Augmenting Android Data Collection with Machine Learning

Marvin Newlin
Electrical and Computer Engineering
Air Force Institute of Technology
Wright-Patterson AFB, OH, USA
marvin.newlin@afit.edu

***Abstract*—Android devices compose the majority of mobile devices on the market today. Data mining is an important aspect for researching security of Android devices, however, there is a lack of datasets available for research purposes with Android devices. In this paper, we present a survey of Android logging and data collection tools that can be used to collect data for research with Android devices. Additionally, we explore some of the available mobile datasets and present the idea of augmenting Android data using machine learning and deep learning to synthetically generate data to increase the opportunity for research involving Android data.**

***Index Terms*—Android Logging, Data Mining, Mobile Data Collection, Machine Learning**

I. Introduction

Android devices make up a majority of the mobile device market share [1]. Research involving Android and other mobile device data requires datasets with which to conduct experiments. One issue in this area is that there is a lack of available up to data datasets to conduct experiments on mobile devices [2]. Part of this is due in part to the fact that there are privacy concerns involving the use of mobile device data because of the requirement for humans to use mobile devices in order to collect meaningful data [3].

One solution to this problem of dataset availability and privacy concerns is to synthetically generate data that possesses the same characteristics as real data which can then be used to conduct research on mobile devices. This process involves collecting data, generating synthetic data that models the real data and then evaluating the quality of that data for research purposes.

The rest of this paper is organized as follows. In Section II we detail some of the available tools for collecting data from mobile devices. Section III discusses some of the available mobile datasets and their pros and cons. Section IV discusses the idea of synthetically generating data in order to augment the process of conducting research involving mobile device data.

II. Android Logging Tools

In this section, we present a survey of various tools that exist for Android data collection. We have organized the tools into three categories: application logging tools, research data collection tools, and forensics data collection tools.

A. Application Data Collection

The default logging tool for all Android application development is Logcat [4]. This tool is built into Android Studio and is a very useful tool for debugging Android applications. Logcat works directly on top of the Android Log Application Programming Interface (API) [5]. The Android Log API defines five levels of information for logging: error, warn, info, debug, and verbose. The error level provides information on the cause of the error and the type of exception that occurs in an error. The info level is designed for logging general information in the application. The warn level is for logging information that requires a warning message along with any exceptions necessary. The debug message is designed for the application development phase and usually provides information for the developer. These messages are generally removed from applications in the release build. The verbose type provides all information in the log. All log messages also allow for a tag, a string that is conventionally the method name which the error is coming from. This allows for high granularity in the log messages.

Logcat provides a variety of information in its log messages. Logcat can dump stack traces, print debug information, or print application information [4]. Additionally, Logcat can be utilized via Android studio while developing applications or it can be used to view and filter logs via the Android Debug Bridge (ADB) [6]. One advantage of this feature with Logcat is that it allows for filtering for different log types. Logs can be filtered on their level, tag, or any regular expression, making it useful for focused data collection in applications [6].

Hirabe et al. in [7], utilize the existing Android Operating System (OS) logging to log all touch operations on an Android phone. They utilized a combination of static analysis of logs and a system dump to analyze and report the logs produced by the touch operations on an Android phone. With this tool, they were able to successfully characterize ~~and~~ single touch, multi-touch, single and multiple swipes, pinching in and out, and rotation.

B. Research Logging Tools

Spolaor et al. developed the Data Extraction and Logging Tool for Android (DELTA). They recognized a lack of tools for collecting Android data for research purposes and developed this tool to collect data for research purposes. Their

tool collects data in three different categories: sensor data, device/OS context data, and user interaction data. The sensor data category involves data collected from sensors like GPS and temperature. Device context category involves OS level data like log files. User interaction data includes elements like touch operations. Overall, DELTA is able to collect data from 44 different sources, a far sight better than its nearest competitor at 17 sources [8].

The novel part about DELTA is the development of the logging framework, the backbone of the tool. This framework allows for easy extension onto the tool and additionally allows for the customization of permission level based on the purpose of experiment, rather than over-allocating permissions ahead of time [8].

