoo",
        "Date" : ISODate("2017-11-19T11:47:28.000+0000"),
        "PV" : "V"
    },
    {
        "SessionId" : NumberInt(2),
        "Username" : "cindylovez2dance",
        "Text" : "hi hi",
        "Date" : ISODate("2017-11-19T11:47:31.000+0000"),
        "PV" : "V"
    },
    {
        "SessionId" : NumberInt(2),
        "Username" : "cindylovez2dance",
        "Text" : "did u leave",
        "Date" : ISODate("2017-11-19T11:56:19.000+0000"),
        "PV" : "V"
    },
    {
        "SessionId" : NumberInt(2),
        "Username" : "crazytrini85",
        "Text" : "hi",
        "Date" : ISODate("2017-11-19T12:08:26.000+0000"),
        "PV" : "P"
    },
```

Figure 15. Sample of the annotated dataset for crazytrini85 dialog as stored in our database

8.1.1.2. Sentiment and Affect Annotation

After creating an annotated version of the collected dialog dataset in a common format we proceed with analysing the conversation text to identify sentiment and affects showed by the participants. To do so we first cleanse the data to remove stop words and punctuation marks. Additionally, we use the WordNet Lemmatizer to lemmatize each word in the dataset. After these processes we proceed with the following steps for each chat line:

- Use of **python NLTK and textblob** to include a sentiment score for each sentence in the dataset.
    - **Sentiment score** should include sentiment polarity (scale [-1,1]) showing how negative or positive the phrase.
    - **Sentiment subjectivity** (scale [-1,1]) showing how objective or subjective the sentence is.

- Use of **NRC word-emotion association lexicon**[3] to identify the affect (emotion) of the phrases in the dataset. Emotion consists of anger, anticipation, disgust, fear, joy, sadness, surprise and trust. We annotate each chat line with the most dominant affect returned by the lexicon.
- Identify the topics of phrases in each exchanged post using **genism library**[4], via the use of Latent Dirichlet Allocation[5] for Topic Modelling.

8.2. Initial Data Analysis

The previous section described how we managed to retrieve the Perverted Justice convicted sexual harassment cases conversation and the method followed to process and annotate the data. In this section we provide some initial results on the analysis performed. Recall that our final goal is to provide a method for early identification of sexual predators in online chats. Towards this goal we aim at identifying specific behavioral patterns of the predator in chat conversations.

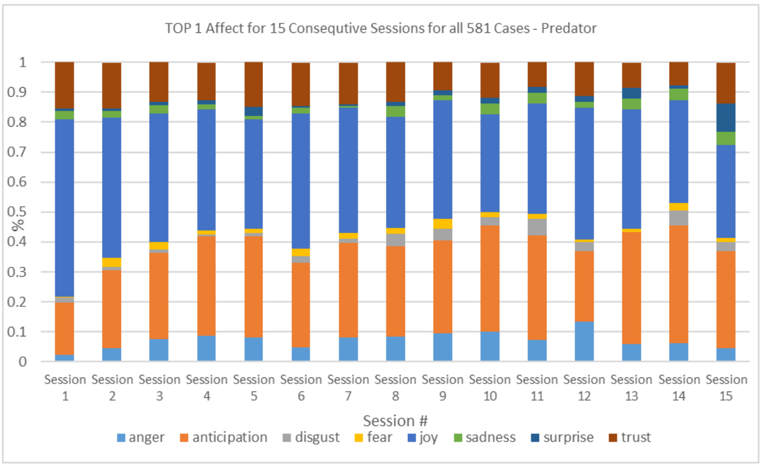
In this section we focus on the identified sessions as defined in the previous chapter. For each of these sessions we first summarize the individual chat lines sentiment and affect values into the dominant affect for the session. To do so we first create a normalized affect table for each chat line. NRC library returns the number of words in each document that can be associated with each affect. To get the line-wide normalized table we add divide these numbers with the total number of words in the chat line. To get the overall affect score for the session we then sum the affect score over all chat lines of the session and normalize with the total number of lines. The above normalizations result to an ordered list of affects for the session, for both participants.

Using the above list when then proceed to examine the predator affect transitions for each session. **Error! Reference source not found.** presents the top 3 predator affects for the first 15 sessions of all collected conversations. As it can be seen, three are the most prevalent affects showed by the predator, Joy, Anticipation and Trust. It is our speculation that the goal of the predator in the initial steps of the conversation is to come close to the victim through a joyful conversation and gradually build trust to be able to “ask for more”. Anticipation is experience since the predator constantly request for the response of the victim to progress the conversation as fast as possible. Other affects like anger and disgust also appear later in the discussion (in subsequent sessions). Manually observing some of the conversations these affects depict the effort of the predator to find some common ground with its victim by exploiting some common “enemy” like parents, city authorities etc.

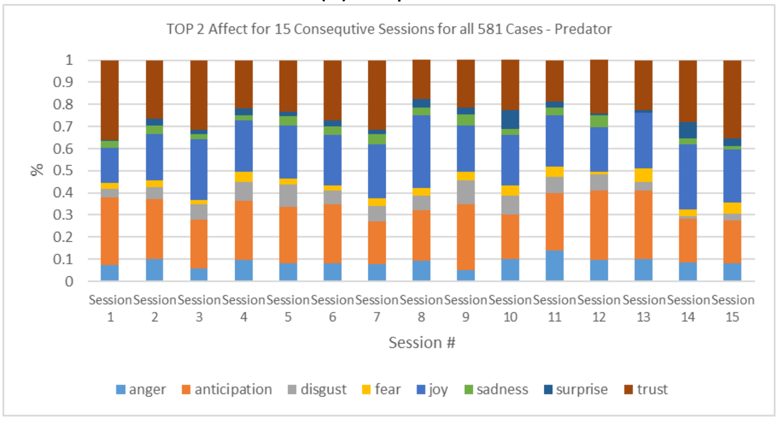
The next question we examined was how these affect transition from one session to the other for the same predator. Figure 16 tries to make a fist evaluation over these transitions. The figure shows the movement of predators from one affect to the other as the conversation evolves to the next session. Sessions evolve from left to right in the figure. Each column lists all the affects observed in that session along with the number of predator sessions where that affect was the top-1 affect. Transition lines show the percentage of predators that changed from one affect to the other in the subsequent session. To avoid confusion, we only depict in the Figure the transitions from the three

