

Marie Skłodowska Curie,

Research and Innovation Staff
Exchange (RISE)



European
Commission

Ref. Ares(2018)6678969 - 31/12/2018

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



WP5 – Fake activity detection and suppression

Deliverable D5.2 “Implementation of browser add-on that detects false information and fake identities in OSN”

Editor(s): Antonis Papasavva (CUT)

Author(s): Antonis Papasavva, Kostantinos Papadamou, Charalambos Partaourides, Savvas Zannettou, Petros Papagiannis, Michael Sirivianos, Costas Tziouvas, Nicolas Tsapatsoulis, Anastasios Antoniadis (CUT), Enrico Mariconti, Guillermo Suarez-Tangil, Tristan Caulfield, Juan Echeverria (UCL), Jordi Luque Serrano, Ioannis Arapakis, Ilias Leontiades (TID), Nikolas Kourtellis, Dimitrianos Savva (LST)

Dissemination Level: Public









Nature: Demonstrator

Version: 0.4

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 1 st , 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

Document History

AUTHORS

- (CUT) Antonis Papasavva, Kostantinos Papadamou, Charalambos Partaourides, Savvas Zannettou, Petros Papagiannis, Michael Sirivianos, Costas Tziouvas, Nicolas Tsapatsoulis, Anastasios Antoniadis
- (LST) Dimitrianos Savva
- (UCL) Enrico Mariconti, Guillermo Suarez-Tangil, Tristan Caulfield, Juan Echeverria
- (TID) Jordi Luque Serrano, Ioannis Arapakis, Ilias Leontiades, Nicolas Kourtellis

VERSIONS

Version	Date	Author	Remarks
0.1	07.11.2018	CUT	Initial Table of Contents
0.2	10.12.2018	ALL AUTHORS	Contribution
0.3	11.12.2018	CUT	Addition of Executive Summary and Introduction
0.4	28.12.2018	CUT	Final version

Executive Summary

Online Social Networks (OSN) have a big part of our life today. OSNs have numerous advantages as they allow people to connect, share information easily, and many more, all with some clicks in our browsers. On the other hand, these media can become very powerful tools in the hands of malicious users. Many people find the idea of hiding behind a screen to share their hate, raid, racist ideas very appealing. In addition, this becomes even more tempting knowing that they can completely cover their identity by creating a fake account in OSNs so they can stay untraceable and unlinkable with their real identity.

More scary, these malicious users often use those fake accounts to lure young people closer to them with the intention of obtain sexual contact with unsuspecting youngsters. Actually, recent studies revealed that one in five teenagers who regularly log on to the Internet says they have received an unwanted sexual solicitation via the Web. Solicitations were defined as requests to engage in sexual activities or sexual talk, or to give out personal sexual information. The most worrying part of this study is that only 25% of these youngsters told their parent about those incidents¹.

Having all the above in mind, one of the main objectives of the ENCASE project is to implement and design a browser add-on, along with its corresponding Intelligent Web-Proxy (IWP) for capturing and analysing the social network information of a user and identifying whether that user is fake. In addition, the add-on should warn the minor when he communicates with persons that lie about their identity or when they communicate with persons of bad reputation due to cyberbullying or predation.

Moreover, the custodians of the minors are notified in case any malicious activity of that kind is detected by the IWP, through their Parental Console, which was also designed and implemented as part of this effort.

This document demonstrates the browser add-on, installed on the browser of the minor that is able to detect whether a user is a sexual predator by analysing the nature of the conversation the minor had with that user. In addition, we demonstrate the practice of our research for detecting fake users and bots in Twitter. Last, we demonstrate the Parental Console, where the custodians of the youngster receive notifications of this activity, along with monitoring the chat of the minor, if and only if the minor accepts to share this information with their parents.

