Sexual Predator Identification using Machine Learning on Android

Bachelor's Thesis Exposé

Matthias Vogt[1,*]

[1]*Department of Computer Science, Humboldt Universität zu Berlin*

(Dated: January 3, 2020)

I. INTRODUCTION

Due to the prevalence of social media in children's lives, they are facing many risks. One of them is child grooming, which is defined as befriending and establishing an emotional connection with a child to lower the child's inhibitions, with the objective of sexual abuse (Ost, 2009) or obtaining sexual content from them such as images. This is a major concern of public safety.

This problem could be solved by creating a software that disrupts the grooming process. Outside of the scope of this thesis, in cooperation with Charité – Berlin University of Medicine – an Android app will be created for this purpose. It will analyse messenger conversations on the user's device, classifying them as grooming attempts or regular messages, and alerting the user in time to stop a possible grooming attempt. For privacy reasons, this app has to work without sending chat data to a server, thus the chat classification needs to happen on the device.

Research Goal

The goal of our research is creating a text classification algorithm. We are given an ongoing chat between a child and a chat-partner, where we know which messages belong to the child. The algorithm should decide, whether the chat-partner is making a grooming attempt. It should be able to recognize grooming attempts early, and accurately, and also run efficiently on a regular Android phone. As typical in the literature, we will implement the classification algorithm using machine learning.

II. RELATED WORK

Sexual Predator Identification in english online conversations is a heavily researched topic. A notable example is the *Sexual Predator Identification* competition held at the PAN[1] evaluation lab at the 2012 CLEF conference. The goal of the competition was to compare methods from different disciplines as to which is the most effective for sexual predator identification. A large dataset (PAN12) of chat logs between two users was given to 16 teams (Inches and Crestani, 2012). The teams had to

1. identify the predators among all users in the different conversations (problem 1), and
2. identify the part (the lines) of the conversations which are the most distinctive of the predator behaviour (problem 2).

For this thesis, only problem 1 is relevant. Additionally, our problem is easier than problem 1, as we already know which of the users is the child, and only have to decide if the chat-partner is a predator. To evaluate the approaches of the PAN competition, the organizers used an F Score, an accuracy measure based on precision an recall. They used $\beta = 0.5$ (an $F_{0.5}$ score), which favours precision over recall. The approach which the winning team used for classification is discussed in section II B.

A. Datasets

The quality of a classification algorithm is highly dependent on the dataset used for training and evaluation. Thus, we will present some important datasets from the literature.

1. The PAN12 Dataset

The PAN12 dataset consists of (P) grooming conversations between predators, and volunteers posing as children, (A) sexual conversations between consenting adults, and (N) non-sexual chat conversations. Because the availability of chatlogs of actual grooming-victims is very limited, most researchers resort to type P data. As is often the case in the literature (McGhee et al., 2011, Gupta et al., 2012, Bogdanova et al., 2014, Ebrahimi et al., 2016), this data stems from the *Perverted Justice Foundation*[2] (PJ).

*✉: matthias@vo.gt

[1] A benchmarking activity on uncovering plagiarism, authorship and social software misuse `http://pan.webis.de`

[2] `https.//perverted-justice.com`

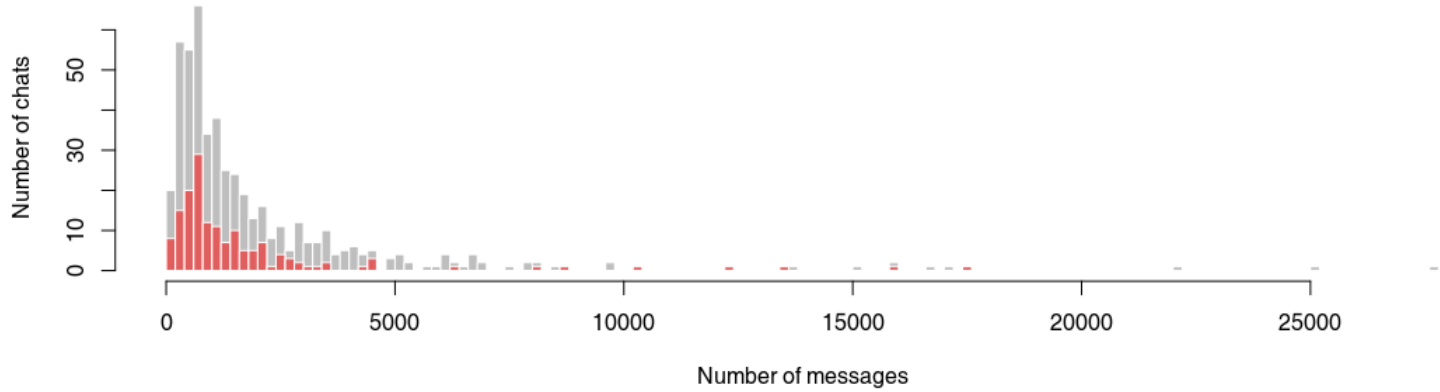

FIG. 1: Histogram of number of messages per chat in *ChatCoder2*. Labelled chats are in red.