[3] **NRC word-emotion association lexicon:** http://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm
[4] **Gensim library:** https://radimrehurek.com/gensim/
[5] **Latent Dirichlet allocation:** https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation

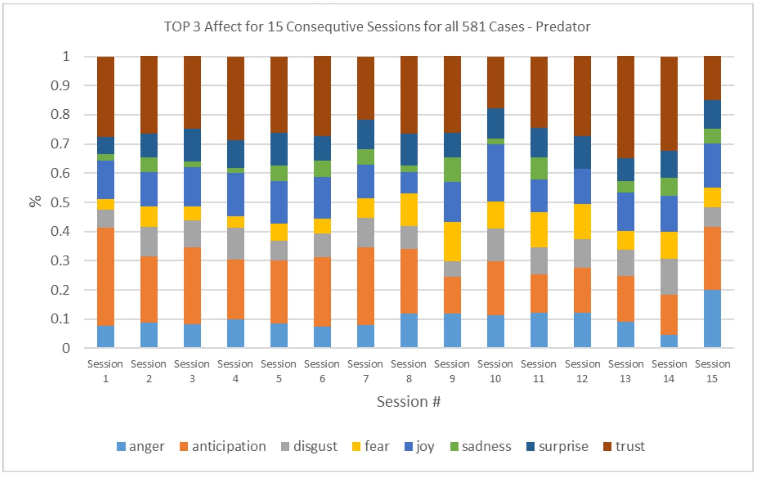
top affects (Joy, Anticipation and Trust). On the lower part it lists the number of conversations that proceeded to that session (polygonal shapes).



(a) Top 1 affect



(b) Top 2 affect



(c) Top 3 affect

Figure 16. Top 3 affects per session for all collected conversations

The transitions from Figure 17 show that predators mostly interchange their behavior in the three dominant affects. In both Joy and Anticipation, the largest percentage of predators continues to show the same affect in subsequent sessions. Smaller but significant percentages of predators move from either the two affects to the other or Trust. These transitions are inline with our previous speculations that predators aim at building a joyful trusted relationship with the victim while are also egger to progress the conversation. Predators that begin the conversation with trust building efforts (i.e. having Trust as the top-1 affect) quickly move to show Joy and Anticipation in the subsequent sessions.

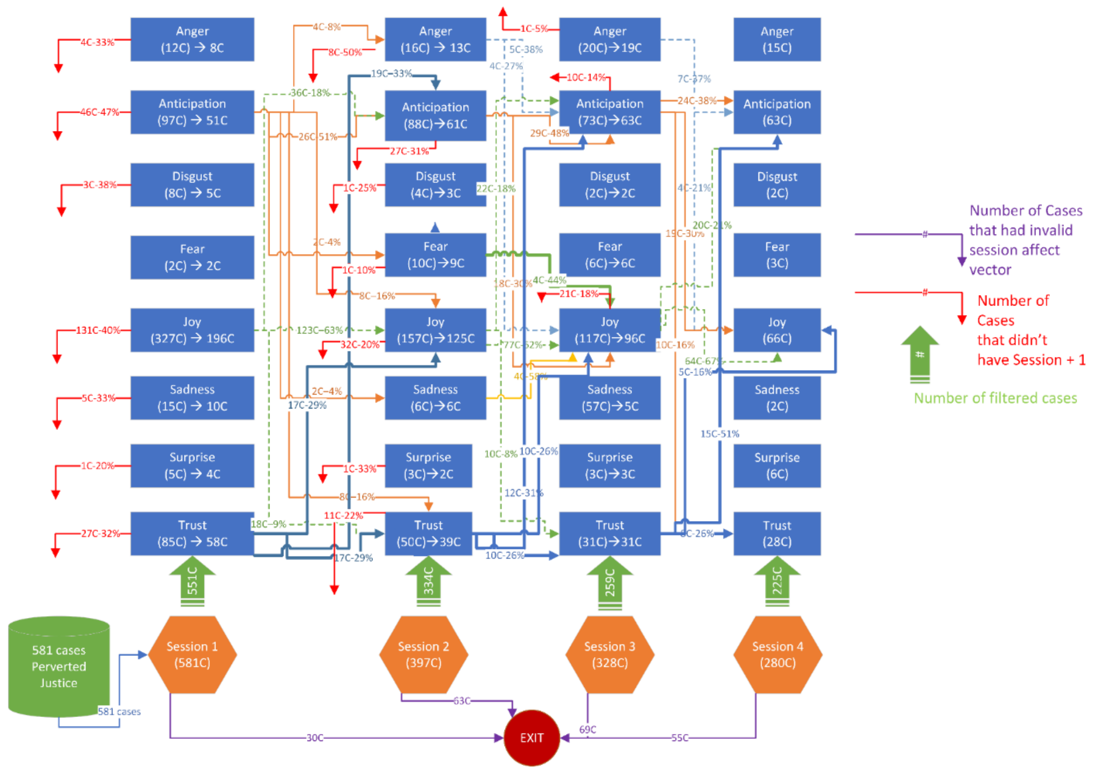


Figure 17. Affect transition diagram

8.3. Conclusion

Secondment 37 targeted the online and real-time classification of Cyberbullying and Online Sexual Harassment observed in online conversations such as online chat and comment discussions. The initial part of the secondment was spend in familiarizing the secondee with the research area and the research methodology to be followed. A significant part of the secondment was spend into selecting, collecting and annotating the data to be used for such a research endeavor. This process is explained in the above sections and in more detail in the internal report delivered by the secondee. The data are already available to the project partners and is expected to be made openly available (with proper documentation) in the forthcoming months.

The initial analysis performed show promising results for the research target of the project. Predators seem to use three main affects in their soliciting efforts, namely Joy, Anticipation and Trust, and mainly interchange between the three through the conversation timeline.

Based on these initial results, further analysis is planned to model and correctly predict these transitions as well as comparing them with benign conversations. Additionally, the examination of time-defined sessions, i.e. separating conversation based on the time gap between them, instead of the current conversation based sessions is planned. Also, the methods and data used for analysis is planned to be presented as an online demonstration with the goal of their integration into the browser extensions and plugins to be provided by the ENCASE project.

9. Crowdsourcing Large Training Datasets (Ongoing work)

The overall aim of this task is to use crowdsourcing to accurately annotate a large dataset of tweets according to the inappropriate speech they contain. Such a dataset, can facilitate the engineering of advanced techniques capable of protecting users from malicious behaviors in OSN.

9.1. Preliminary Methodology

The first step of the process was to collect a random set of tweets. To do so, we utilized the Twitter Stream API and we collected all the tweets provided by the API (1% of the entire traffic) during the period of 30th March 2017 – 9th April 2017 consisting of 32 million tweets. To handle such a big dataset, we decided to use a front-loaded approach. In particular, we decided to apply a set of data augmentation techniques as a preprocessing step, and then store the augmented tweets into a database (elasticsearch in our case). Using this approach, the problem of selecting any subset was reduced to a single query on the database. The overall architecture is depicted in the following figure.

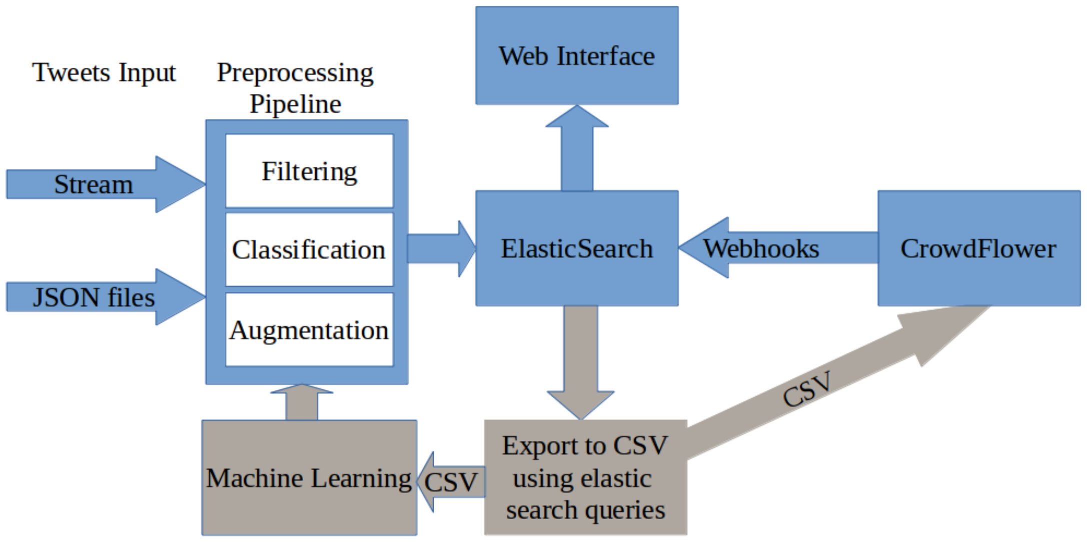
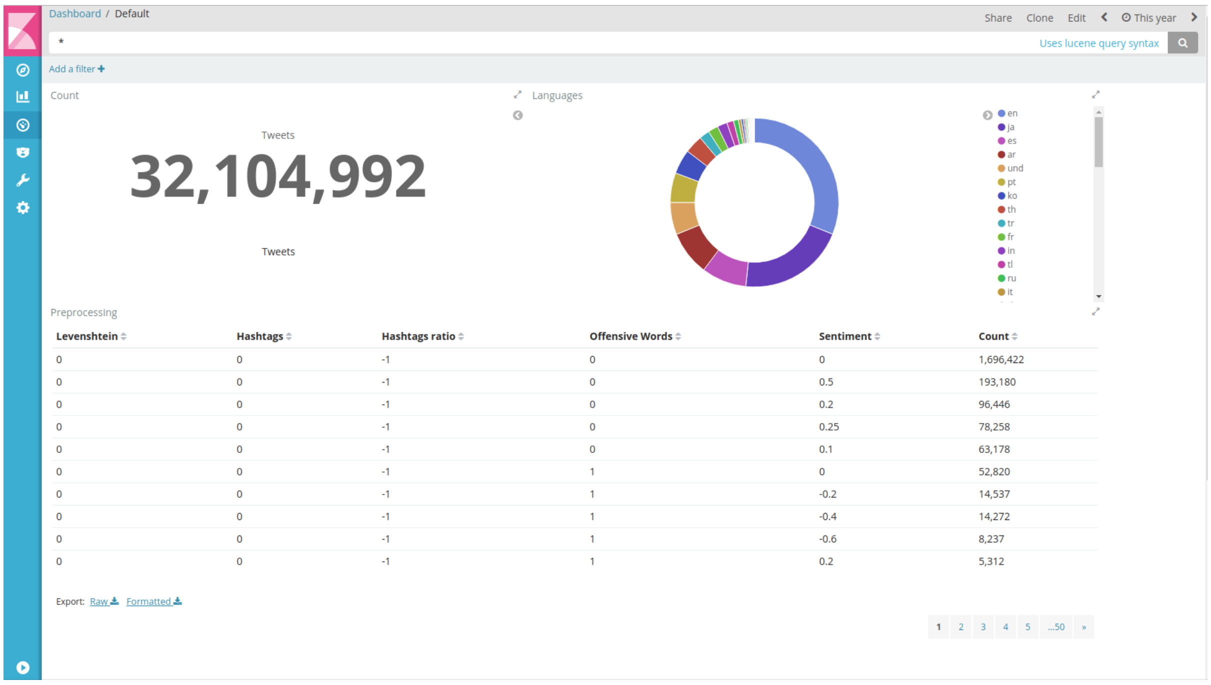


Figure 18. Overall Architecture

As mentioned above, the proposed architecture reduces the problem of selecting any arbitrary subset of the entire dataset to a single database query. Bellow, we provide an example of such a query implemented in Python; it selects 1000 random tweets in english that are not retweets, they contain up to 4 hashtags, and their length is between 100 and 160 characters.