¹ <http://www.puresight.com/Pedophiles/Online-Predators/online-predators-statistics.html>

Table of Contents

Executive Summary.....	4
List of Figures	6
1. Introduction	7
2. Sexual Predator Detection (demo)	8
2.1. Summary	8
2.2. Methodologies and tools	8
2.2.1. Demonstration Description.....	9
2.3. Section Conclusion	9
2.4. Section Future work.....	9
3. Fake and Bot user detection in Twitter (demo).....	10
3.1. Dataset	10
3.2. Methodology.....	11
3.2.1. Features	12
3.3. Demonstration Description.....	13
4. Conclusion.....	14
5. Publications.....	14
6. Copyright and Intellectual Property.....	14

List of Figures

Figure 1. Data Collection Strategy	10
Figure 2. Abstract representation of the LOBO test. The classifier gets trained on all bot classes BCI except the target.....	12

1. Introduction

Mainstream OSNs attracted millions of users in the last years. Knowing that most of the people use social media, for-profit and non-profit organizations use these platforms for advertising their products and services. The potential of these platforms is unfortunately misused by malicious actors who use them for extracting sensitive private information from unaware users. In addition, they may use those tools to spam other users, share racist and inappropriate ideas, while staying anonymous and untraceable behind a screen. Another highly worrying misuse of these platforms is sexual predation. Sexual predators no longer have to take to the streets to find their next possible victim. They just hide their original identity and try to lurk unsuspected youngsters, using online methods and platforms.

This document demonstrates our solution for the aforementioned problems. Based on the research we undertook for Task 5.1 – “Detection of false information propagation”, Task 5.2 – “Detection of fake identity and reputation in online social networks” and Task 5.3 – “Understanding and countering fraudulent audience boosting in social networks”, we present the results the design and implementation of a browser add-on that warns users when they are chatting with sexual predators, or when they are viewing a fake or bot user in Twitter.

In addition, we demonstrate the Parental Console, a platform designed for the custodians of the youngsters where they can monitor the conversation their child has with malicious users. Also the console notifies the parents in case any other inappropriate behavior is detected by the Intelligent Web-Proxy, the machine that hosts the trained classifiers implemented in previous Tasks.

Specifically, this document demonstrates the detection of sexually abusive behavior against minors in Section 2. Our approach utilizes the minor’s conversation to determine if the person he/she is talking to is a sexual predator. In section 3, we present our methodology to detect and characterize bots and fake users in Twitter. Last, we conclude this demonstration report in Section 4.

2. Sexual Predator Detection (demo)

2.1. Summary

In this task, we formulate, train and deploy a machine learning algorithm to infer a sexual predator based on the OSN conversation. To achieve it, we developed an innovative machine learning model that efficiently captures sexual predator patterns in the conversation.

2.2. Methodologies and tools

The machine learning model was conceptualized with the purpose to increase the model capacity in capturing sexual predator patterns by including the conversation advancement in the process. The model is formulated and trained in Tensorflow and its functionality is utilized in the ENCASE framework using the Falcon software.

Due to the nature of the task, we have given special attention to make the model fast, accurate and real-time applicable. To this end, the model can predict sexual predator patterns live and on the conversation progress so far. There is no limitation on what the length of the conversation can be which is mainly due to the usage of RNNs and their inherent nature to support variable sequence length.

In more detail, the model formulation consists of a Bidirectional RNN for encoding the spoken sentence (list of words) of each speaker into a data representation whose output entails the dependencies between the words in the sentence. This representation is followed by, an RNN for encoding the conversation (list of sentences) into a data representation that entails the dependencies between the sentences in the conversation.

In conjunction, the model includes an embedding layer, before the Bidirectional RNN, for the representation of the words to numerical vectors. The whole process is trained into an end to end manner that renders the model flexible enough to capture the dependencies between the conversation and the sexual predators’ patterns. Specifically we trained the model on machine learning servers provided by TID and we used the Perverted Justice dataset. The dataset consists of several convicted sexually predators conversations.

After the training process, we deployed the trained Sexual Predator Deep Learning model on the ENCASE Proxy Server where the REST APIs are co-located. It was necessary to do that in order to be accessible by our REST API calls. In order to detect whether the person that a minor is talking to in a conversation is a sexual predator, our Front-end applications can call the `<encase_proxy_server_url>/sexual_predator_detection` REST API call that we have implemented.

To call this REST API call, one has to provide as a POST parameter a conversation URL. This URL allows our API call to request from the Data Access Layer (DAL) REST API all the information of the conversation in question. In our API call after retrieving the details of the conversation, we pre-process them in order to convert them to the format required by our DL Model and then we calling our model providing as input the pre-processed conversation to infer the probability of whether the person that a minor is talking to is a sexual predator or not. At the end, we receive this probability, which is also the output of our API call.