PJ uses trained volunteers posing as children in public chatrooms in order to find sexual predators and help authorities convict them. They provide these chatlogs on their website.

There are 357 622 chats in the PAN12 dataset. Of these chats, 11 350 (about 3%) are of type P from PJ, and 346 272 are of types A and N. The latter 97% are based on logs of the chat site Omegle, and of several IRC channels. Thus, the dataset is highly imbalanced. This mixture of different types of conversations aims to represent the distribution of conversations actually happening online. This makes the detection of sexual predators challenging, because of the potential of false positives through chats of type A.

2. *The ChatCoder2 Dataset*

Upon request, we were kindly provided the *ChatCoder2* Dataset created by McGhee et al. (2011). It is a subset of the PJ dataset, containing 497 chats between predators and pseudo-victims with about 2 000 messages each on average. A subset of 155 chats with a total of 264 381 messages was manually labelled by the researchers. The messages were labelled as belonging to the categories

- *Personal Information*, with messages related to demographic details, personal values, personal relationships, and activities,
- *Grooming*, with messages related to sexual activities, sexual attitudes, sexual organs, physical intimacy, and sexual language, and
- *Approach*, with messages related to arrangement of meeting, sharing location, and isolation of the victim.

Moreover, even for unlabelled chats, the chat data always contains information about whom of the chat-partners is the predator. The distribution of the number of messages per chat is visualized in Figure 1, and the proportions of the data labelings are visualized in Figure 2.

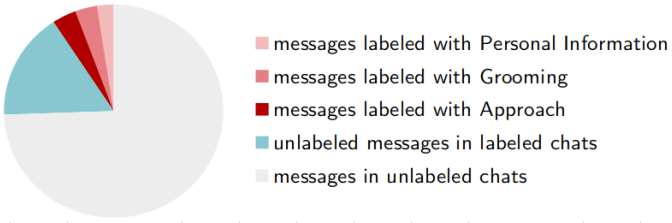

FIG. 2: Dataset labelling. Of all 1 035 101 messages, 264 381 (26%) belong to labelled chats. Of the messages in labelled chats, 25 241 messages (10%) were labelled with Personal Information, 33 457 messages (13%) were labelled with Grooming, 36 947 messages (14%) were labelled with Approach, and 168 736 messages (64%) were unlabelled.

3. *Other Datasets*

Other datasets include the MovieStarPlanet data set used by Cheong et al. (2015), with a large corpus of chatlogs from *MovieStarPlanet*, a massively multiplayer online game for children. It consists of chats by 59 predators of 40 413 lines and 8 707 non-predators of 62 704 lines, labelled as such by the *MovieStarPlanet* moderator team. To our knowledge, the paper by Cheong et al. (2015) is the only research paper based on chat data of actual victims in contrast to pseudo-victims.

Another dataset similar to *ChatCoder2* was used by Gupta et al. (2012). They manually labelled a subset of 75 chats from PJ by assigning the messages to one of the stages (1) *Friendship forming*, (2) *Relationship forming*, (3) *Risk assessment*, (4) *Exclusivity*, (5) *Sexual*, and (6) *Conclusion*. Through the labelling and word counting, they analysed the conversations in order to better understand their structure and how the stages progress.

B. Approaches in the literature

The approach with the best results of the PAN competition is described in Villatoro-Tello et al. (2012).

To solve the two PAN problems, they used a two-step approach, first identifying suspicious conversations and then identifying which of the chat-partners is the predator. Only the first step is relevant for our problem. They formulated two hypotheses upon which they based their work:

1. Terms used in the process of child exploitation are categorically and psychologically different than terms used in general chatting; and
2. Predators usually apply the same course of conduct pattern when they are approaching a child.

For the identification of suspicious conversations, the researches used two classifiers from the CLOP toolbox: Neural Networks (NN) and Support Vector Machines (SVM). The NN classifier was set as a two layer neural network with a single hidden layer of 10 units. For the SVM they tried linear and polynomial kernels. They used bag-of-words representations of chats and experimented with binary and tf-idf weighting schemes. Their classifier with the best performance is a NN classifier, which used chats represented by a bag-of-words model with a binary weighting scheme, and obtained an $F_{0.5}$-score of 0.94 on the training data. Together with the victim-from-predator classifier, which is irrelevant to our research problem, they were able to obtain an $F_{0.5}$-score of 0.93 on the competition's final test data (Villatoro-Tello et al., 2012).