```
body={

  "from" : 0,

  "size" : 1000, #Number of entries to be exported

  "query": {
      "function_score": {

        "random_score": { "seed": 1 },

        "query" : {

          "bool":{

            "filter": [

              { "term":  { "lang": "en" }},

              { "term":  { "retweet": 0 }},

              { "range" : { "preprocessing.hashtags" : { "gte" : 0, "lte" : 4 } } },

              { "range" : { "preprocessing.length" : { "gte" : 100, "lte" : 160 } } }

            ]

          }

        },

        "boost_mode" : "replace"

      }

    }

})
```

Finally, a Kibana dashboard was also implemented (Figure 19). Using a web interface, one can easily extract some aggregated results about the stored dataset. Furthermore, custom queries can be sent to the database allowing for the in depth exploration and the exploitation of our dataset.



Levenshtein	Hashtags	Hashtags ratio	Offensive Words	Sentiment	Count
0	0	-1	0	0	1,696,422
0	0	-1	0	0.5	193,180
0	0	-1	0	0.2	96,446
0	0	-1	0	0.25	78,258
0	0	-1	0	0.1	63,178
0	0	-1	1	0	52,820
0	0	-1	1	-0.2	14,537
0	0	-1	1	-0.4	14,272
0	0	-1	1	-0.6	8,237
0	0	-1	1	0.2	5,312

Figure 19. Kibana Dashboard

9.1.1.1. Preprocessing

A necessary phase for the aforementioned architecture is the preprocessing. During preprocessing, tweets are enriched with some extra fields that they can be later used for selecting subsets of the original dataset with different characteristics. The preprocessing phase yields the following augmentation fields:

- retweet: 0 if a tweet is not a retweet, 1 otherwise.
- length: The length of the tweet
- urls: The number of URLs in the tweet
- hashtags: The number of hashtags in the tweet
- hashtags_ratio: Unique hashtags / All hashtags
- mentions: The number of mentions in the tweet
- mentions_ratio: Unique mentions / All mentions
- reserved: The number of reserved terms in the tweet (RT, DM, OH, HT, etc.)
- emojis: The number of emojis in the tweet
- emojis_ratio: Unique emojis / All emojis
- smileys: The number of smileys in the tweet
- smileys_ratio: Unique smileys / All smileys
- numbers: The number of arithmetic values in the tweet
- polarity: The polarity of the sentiment analysis applied on the tweet in the range [-1,1]
- subjectivity: The subjectivity of the sentiment analysis applied on the tweet in the range [0,1], where 0 is very objective and 1 is very subjective
- offensiveness_count: Number of stemmed offensive words found in the tweet.
- levenshtein: If the tweet is a retweet, this is the Levenshtein distance between the original tweet and the retweet. Results are in the range [0,1], where 0 is completely different and 1 is identical

Clearly, all but the last four fields derive directly from the tweet’s content. Polarity and subjectivity were created using the TextBlob Python library. TextBlob produces a polarity output in the range of [-1.0, 1.0] and a subjectivity score in the range of [0,1]. Furthermore, for creating the offensiveness_count field, two dictionaries containing hate-base and offensive words were utilized. All words in a tweet were stemmed and lower-cased, and matched against the two dictionaries; matching entries were counted and the final score was added to the augmented tweet.

Finally, one of the tasks of this preprocessing phase was to facilitate spam detection. More precisely, it aims in identifying entries that undoubtedly correspond to Spam and thus their content can be considered as noise (e.g., only contain URLs and not actual content, too many hashtags, etc.). Such entries will be removed in an attempt to send to annotators only tweets that can actually contribute in the learning process. Even though we are aware of a number of advance techniques for tweet spam detections [1-5], we decided to use more naive approaches. This is due to the huge data set, limiting us to use techniques that solely rely on tweets' content analysis and not on any other meta-data (e.g., users' behavior) extracted from Twitter. Towards this end, the Levenshtein distance between the original tweet and the retweet is calculated since retweets very similar or even identical to the original tweet is an indication of spamming.

9.1.1.2. Methodology

Having preprocessed and loaded all the tweets into the database, the next task was to annotate a subset of them using the Crowdflower (CF) service. The first step for any annotation is to define the labels to be used; in our case, that is to identify the types of inappropriate speech. This task, however, is rather complicated due to a general confusion following the use of different categories of inappropriate speech.

In general, there is a high amount of inappropriate labels used in the literature of this subject. Hate speech detection, cyberbullying identification, the recognition of abusive, or offensive language, as well as detection of racism and sexism are some of these categories. Yet, to this day, no empirical study exists to try untangle the hierarchy and relationships between the various labels that are being used. Hence, in most works the labels either overlap or are being used interchangeably. A major contribution of this work is to attempt solving this confusion. To achieve this, we selected the most popular inappropriate labels and conducted a series of CrowdFlower experiments. The final target is to merge the most related labels and come up with the set of most important ones so that we capture the essence of all the others. We decided to use the following categories: ‘Offensive’, ‘Abusive’, ‘Hateful’, ‘Aggressive’, ‘Cyber-bullying’.

The methodology we follow is separated into two consequent levels of annotations, one small-scale and one large-scale. On the former we concentrate on selecting the most appropriate set of labels based on a thorough analysis, as described below, while on the latter we apply the knowledge we gained from the first level and annotate the large-scale dataset.