Following is an example of the output return of our Sexual Predator API Response:

```
{  
  "case_id": "conv_https--www.facebook.com-somebaduser.papas.5_https--www.facebook.com-petran88",  
  "predictions": {  
    "predator": "88%"  
  },  
  "desc": "This is the confidence score for the possibility the child is talking with a sexual predator. "  
}
```

2.2.1. Demonstration Description

The above model was trained and the trained classifier has been deployed on the IWP. The IWP records the conversations through Facebook and keeps them securely in its own Database. It is important to note that this database is not accessible, not even by the administrators of ENCASE and only the custodian of the minor can see this data. In addition, the data are not kept for more than 20 days in the ENCASE IWP, unless the custodian choses otherwise.

Every day the IWP will get the full conversations happened throughout the day, and send them for analysis to the *Sexual Predator Detection* trained classifier. In case the returned predator confidence is more than 65%, then the IWP will push a notification to the browser add-on of the minor, notifying them that signs of sexual predator have been detected. Relevant information like the name of the suspected predator will be displayed to the child.

In addition, the IWP will also push a notification to the Parental Console to notify the parent that a sexual predator suspect has been detected. The parent can see relevant information, along with the actual dialog the minor had between the predator. The chat between the minor and the predator will be displayed to the custodian, onbly if the minor consents to do so.

The above described detection is deployed on the IWP and it has been alpha tested. Please visit this link here: <https://www.dropbox.com/s/te1fhjhzpyh0ag1/SexualPredatorDetection.mp4?dl=0> to see the demonstration.

2.3. Section Conclusion

The sexual predator detection machine learning model manages to capture more subtle and complicated sexual predator patterns that may exist in minor's conversation. This is of utmost importance in our attempt to increase the safety of minors in OSNs.

2.4. Section Future work

The confidence score returned from the API is an indication that the minor is having a correspondence with a sexual predator. The applicability of the model in the OSN domain is under investigation and the logs derived from the usage of the model will provide valuable insights for its further improvement.

3. Fake and Bot user detection in Twitter (demo)

In this work, we set to study the problem of fake and bot user detection in Twitter. Firstly we collect a dataset that contains more than 20 different bot classes. Secondly, we propose a methodology to overcome this issue and produce a generalized bot detection method. This methodology takes into account multiple types of bots, and leverages state-of-art machine learning algorithms for detection of different types of bots. The training and testing we introduce is done using an effective “Leave-One-Botnet-Out” (LOBO) method, which allows the machine learning algorithms to train on data produced by many and diverse bots, and test its accuracy on data produced by many and diverse bots, and test its accuracy on datasets which include bots with behaviors never seen before by the classifiers.

In particular, we use this novel methodology on these classes of Twitter bots testing on over 1.5 million bots. Our approach of training a model to detect bots using single bot dataset is extremely effective, effortlessly reaching >97% accuracy. However, the way these datasets are collected prevents them from being representative of all bots in Twitter. We demonstrate that when we mix bot classes equitably in a single dataset, the prediction power of the same classifier drops significantly.

More importantly, we demonstrate that even this bot-detection system that has been trained with a variety of bots is incapable of detecting new bot families that were never observed before. In fact, some “target” botnets completely mislead the classifier resulting to less than 1% detection accuracy, meaning 99% of the bots in that class were classified as users.

This methodology provides a proxy for the real world generalization performance of the bot classifier being evaluated. It further aids in identifying how much each target class is related to the rest of the bot classes without the need of extensive and costly inspection.

3.1. Dataset

Two datasets were compiled for this work. First, a botnet dataset that contains the aggregated content generated from a variety of bot datasets and, second, a real-user dataset. Each dataset includes the information available from the user’s profile, and all the retrievable tweets at collection time in accordance to Twitter’s API limitations.

This means that each account in our dataset contains a maximum of 3,200 tweets authored or retweeted by that account. The way these datasets were finally constructed is illustrated in Figure 1, below.