For grooming detection, Bogdanova et al. (2014) compared low-level lexical features to high-level psychological features, such as emotions, neuroticism, and language patterns like fixated discourse. They used the Perverted Justice corpus for positive-, as well as chats of type A and N of negative data. They used SVMs based on character trigrams for their low-level analysis, and, for their high-level analysis, an SVM classifier, word-similarity, lexical chains, and sentiment analysis. When attempting to distinguish grooming- from non-sexual chat conversations, the researchers found that using low-level features is more effective as the vocabulary of these kinds of chats is very different from each other. Their low-level classifiers were able to reach an accuracy of 97%, while the high-level classifiers reached an accuracy of 81%. Interestingly, though, the high-level classifiers reached a much higher accuracy at distinguishing consentual adult cybersex from grooming attempts, where the vocabulary is similar. Here high-level feature classifiers reached an accuracy of 94% while low-level classifiers were only able to reach 64%.

Recently Ebrahimi et al. (2016) proposed a novel approach for detecting predatory conversations using deep convolutional neural networks (CNN). They compared this with the more traditional methods of SVMs and NN on the PAN12 corpus and CNNs performed best, obtaining an $F_{0.5}$ score of 0.86. For their implementation, they used the C++ library ConText 2.0. The researchers found a single convolution layer outperformed multiple convolution layers as multiple convolutions lead to overfitting, and that rectifier activation functions outperforms sigmoid or tanh functions. Nicely, the authors included scripts in their paper to be able to reproduce their research, which will be of great help to us.

C. Executing a machine learning model on Android

Having the execution of the model run on the android device itself is a big challenge. The model file can be large if unoptimized. However, to be able to be executed fast, the model file needs to fit in main memory. For comparison, the currently most popular smartphone memory sizes are shown in Figure 3. Moreover, the model should execute quickly in order to be able to react to messages in time, and it should not excessively consume processing power as to not disrupt the user experience. For these and other reasons, there has been little incentive to run machine learning models on Android. Instead, in practice, many developers outsource this work to a server, as usually the data to run the model on is already available on a server as well. However, this is no option for us due to privacy concerns.

RAM	Share
4GB	13.91%
3GB	16.38%
2GB	23.30%
1GB	22.90%
0.5GB	4.73%

FIG. 3: Shares of RAM sizes on Smartphones in Germany in Q2 2019, based on web traffic (DeviceAtlas, 2019)

There are multiple methods for executing the machine learning model on the user's device. The most common one is provided by the machine learning platform *TensorFlow*[3]. It provides *TensorFlow Lite*[4], which can be used to optimize and convert regular TensorFlow models for mobile use. It also enables the converted models to use hardware acceleration through Android's Machine Learning APIs. For example, Panchal (2019) has used this to build a spam-email classifier, which is a text-classification task like ours. He built a TensorFlow Model in Python and converted it for mobile with TensorFlow Lite to use it in an Android app[5]. However, at the time of writing in late 2019, TensorFlow Lite only supports a subset of TensorFlow's operations. This means, for example, that not all Long short-term memory (LSTM) Networks or other Recurrent Neural Networks (RNN) can be converted (TensorFlow Lite Documentation, 2019), which is being worked on (TensorFlow Lite 2019 Roadmap, 2019).

[3] `https://www.tensorflow.org/`
[4] `https://www.tensorflow.org/lite`
[5] `https://github.com/shubham0204/Spam_Classification_Android_Demo`

A popular language representation model is *BERT*[6] (Devlin et al., 2018)-paper. TensorFlow Lite can also be used to convert BERT models for use on mobile. Another popular machine learning framework is PyTorch which can also be ported to Android using *QNNPACK*[7] and *ONNX*[8]. Since Android apps can use Java code, in principle, Java frameworks for machine learning could be ported to work on Android as well. However, the frameworks will usually be built to be run on desktop PCs and possibly not be very performant. Moreover, Android apps can use C++ code, which is an interesting opportunity to run models because it can be more performant than Java. Furthermore, Android provides some specific on-device machine learning algorithms with *ML Kit*, however, these are not suitable for solving our problem.

III. OUR APPROACH

We will interpret the problem as a binary text classification task. Firstly we are going to further explore the *ChatCoder2* dataset to find out which chats are usable. Then we will research for adequate negative datasets of type A and N data. For clarity, we will introduce hypotheses (or assumptions) about predatory behaviour on which we will base our classification algorithm's design—similar to Villatoro-Tello et al. (2012).

We will divide the resulting dataset into a training, a validation, and a test set. Afterwards, we will fine tune a first naive BERT model from the training and validation sets, using supervised learning. We will label the chat messages as being of type P or not, and initially not use any of the grooming-stage-labels from our dataset. We will briefly evaluate it on the test corpus, and, after optimizing it a little, we will convert it to run on Android with TensorFlow Lite. Gaining some experience like this will help us conduct further, more advanced research. We will also look for ways to incorporate the grooming-stage-labels from our dataset into our BERT model.