For the primary level, our focus is to solve the previously mentioned obstacle and decide upon the most representative labels for our task. In order to achieve this, we conducted a series of annotation rounds on a small dataset, where the target is to analyse the occurrences and correlations of all inappropriate labels. We begin with a first round that includes 300 tweets and 5 judgments per tweet to collect annotations and assess if the plurality of labels is confusing, how the spamming annotation works, etc. Afterwards, we continue to a second round where we decrease the dataset to keep only tweets previously marked as inappropriate and increase the number of judgments to be at least 10 per tweet. These two rounds gave us valuable insight to eventually narrow down the list of labels into the most representative and important ones. Finally, we run a third round with the initial 300 tweets, where we confirm that our selection of labels is accurate and improves the results, before we run the final and large-scale annotation task. This round also works as a simulation of the next level, in order to test any configurations needed and so on.

Based on the decisions drawn upon the previous results, we then plan the second level of the annotations, where the large-scale dataset will be created. The configurations for this round will be kept the same with the last one, since that was one of its main purposes.

9.1.1.3. Annotation Dataset

In order to accurately capture the correlation of the categories defined above, along with the annotators’ interpretation, we performed a preliminary small-scale annotation round consisting of 300 tweets.

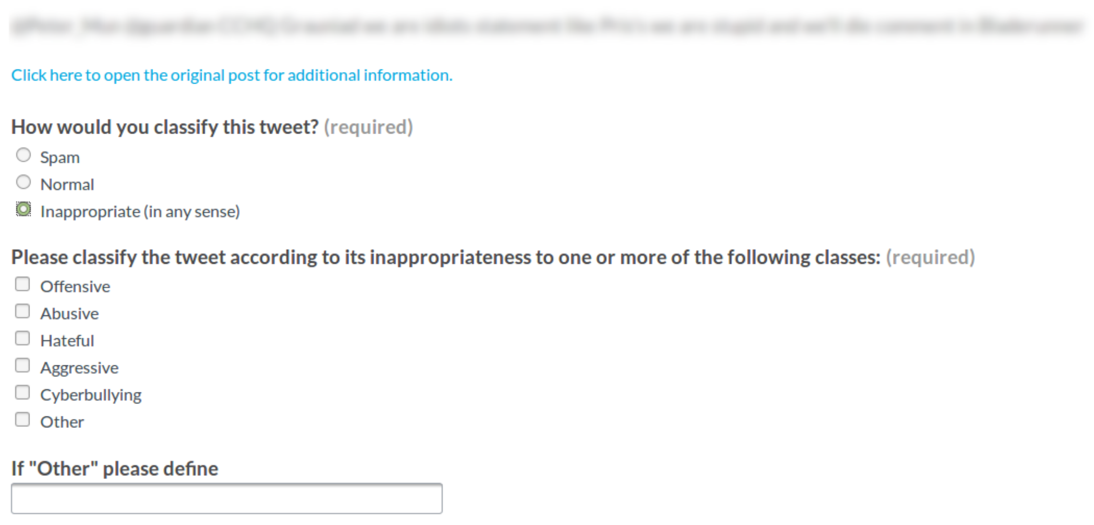
As mentioned above, the dataset used is consisted of 300 english tweets. These tweets were selected from two subsets: a) the first 200 tweets which were selected randomly, and b) 100 tweets sampled to be dense in inappropriate speech. For the former subset we applied three simple filtering criteria. Firstly, we only kept tweets with at most 4 hashtags and length of at least 80 characters. These filtering criteria were applied in an attempt to remove some undoubtedly spam entries. Furthermore, native retweets (i.e., tweets that contain the 'retweeted_status' field), were also filtered out.

The latter subset inherits all characteristics of the subset described above. However, in addition to the filtering applied above, two more filtering criteria were applied. Using two of the injected variables, polarity and offensive words, tweets in this dataset were filtered so that they have a polarity in the range [-1, -0.7] and at least 1 offensive words. Clearly, for creating the aforementioned subsets no filtration or processing on user related fields was applied. Thus, more than one tweet from the same user might appear in the dataset.

9.2.Annotation Rounds Descriptions and Preliminary Results

9.2.1.1. First Annotation Round

During this round, annotators were asked to first classify tweets into three general categories, that is, ‘Spam’, ‘Normal’, and ‘Inappropriate’. In the case that ‘Inappropriate’ was selected, then the five aforementioned inappropriate speech categories were presented so that the users can define the type of inappropriate speech exhibited by the tweet. Furthermore, they had the option to define a new subcategory utilizing the ‘Other’ option and a text box. Finally, the participants were able to select multiple subcategories if an inappropriate tweet falled under multiple categories. A screenshot of the page presented to the annotators is depicted below.



[redacted]

Click here to open the original post for additional information.

How would you classify this tweet? (required)
- Spam
- Normal
- [✓ Inappropriate (in any sense)]

Please classify the tweet according to its inappropriateness to one or more of the following classes: (required)
- Offensive
- Abusive
- Hateful
- Aggressive
- Cyberbullying
- Other

If "Other" please define

Figure 20. Annotation page

In order to perform the annotations, annotators were provided with definitions for each label. All these definitions were constructed based on all the definitions we found on the related literature, as cited on each category.

For each of the inappropriate labels we have the following definitions:

- **Offensive Language**: Profanity, strongly impolite, rude or vulgar language expressed with fighting or hurtful words in order to insult a targeted individual or group. [6,7,8,9]
- **Abusive Language**: Any strongly impolite, rude or hurtful language using profanity, that can show a debasement of someone or something, or show intense emotion. [9,10,11,12,13]
- **Hate Speech**: Language used to express hatred towards a targeted individual or group, or is intended to be derogatory, to humiliate, or to insult the members of the group, on the basis of attributes such as race, religion, ethnic origin, sexual orientation, disability, or gender. [13,14,15,16,17,18,19]
- **Aggressive Behavior**: Overt, angry and often violent social interaction delivered via electronic means, with the intention of inflicting damage or other unpleasantness upon another individual or group of people, who perceive such acts as derogatory, harmful, or unwanted. [13,20,21]
- **Cyber-bullying Behavior**: It's the use of force, threat, or coercion to abuse, embarrass, intimidate, or aggressively dominate others, using electronic forms of contact. It typically denotes repeated and hostile behavior performed by a group or an individual. [13,20,21,22,23,24]

Moreover, we need to define ‘Normal’ and ‘Spam’ as they are also part of the annotations. We define spam as: “posts that are consisted of related or unrelated advertising / marketing, selling products of adult nature, linking to malicious websites, phishing attempts and other kinds of unwanted information, usually executed repeatedly”. Finally, ‘Normal’ refers to all the legitimate tweets that do not fall on any of the prior categories.

9.2.1. Results

The dataset described above was sent to Crowdflower for annotation, asking for five judgments for each tweet. In this section we present the results acquired by the annotation process.

First we present the agreement among annotators on the three general categories, ‘Normal’, ‘Spam’, and ‘Inappropriate’. To do so, we calculate each tweet’s majority label and its corresponding score. For example, if 4 out of 5 annotators judged a tweet as ‘Normal’, the majority label will be 'Normal' and the score will be 80%. Based on the majority scores, we defined three agreement groups:

- Overwhelming Majority: score >= 80%
- Strong Majority: 50% <= score < 80%
- Simple Majority: score < 50%

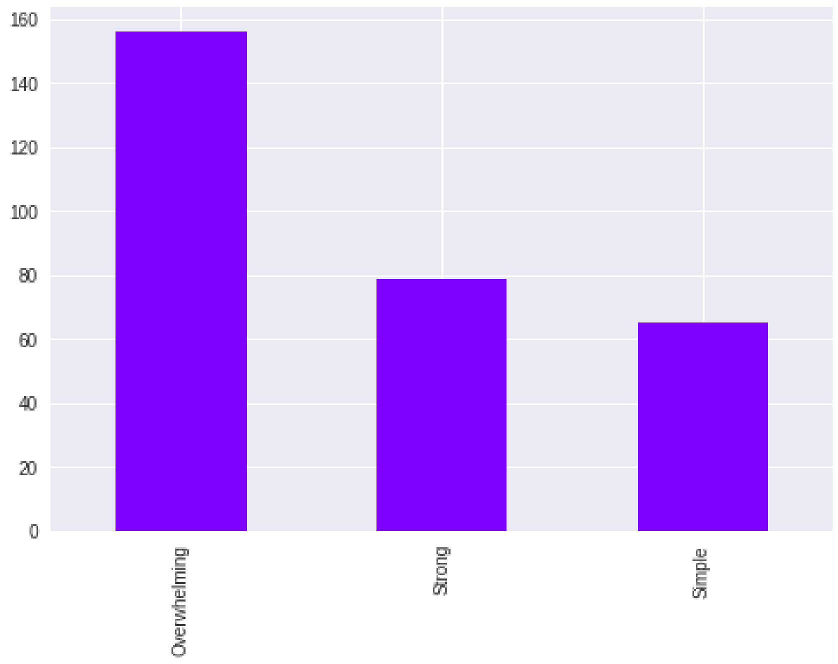


Figure 21. Distribution among agreement groups

The figure 21 above depicts the distribution of tweets among each agreement group. They sum up to 272, since 28 didn’t exhibit any majority label, i.e., two categories got the same number of annotations. Furthermore, we clearly observe that the high majority of annotators agreed on their judgments with consistency over 80%.

As already mentioned, an important goal of this work is to give some insights on how regular users understand and perceive the different categories of speech exhibited online. To do so, we conducted some analysis on the annotations, where we measure the co-occurrences of the various labels, we calculate their correlations and measure their similarities. We begin with the analysis of the co-occurrences, where we flatten out the Inappropriate general category and we calculated the cooccurrences of labels for each one of the three agreement groups defined above (Tables 13-15). Rows represent the majority label and columns the labels coexisted with the majority label - this is the reason that the tables are not symmetric. Finally, the diagonal represents the number a label was the majority label.

Coexisting Labels							
Majority Label	**Abusive**	**Aggressive**	**Cyberbullying**	**Hateful**	**Normal**	**Offensive**	**Spam**
Abusive	3	3	0	0	0	0	0
Aggressive	0	1	0	0	0	0	0
Cyberbullying	0	0	0	0	0	0	0
Hateful	1	0	0	2	0	0	0
Normal	3	1	2	3	129	15	26

Offensive	0	0	0	0	0	0	0
Spam	0	0	0	0	10	1	21

Table 13. Cooccurences of labels for tweets with overwhelming agreement

Coexisting Labels / Majority Label	Abusive	Aggressive	Cyberbullying	Hateful	Normal	Offensive	Spam
Abusive	9	6	1	2	2	6	0
Aggressive	2	3	0	1	0	3	0
Cyberbullying	0	0	0	0	0	0	0
Hateful	1	0	0	2	0	2	0
Normal	5	6	1	8	36	12	22
Offensive	8	2	1	4	5	11	0
Spam	1	0	0	1	18	2	18

Table 14. Cooccurences of labels for tweets with strong agreement

Coexisting Labels / Majority Label	Abusive	Aggressive	Cyberbullying	Hateful	Normal	Offensive	Spam
Abusive	8	7	1	6	4	6	0
Aggressive	3	4	0	4	2	4	0
Cyberbullying	0	0	0	0	0	0	0
Hateful	6	6	1	9	6	7	2
Normal	4	3	0	5	6	6	2
Offensive	8	3	3	6	5	8	0
Spam	1	0	1	1	2	1	2

Table 15. Cooccurrences of labels for tweets with simple agreement

The results show that users seem to be very confused about selecting a label, resulting in low levels of agreement for most of the inappropriate tweets. As expected, this is more apparent when tweets are harder to categorize, i.e., simple agreement group. Furthermore, we can see that in all agreement group (i.e., overwhelming, strong, simple), ‘Spam’ regularly coexisted with the ‘Normal’ category. Finally, among the Inappropriate subcategories, the ‘Offensive’ is sometimes confused with ‘Normal’.

Finally, we would like to measure the correlation among the Inappropriate category labels. In this work, we calculate correlations and significance using Pearson, Spearman, and Kendall Tau Correlation Coefficients. Table 16 present these results.

	PCC	p-PCC	SCC	p-SCC	KTCC	p-KTCC
Offensive - Abusive	0.057672	0.4863	0.10946	0.1854	0.095695	0.0844
Offensive - Hateful	-0.064017	0.4395	-0.00829	0.9203	-0.007995	0.8854
Offensive - Aggressive	-0.122807	0.137	-0.124149	0.1327	-0.110099	0.0471
Offensive - Cyberbullying	-0.083271	0.3143	-0.055111	0.5059	-0.049595	0.3711
Abusive - Hateful	-0.096501	0.2433	-0.03605	0.6636	-0.033111	0.5504
Abusive - Aggressive	0.195979	0.017	0.324244	0.0001	0.285359	0
Abusive - Cyberbullying	0.042049	0.6118	0.06507	0.432	0.060229	0.2774
Hateful - Aggressive	-0.042224	0.6104	0.024063	0.7716	0.020994	0.705
Hateful - Cyberbullying	-0.076778	0.3537	-0.080666	0.3298	-0.076305	0.1688
Aggressive - Cyberbullying	-0.142836	0.0833	-0.16974	0.0392	-0.161572	0.0036

Table 16. Correlation Coefficient & P-Value results for each pair of labels

All three correlation coefficient metrics show low correlation between the labels. The only correlation that seems consistently more significant in all cases is between 'Abusive' and 'Aggressive'. Moreover, 'Aggressive' and 'Cyberbullying' seem to be somewhat correlated, but the significance is not consistent. Finally, Kendall Tau also shows a significant relationship between 'Offensive' and 'Aggressive', which does not appear in the other two cases. The rest of the combinations do not exhibit any important correlations.

Finally, the similarities between the labels are measured using Cosine Similarity (Table 17). For each pair of labels, we calculate their similarity vectors, in order to gain some insight on the correspondence of pairs in accordance with their ranking. We see on Table 17 that the values of the similarities are not very high. Nevertheless, the most highly similar pairs are Offensive and Aggressive with Abusive.

	Cosine Similarity
Offensive - Abusive	0.536908
Abusive - Aggressive	0.478639
Offensive - Hateful	0.410749
Offensive - Aggressive	0.348367
Abusive - Hateful	0.320653
Hateful - Aggressive	0.279881
Abusive - Cyberbullying	0.251285

Offensive - Cyberbullying	0.209020
Hateful - Cyberbullying	0.133986
Aggressive - Cyberbullying	0.066667

Table 17. Cosine Similarity results for each pair of labels, sorted by ranking

9.2.2. Second Annotation Round

Results presented above gave some insight on the correlation among inappropriate speech categories. However, our confidence on these results is low, mainly because of the low amount of annotations per tweet, in a small sample. For this particular reason, we decide to proceed in a new annotation round. In this round, only tweets classified as Inappropriate in the previous round will be used (88 tweets) and each tweet will be annotated by at least 10 users.

9.2.2.1. Results