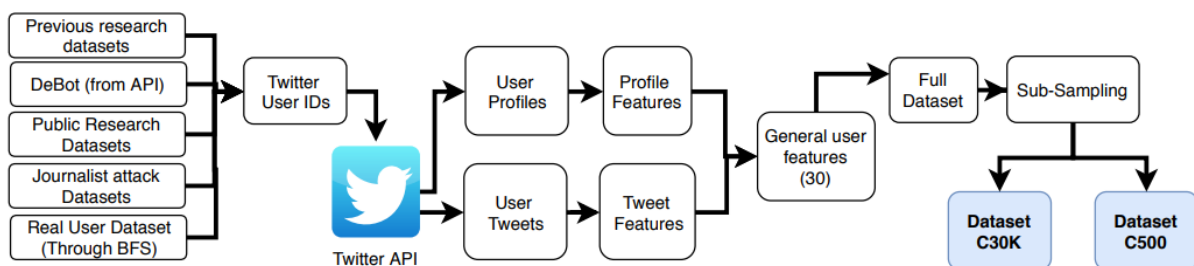


Figure 1. Data Collection Strategy

3.2. Methodology

Evaluating bot classifiers faces an important challenge. For obvious reasons, we are unable to evaluate the performance on the bots that we haven’t seen. This is a real problem, as bots mutate all the time, and botmasters are actively and creatively trying to get around any form of detection (which sometimes means suspension from the service).

In this work we propose a new form of testing accuracy for generalization of bot detection strategies. We call it the LOBO test, for Leave One Botnet Out. It derives inspiration from cross validation where a section of the available data is kept out, and used for testing on N number of “fold.” Then, the section of the data used for testing changes on each fold. The LOBO test was created to assess whether a classifier can detect a bot class, which we will call the target class, by training only with other bot classes, explicitly without any direct knowledge of the target class itself.

It was conceived strictly in the context of a binary classification between bots and users, to specifically address the variety of bot classes that such a classifier would face. We assume the LOBO test to be a proxy for generalization. For example, let us take the Bursty bots as the target class. We train the classifier with all other datasets except dataset B, then we test the classifier against dataset B. If the classifier performs well on the Bursty bots when it hasn’t actually trained on them, then it has “generalized” from the seen bot classes to this target class.

A flowchart of how the test should be applied can be seen in Figure 2. Train-Test split. We now face the need to compare between a classifier that has been trained with and without the target class. Addressing this need, we decided to use a 70-30 train-test split instead of cross validation. Each bot class is randomly and independently sampled so that 70% of each bot class is in the training set and 30% in the testing set. This is to ensure that the smallest classes will still be represented and tested properly.

This strategy allows us to test a single bot class using the 30% test data for that specific class, which has never been seen by the classifier. One could argue that comparing accuracy on 30% of the target class with accuracy tested on 100% of the target class is misleading. However, this split is not strictly part of the LOBO test, we find it useful to test on at least 30% of each bot class to provide context to the performance of the classifier on the target class when it has already trained on it.

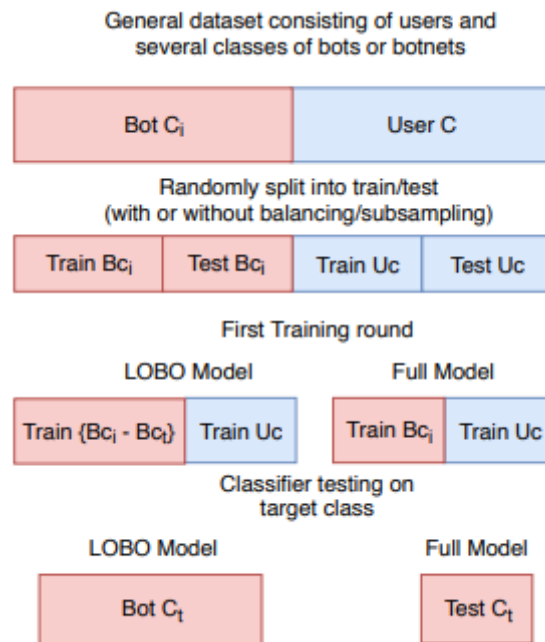


Figure 2. Abstract representation of the LOBO test. The classifier gets trained on all bot classes B_{C_i} except the target

3.2.1. Features

3.2.1.1. User profile

We include several features obtainable directly from a user profile. Seconds active and days active is the number of seconds and days between account creation and last obtainable tweet. Their maximum value is 1 month and 3 years respectively, to account for differences in collection dates. While these two features might seem redundant, seconds active is able to detect bots that tweet immediately after creation date and then fall silent.