It might be the case that our BERT model performs poorly because the standard BERT model is trained on well formed language (Wikipedia and Books, Devlin et al. (2018)), in contrast to our training data, which contains many lexical and grammatical errors. In that case, we will explore more methods of running machine learning models on Android to guide our selection as to which methods for sexual predator identification from the literature we could try and implement. Then we would further try to understand the methods used in the literature and see how we can apply them to reach our goals, keeping in mind what we know about the possibilities of Android. Specifically, finding out whether the results by Ebrahimi et al. (2016) and Villatoro-Tello et al. (2012) can be ported to Android might be important to us.

With our dataset, early identification translates into, ideally, never reaching a part of the grooming conversation labelled with *Grooming* or *Approach*. Thus, we might attempt to predict these chat labellings in a given conversation, and use a high confidence for these labels as our classifier. However, we might not have enough data to do this meaningfully. In any case, we will try approaches not using any labelling information, as these have also been successful in the literature.

Inputs and training data for our final classification algorithm will first have to be preprocessed for multiple reasons. Special Unicode characters sent by an attacker, with the intention of confusing the classification algorithm, need to be replaced or removed, and possibly, spelling should be corrected. However multiple researchers confirmed (Ebrahimi et al., 2016, Villatoro-Tello et al., 2012), that preprocessing procedures such as stemming, removal of stop words, and removal of numbers or symbols decreases classification performance as it removes important contextual information like emphasis or emoticons.

We will evaluate our classification algorithm on the test dataset, judging how accurate our classification is, and how early it can classify predatory conversations (its identification speed). Our goal is to be able to recognize grooming attempts early enough as to be able to prevent them, and accurately enough as to have the user's trust. Thus, we will we will use an $F_{0.5}$-score for accuracy measurement, as in the PAN competition, because we prefer precision over recall. To measure identification speed, we will analyse the algorithm's confidence after each message in a given chat. Moreover, we will analyze the algorithms accuracy in dependence of the number of messages given from a chat. We will also visualize this on an example. We will compare our results to a baseline classifier, like a simple SVM classifier. Finally, we will compare the execution speed of our model on desktop and mobile.

S. Ost, *Child pornography and sexual grooming: legal and societal responses.*, Cambridge Studies in Law and Society (Cambridge University Press, 2009), ISBN 9780521885829.

G. Inches and F. Crestani, Working Notes Papers of the CLEF 2012 Evaluation Labs (2012).

I. McGhee, J. Bayzick, A. Kontostathis, L. Edwards, A. McBride, and E. Jakubowski, International Journal of Electronic Commerce (2011), ISSN 10864415.

A. Gupta, P. Kumaraguru, and A. Sureka, arXiv preprint arXiv:1208.4324 (2012), 1208.4324, URL `http://arxiv.org/abs/1208.4324`.

D. Bogdanova, P. Rosso, and T. Solorio, Computer Speech and Language (2014), ISSN 08852308.

M. Ebrahimi, C. Y. Suen, and O. Ormandjieva, Digital Investigation (2016), ISSN 17422876.

Y. G. Cheong, A. K. Jensen, E. R. Gudnadottir, B. C. Bae,

and J. Togelius, IEEE Transactions on Computational Intelligence and AI in Games **7**, 220 (2015), ISSN 1943068X.

E. Villatoro-Tello, A. Juárez-González, H. J. Escalante, M. Montes-y Gómez, and L. Villaseñor-Pineda, CLEF (Notebook Papers/Labs/Workshop) pp. 1–12 (2012), URL http://www.perverted-justice.com/http://ceur-ws.org/Vol-1178/CLEF2012wn-PAN-VillatoroTelloEt2012b.pdf.

DeviceAtlas, *Most common smartphone ram by country*, https://deviceatlas.com/blog/most-common-smartphone-ram-by-country (2019), (Accessed on 10/03/2019).

S. Panchal, *Text classification in android with tensorflow*, https://towardsdatascience.com/spam-classification-in-android-with-tensorflow-lite-cde417e81260 (2019), (Accessed on 10/03/2019).

TensorFlow Lite Documentation, *Convert rnn models | tensorflow lite*, https://www.tensorflow.org/lite/convert/rnn (2019), (Accessed on 10/03/2019).

TensorFlow Lite 2019 Roadmap, *Tensorflow lite 2019 roadmap | tensorflow*, https://www.tensorflow.org/lite/guide/roadmap (2019), (Accessed on 10/03/2019).

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova (2018), 1810.04805, URL https://github.com/tensorflow/tensor2tensorhttp://arxiv.org/abs/1810.04805.