Results acquired by this annotation round are presented in this section. First, in Figure 22, we present the agreement groups (defined earlier). The results clearly show the confusion and the difficulty of users assigning a tweet to an inappropriate speech class. Among the things we notice, we see that the greatest part of the tweets' judgments disagree on their label and there are very few of them that agree with more than 80%. Also, we see that for 13 of the 88 tweets, the participants were divided and could not decide on one label (the result of the previous round was 28 out of 300). Therefore, we suppose almost half of the tweets that did not reach a majority agreement on one label are ‘Normal’ and the rest are ‘Inappropriate’.

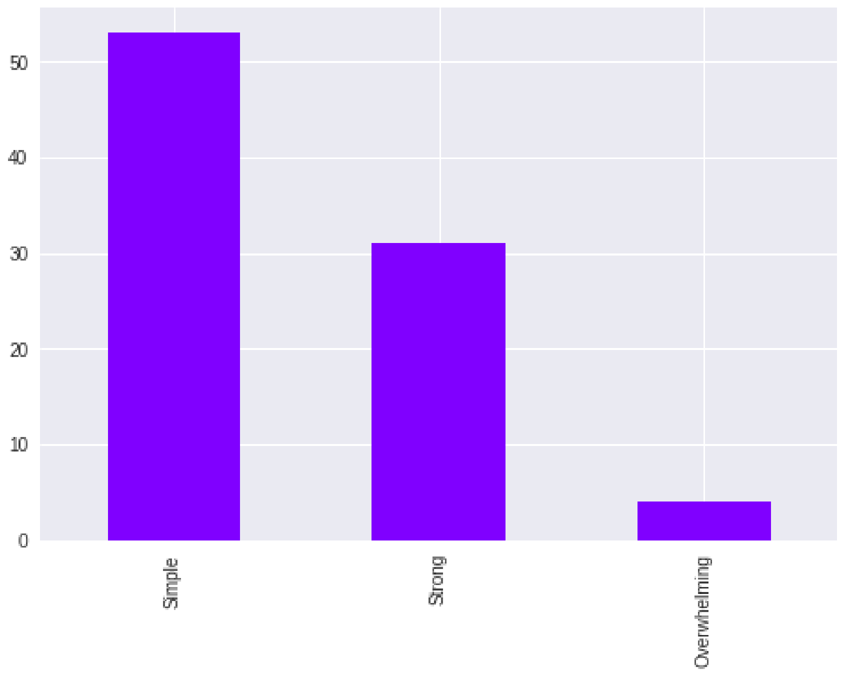


Figure 22. Distribution among agreement groups

Next we present three tables (Tables 18-20) showing the coexistence of labels in the three agreement groups. As in the previous section, rows represent the class that the tweet was assigned to (majority label) and columns show the number other labels coexisted with the majority label[6]. Finally, the diagonal shows the times a label was a majority label.

Coexisting Labels / **Majority Label**	**Abusive**	**Aggressive**	**Cyberbullying**	**Hateful**	**Normal**	**Offensive**	**Spam**
Abusive	0	0	0	0	0	0	0
Aggressive	0	1	0	0	0	1	0
Cyberbullying	0	0	0	0	0	0	0
Hateful	0	2	0	2	0	2	0
Normal	0	0	0	0	0	0	0
Offensive	0	0	1	0	1	1	0
Spam	0	0	0	0	0	0	0

Table 18. Cooccurrences of labels for tweets with overwhelming agreement

Coexisting Labels / **Majority Label**	**Abusive**	**Aggressive**	**Cyberbullying**	**Hateful**	**Normal**	**Offensive**	**Spam**
Abusive	8	6	1	6	3	8	1
Aggressive	4	4	0	3	1	4	0
Cyberbullying	0	0	0	0	0	0	0
Hateful	5	5	1	6	1	5	1
Normal	1	1	1	0	1	1	1
Offensive	9	9	4	10	3	12	2
Spam	0	0	0	0	0	0	0

Table 19. Cooccurrences of labels for tweets with strong agreement

Coexisting Labels / **Majority Label**	**Abusive**	**Aggressive**	**Cyberbullying**	**Hateful**	**Normal**	**Offensive**	**Spam**
Abusive	19	16	10	18	11	19	5
Aggressive	0	0	0	0	0	0	0
Cyberbullying	0	0	0	0	0	0	0

[6]Tweets without a majority label were omitted from the tables.

Hateful	6	5	1	7	3	7	4
Normal	1	1	1	1	1	1	0
Offensive	12	12	5	9	7	12	4
Spam	1	1	0	0	0	1	1

Table 20. Cooccurrences of labels for tweets with simple agreement

Clearly, this lower level analysis more prominently shows the confusion among the different types of inappropriate speech. More particularly, ‘Abusive’ seems to be used a lot of times and is often the majority label, but it's always confused with many of the other labels (especially ‘Offensive’). Moreover, ‘Offensive’ is also very confused with ‘Abusive’ and ‘Aggressive’, and frequently also with ‘Hateful’. ‘Cyberbullying’ still never becomes a majority-label.

When it comes to Correlation Coefficients (Table 21), there are generally low significant correlations between the labels, with some exceptions. ‘Offensive’ and ‘Abusive’ are consistently significantly correlated. Furthermore, in most of the cases, both are also significantly correlated to ‘Cyberbullying’. Spearman and Kendall also show some correlation between ‘Offensive’ and ‘Abusive’ with ‘Aggressive’. ‘Hateful’, never seems to be correlated to the rest of the labels.

	PCC	p-PCC	SCC	p-SCC	KTCC	p-KTCC
Offensive - Abusive	0.322597	0.0022	0.408552	0.0001	0.294481	0
Offensive - Hateful	-0.076287	0.4799	-0.130442	0.2258	-0.095233	0.1889
Offensive - Aggressive	0.056567	0.6006	0.245213	0.0213	0.186104	0.0102
Offensive - Cyberbullying	0.230017	0.0311	0.191246	0.0743	0.157496	0.0298
Abusive - Hateful	0.126504	0.2402	0.118195	0.2727	0.079851	0.2706
Abusive - Aggressive	0.011576	0.9148	0.270948	0.0107	0.199341	0.006
Abusive - Cyberbullying	0.243139	0.0225	0.241344	0.0235	0.202128	0.0053
Hateful - Aggressive	-0.072054	0.5047	-0.039991	0.7114	-0.030565	0.6733
Hateful - Cyberbullying	0.013788	0.8985	0.003095	0.9772	0.003917	0.9569
Aggressive - Cyberbullying	-0.001821	0.9866	0.125888	0.2425	0.10938	0.1313

Table 21. Correlation Coefficient & P-Value results for each pair of labels

Finally, we calculate again the Cosine Similarity between the various pairs of labels (Table 22). We notice that the values of the similarities are much higher than before, while again the most similar pair of labels is Offensive and Abusive, followed, this time, by Abusive - Hateful. We notice in general, on this annotation round, that hateful seems to be more related than it was on the previous round, but the correlation results still don’t indicate a strong correspondence.

	Cosine Similarity
Offensive - Abusive	0.741228
Abusive - Hateful	0.619584
Offensive - Hateful	0.544227
Abusive - Cyberbullying	0.501380
Offensive - Cyberbullying	0.497397
Offensive - Aggressive	0.482113
Abusive - Aggressive	0.451047
Hateful - Aggressive	0.374166
Hateful - Cyberbullying	0.350230
Aggressive - Cyberbullying	0.268009

Table 22. Cosine Similarity results for each pair of labels, sorted by ranking

9.2.2.2. Third Annotation Round

The results of the first two annotation rounds allowed us to draw the following conclusion regarding the use of the five inappropriate speech labels:

- We can form three groups of labels according to their popularity; ‘Abusive’ and ‘Offensive’ are the two most popular labels, ‘Hateful’ and ‘Aggressive’ are somewhat popular, and ‘Cyberbullying’ is rarely used.
  Therefore ‘Cyberbullying’ can be safely eliminated from the list of inappropriate labels, mainly due to the very few times it was selected on both the annotation rounds. This decision, though, is also supported by the very nature of cyberbullying, which according to its definition should be repetitive. Yet, in our case we have no sense of time or repetition, since we work with individual tweets.
- ‘Abusive’, ‘Offensive’ and ‘Aggressive’ seem to be significantly correlated, highly coexisting in the annotations and very similar on the similarity results. ‘Abusive’ is the most popular among the three and the most ‘central’ (i.e. the other two labels are much more related with this than with each other).
- While ‘Hateful’ is frequently coexisting with other labels (indicating a confusion among users in the use of this label), it does not appear to be significantly correlated with any other label. This is also supported by the definition of ‘Hateful’, since we notice a better-defined description of the target groups of this category, compared to the rest.