"Days active" is better suited to address how long a user has been tweeting. Merging them in a single feature would possibly lose some of the details. Total tweet count is the number of lifetime tweets the account has created or retweeted, and this number comes directly from the profile and is not limited to the 3200 tweets that we get through the API. It includes tweets that have been deleted. We consider these profile features because they can be obtained from a single API call to the user profile (which includes the user's last tweet).

3.2.1.2. Tweet Features

We use the tweets and their text obtained from users to extract useful features. The most basic ones such as number of tweets and retweets, average tweet and retweet length, etc., are considered, and other more elaborate features are computed and used.

Hashtags are a way of grouping topics within Twitter. We have created features from the number of hashtags in analyzed tweets and retweets, number of unique hashtags, and the unique hashtag ratio. Unique hashtags are included to account for the difference between accounts tweeting many

times using a small set of hashtags against people who are involved and tweeting over many different hashtags.

Mentions are a way of publicly addressing another user. They are also commonly abused by spammers to generate engagement with unsuspecting users. We use the number of mentions in tweets and in retweets as features. We also include the number of unique mentions as a feature.

Edit Distance. To account for tweets that are equal or with small variations, we use the edit distance between tweets and retweets of a user. The edit distance (or Levenshtein distance) between two strings is the minimum number of one-character edits to turn one string into the other. Because of processing time, we only evaluate this feature for the last 200 tweets and the last 200 retweets of each user; each of these tweets is compared to the rest, and then the mean of the distances is computed.

Geolocation. Depending on user preference, each tweet can have geolocation embedded, consisting of latitude and longitude. We use the number of tweets that are geolocated, as well as the percentage of the analyzed tweets that have this information, as features.

Tweet Sources. When apps are used to publish tweets through Twitter’s API, an app publisher needs to define the "source" of the tweet. We use the number of unique sources used to publish the tweets as a feature. While some older botnets rely on using a single source to publish all of their tweets, other botnets may use as many sources as possible to confuse detection efforts. Regardless of the assumption, we calculate this feature for each of the users in our dataset.

Favorites. Marking a tweet as a favorite or “liking” it, is an action a user can take to endorse a specific tweet. We use the number of tweets a user has liked as a feature. However, we also include how many of a user’s tweets have been marked as favorites by other users, and the ratio between liked tweets and analyzed tweets (or favorites per tweet). This summarizes both directions of the endorsement: how much the user being analyzed endorses other users, and how much other users endorse the user at hand.

3.3. Demonstration Description

The above mentioned trained classifier resulted in a start-up company². There is an API developed which takes as an input the username of the Twitter account someone wants to check, and the classifier responds back with the result of the check. Specifically, it returns *normal* in case the user is not a bot, and it returns *bot* in case the classifier sensed that this user is a bot.

In our scenario, the minor visits a bot account in Twitter and the browser add-on notifies the user that this account is a bot. In addition, the same notification is sent to the Parental Console, so the custodian of the child can see the detection of the classifier.

Please visit this link below to watch the demonstration:

<https://www.dropbox.com/s/cjlolyx5wdgc19f/TwitterBotDetection.mp4?dl=0>

² For more information, please visit <http://astroscreen.co/>

4. Conclusion

In this demonstrator deliverable, we presented the functioning prototype of the ENCASE tool and how it can detect Sexual predators based on chat patterns, and Fake and/or Bots in Twitter.

We also demonstrate the aforementioned detections by the use of videos.

5. Publications

The efforts listed in this deliverable, resulted in the following publications:

1. "LOBO: Evaluation of Generalization Deficiencies in Twitter Bot Classifiers". Juan Echeverria, Emiliano De Cristofaro, Nicolas Kourtellis, Ilias Leontiadis, Gianluca Stringhini, Shi Zhou. Annual Computer Security Applications Conference (ACSAC), 2018.

6. 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.