Given the above conclusions, we decided to proceed to a third round in order to disambiguate the correlation of the ‘Hateful’ label with the other labels. To do so, we first decided to remove ‘Cyberbullying’ (due to point 1). Then, utilizing point 2, we merged ‘Abusive’, ‘Offensive’ and ‘Aggressive’ into a single category. For the reasons explained in point 2, the label of the new category is ‘Abusive’. Finally, we decided to keep ‘Hateful’ separately, as explained in point 3. Thus, in this annotation round, users will be presented with four labels, that is, ‘Abusive’, ‘Hateful’, ‘Normal’, ‘Spam’. The dataset used was the original one containing 300 tweets and we required 5 judgments per tweet.

9.2.2.3. Results

As in the previous rounds, we first present the level of agreement among annotators by forming three agreement groups (Figure 23). Clearly, we can observe a vast improvement since only a few tweets assigned to the simple agreement group, something that designates a simpler task for the annotators.

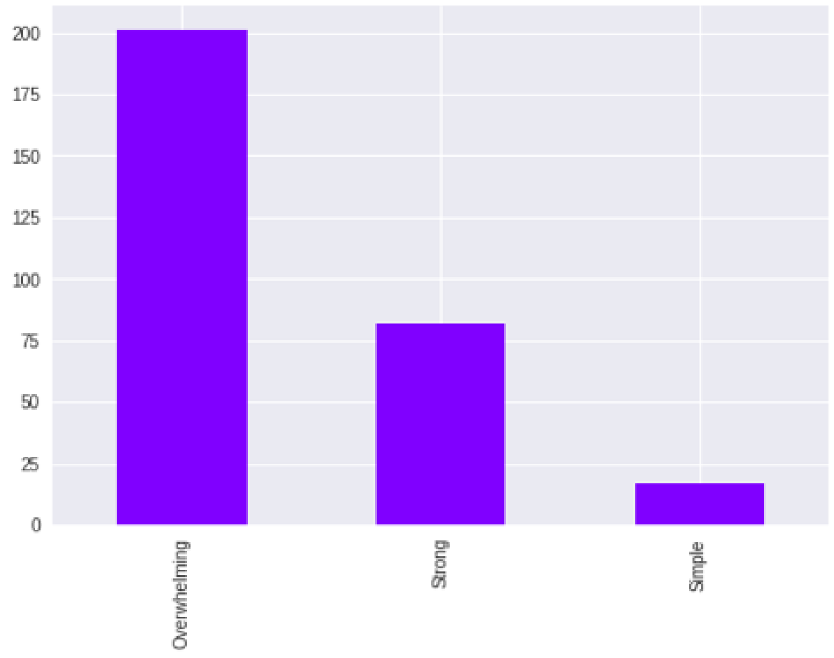


Figure 23. Distribution among agreement groups

The two co-occurrences tables (Tables 23-25) show some similar results in the use of the four annotation labels. More precisely, we can observe an overlap in the use of ‘Normal’ and ‘Spam’ and the ‘Abusive’ and ‘Hateful’. This indicates that sometimes it is not clear to annotators if a tweet is a ‘Spam’ or ‘Normal’ and similarly if a tweet is ‘Abusive’ or ‘Hateful’. Finally, the sample of Table 26 is very small in order to extract any safe conclusions.

Coexisting Labels				
Majority Label	**Abusive**	**Hateful**	**Normal**	**Spam**
Abusive	58	17	8	1
Hateful	1	2	1	0
Normal	13	13	106	31
Spam	0	0	18	35

Table 23. Cooccurrences of labels for tweets with overwhelming agreement

Coexisting Labels				
Majority Label	**Abusive**	**Hateful**	**Normal**	**Spam**
Abusive	18	15	10	3
Hateful	3	3	1	0
Normal	12	13	46	38
Spam	2	1	14	15

Table 24. Cooccurrences of labels for tweets with strong agreement

Coexisting Labels				
Majority Label	**Abusive**	**Hateful**	**Normal**	**Spam**
Abusive	1	1	1	1
Hateful	0	0	0	0
Normal	1	1	1	1
Spam	0	0	0	0

Table 25. Cooccurrences of labels for tweets with simple agreement

The correlations coefficient and similarities table also depicts some consistent results. More precisely, we have significant negative p values for 'Abusive' with all other labels (i.e., when the appearance of one variable increases the other decreases) and no important correlation in any other combination of labels. On the last column we see the Cosine Similarities between the pairs, were we notice that the vectors are much less similar, while also 'Abusive' is clearly different than Normal and Spam and not very similar with 'Hateful' either.

	PCC	p-PCC	SCC	p-SCC	KTCC	p-KTCC	CosSim
Abusive - Hateful	-0.378388	0.0000	-0.436322	0.0000	-0.375022	0.0000	0.382150
Abusive - Normal	-0.839857	0.0000	-0843253	0.0000	-0.738567	0.0000	0.195936
Abusive - Spam	-0.314023	0.0001	-0.350580	0.0000	-0.304493	0.0000	0.136757
Hateful - Normal	-0068685	0.4101	0.024373	0.7703	0.014200	0.7992	0.406726
Hateful - Spam	-0.072026	0.3876	-0.027309	0.7435	-0.025878	0.6430	0.194662
Normal - Spam	-0.028358	0.7340	0.130268	0.1171	0.111400	0.0460	0.267608

Table 26. Correlation Coefficient, P-Value & Cosine Similarity results for each pair of labels

For the sake of confidence, we also proceeded into a slightly different third round of annotations, where we select to group the labels under 'Offensive' and not 'Abusive'. The reason behind this decision is that we wanted to make sure Abusive was the most appropriate label to represent all the merged categories. The results of this round were very similar with the round described above, therefore we safely conclude that the set of labels we selected are the most appropriate ones.

9.2.3. Next step: Final Annotation Round

The final annotation round is the next thing that will take place in this work. We are planning to annotate a very big dataset of more or less 100.000 tweets, with 5 judgments per tweet. On the first round of annotations we saw that most of the randomly selected 200 tweets were actually normal or spam and very few were inappropriate. On the other hand, the vast majority of the inappropriate-sampled tweets were actually inappropriate. Therefore, we decided that on the final round we will also inject the randomly selected tweets with a small amount of sampled tweets, in order to make sure there will be sufficient inappropriate annotations on the results. The final labels we decided upon are ‘Abusive’, ‘Hateful’, ‘Normal’ and ‘Spam’.

9.3.Conclusion

In this work we focus on the annotation of a large-scale dataset. This annotation is based on inappropriate speech, in the sense of Abusive or Hateful language. We select these two types, out of several inappropriate speech categories, based on an empirical analysis of the relations between such labels. More specifically, we select the most popularly used types of inappropriate speech and conduct a series of annotation rounds. We analyze these annotations in matters of correlations and similarities between the labels and calculate their co-occurrences. When we thoroughly understand the relations between the labels, we merge some of them into the most representative one - which in this case is proven to be ‘Abusive’ - and eliminate some less important, i.e. ‘Cyberbullying’. When we have the final structure of the annotation task, the next step will be to annotate the large-scale dataset.

9.4. Section References

[1] Arpna Dhingra, Shruti Mittal, "Content based spam classification in twitter using multilayer perceptron learning", https://www.ijltet.org/journal_details.php?id=883&j_id=1880, Volume 5 Issue 4 - July 2015

[2] Santos, I., Minambres-Marcos, I., Laorden, C., Galán-García, P., Santamaría-Ibirika, A. and Bringas, P.G., 2014. Twitter content-based spam filtering. In *International Joint Conference SOCO’13-CISIS’13-ICEUTE’13* (pp. 449-458). Springer, Cham.

[3] Wang, B., Zubiaga, A., Liakata, M. and Procter, R., 2015. Making the most of tweet-inherent features for social spam detection on twitter. *arXiv preprint arXiv:1503.07405*.

[4] Zhou, Z. and Sun, L., Network-based spam filter on Twitter.

[5] Wang, A.H., 2010, July. Don't follow me: Spam detection in twitter. In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on* (pp. 1-10). IEEE.

[6] Waseem, Zeerak, and Dirk Hovy. "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter." SRW@ HLT-NAACL. 2016.

[7] Chen, Ying, et al. "Detecting offensive language in social media to protect adolescent online safety." Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Conference on Social Computing (SocialCom). IEEE, 2012.

[8] Razavi, Amir, et al. "Offensive language detection using multi-level classification." Advances in Artificial Intelligence (2010): 16-27.

[9] Black's Law Dictionary http://thelawdictionary.org
[10] Papegnies, Etienne, et al. "Detection of abusive messages in an online community." Conférence en Recherche d'Information et Applications. 2017.
[11] Park, Ji Ho, and Pascale Fung. "One-step and Two-step Classification for Abusive Language Detection on Twitter." arXiv preprint arXiv:1706.01206(2017).
[12] Nobata, Chikashi, et al. "Abusive language detection in online user content." Proceedings of the 25th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2016.
[13] Cambridge Dictionary http://dictionary.cambridge.org
[14] Davidson, Thomas, et al. "Automated Hate Speech Detection and the Problem of Offensive Language." arXiv preprint arXiv:1703.04009 (2017).
[15] Badjatiya, Pinkesh, et al. "Deep learning for hate speech detection in tweets." Proceedings of the 26th International Conference on World Wide Web Companion. International World Wide Web Conferences Steering Committee, 2017.
[16] Warner, William, and Julia Hirschberg. "Detecting hate speech on the world wide web." Proceedings of the Second Workshop on Language in Social Media. Association for Computational Linguistics, 2012.
[17] Schmidt, Anna, and Michael Wiegand. "A survey on hate speech detection using natural language processing." Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media. Association for Computational Linguistics, Valencia, Spain. 2017.
[18] Djuric, Nemanja, et al. "Hate speech detection with comment embeddings." Proceedings of the 24th International Conference on World Wide Web. ACM, 2015.
[19] Definitions for "hate speech", Dictionary.com
[20] Chatzakou, Despoina, et al. "Mean Birds: Detecting Aggression and Bullying on Twitter." arXiv preprint arXiv:1702.06877 (2017).
[21] Hosseinmardi, Homa, et al. "Analyzing labeled cyberbullying incidents on the instagram social network." International Conference on Social Informatics. Springer, Cham, 2015.
[22] Dinakar, Karthik, Roi Reichart, and Henry Lieberman. "Modeling the detection of Textual Cyberbullying." The Social Mobile Web 11.02 (2011).
[23] Riadi, Imam. "Detection Of Cyberbullying On Social Media Using Data Mining Techniques." International Journal of Computer Science and Information Security 15.3 (2017): 244.
[24] Kansara, Krishna B., and Narendra M. Shekokar. "A framework for cyberbullying detection in social network." International Journal of Current Engineering and Technology 5 (2015).

10. Conclusion

In this document we provided details about the ongoing work related to the development of automated techniques to detect early indications of malicious behavior of social network users.

During this project we attempted to identify and quantify the types of online abuse (sections 2 and 7). A number of techniques such as machine learning, deep learning and sentiment analysis have been applied to detect hateful content, raid, abuse and bulling (sections 3, 4, 5, 6, 8). The initial results indicate that such methodology is effective. The importance of having a large, good-quality annotated dataset came up multiple times during the implementation of our algorithms. Therefore, as part of this project we are also collecting a large-scale crowdsourced dataset (section 9)

This project has made significant steps towards automatically detecting malicious behavior.

11. Publications

The following publications have been submitted as part of our effort in D4.1.

1. D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, A. Vakali, ***Mean Birds: Detecting Aggression and Bullying on Twitter***, **WebSci17**, under review
2. D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, A. Vakali, ***Hate is not binary: Studying abusive behavior of #GamerGate on Twitter***, **HT'17**, under review
3. D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, A. Vakali, ***Measuring #GamerGate A Tale of Hate, Sexism, and Bullying***, WWW'17, CyberSafety workshop
4. D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, A. Vakali, ***Detecting Aggressors and Bullies on Twitter***, WWW'17, poster
5. J. Serrà, I. Leontiadis, D. Spathis, G. Stringhini, J. Blackburn, & A. Vakali, **Class-based prediction errors to detect hate speech with out-of-vocabulary words**, ACL’17
6. Antigoni-Maria Founta, Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Athena Vakali and Ilias Leontiadis **A Unified Deep Learning Architecture for Multi-facet Abuse Detection,** submitted to WWW’18
7. Enrico Mariconti, Guillermo Suarez-Tangil, Ilias Leontiadis, Nicolas Kourtellis, Emiliano De Cristofaro, Jeremy Blackburn and Gianluca Stringhini **Proactive Detection of YouTube Videos Targeted by Hate Attacks,** submitted to WWW’18
8. Savvas Zannettou, Tristan Caulfield, Emiliano De Cristofaro, Nicolas Kourtellis, Ilias Leontiadis, Michael Sirivianos, Gianluca Stringhini, Jeremy Blackburn, **The Web Centipede: Understanding How Web Communities Influence Each Other Through the Lens of Mainstream and Alternative News Sources,** IMC’17
9. Gabriel Emile Hine, Jeremiah Onaolapo, Emiliano De Cristofaro, Nicolas Kourtellis, Ilias Leontiadis, Riginos Samaras, Gianluca Stringhini, Jeremy Blackburn**, Kek, Cucks, and God Emperor Trump: A Measurement Study of 4chan’s Politically Incorrect Forum and Its Effects on the Web**, ICWSM’17

12. Copyright and Intellectual Property

The intellectual property will be jointly owned between the Institutions that each of the ENCASE partners. If a project partner decides to move institutions for the duration of the project the Institution to which they move would not become a join owner, and the ownership will remain with the institution at which partners are originally based.