SystemSens is a system developed by Xu et al. that is designed to capture device usage information in a research context [9]. The focus of SystemSens is on usage data such as battery and CPU usage. The framework they developed consists of a phone section and a web server section that reports the information collected. They also provide the ability for adding data sources making it extensible.

C. Android Forensics Collection

On the forensics data collection side, automated collection and reporting was introduced in [10] by the MITRE corporation. This tool is focused toward security auditors, forensics investigators, and others in the forensics arena. The tool they developed is called DroidWatch. It monitors and collects data and then stores it locally in a SQLLite database and then periodically sends that data via HTTPS to a server where the data can be viewed via a web interface [10]. The application was able to collect data from a variety of sources and events like application installation/removal, call logs, GPS location, text message data, and many other sources.

DroidSpotter is another forensics tool developed in 2013 by Kramer in [11]. This tool was designed to collect Android location data from applications and identify where location information is stored on an Android phone. This tool was introduced as a prototype and only has limited capabilities.

III. Available Mobile Datasets

In this section, we detail some of the publicly available datasets for research with Android and mobile devices in general.

A. Realit Mining Dataset

The Reality Mining Dataset was first introduced in 2006 in [12]. This dataset was developed by collecting different types of data from 100 different Nokia mobile phones used by participants at Massachusetts Institute of Technology (MIT). The dataset contains information such as call logs, bluetooth devices close to the phone, cell phone tower identification information, application usage, and phone status information. The dataset was anonymized and made publicly available. One of the downsides to this dataset is its age. Since the data was collected in 2004, lots of the information is outdated and is not particularly useful for today's research contexts.

B. Nokia Mobile Dataset

The Nokia Mobile Dataset was introduced in 2010 in [13]. The dataset consists of data collected from 170 mobile phones from users in Lausanne Switzerland. The Nokia dataset contains information from four categories: social interaction data, location data, media creation and usage data, and behavioral data. Social interaction data involves call log data, text message data and bluetooth proximity data like the Reality Mining dataset. The location data includes GPS locations, cellphone network information, and wireless LAN location information. The media creation category includes information such as the location of media storage like images and videos. The behavioral data category contains information on applications used, and inference data on device utilization based on call logs and text messages. All of the data is anonymized to protect privacy information. The most useful aspect of this dataset, like the Reality Mining dataset is the location data.

C. PhoneLab Testbed

In 2013, Nandugudi et al. introduced PhoneLab, a testbed for dataset development for mobile phones [3]. Their motivation was based on a lack of available up to date datasets and they sought to design a system for open experimentation to collect mobile data for research experiments. The testbed consists of 288 Samsung Galaxy Nexus phones. The dataset they developed contains information such as display information, CPU usage, location data, power consumption, and several other categories.

One aspect to notice about these datasets is that they generally involve mostly device data. There are very few if any datasets available that contain log-based text data. In the next section we propose a solution to remedy this lack of data with synthetically generated data.

IV. Synthetic Data Generation

As detailed in Section III, there is a lack of datasets that involve text-based log data for research. In this section, we propose a solution that involves synthetically generating data from collected log data.

The end goal for synthetically generating data is to utilize machine learning from big data to help us generate a model for realistic data which we can then use to create synthetically generated datasets for research purposes. To do this, we could take a small amount of data that would be generated by the collection tools detailed in Section II or other methods. Then we would utilize a variety of machine learning techniques to extract the important features of the data and then based on those features, generate more data. From there we can continuously re-evaluate the data and feature selection to improve the accuracy of the generated data.

Implementing this model, although it sounds straightforward, is not an easy task. One of the main underlying ideas of synthetic data generation is called the Generative Adversarial Network (GAN) introduced in 2014 by Ian Goodfellow [14]. The GAN is composed of a generator and a discriminator. The purpose of the generator is to analyse the data and then

generate a synthetic output that mimics the input data. The output from the generator is then sent to the discriminator, whose job it is to determine if the input is real or if it came from the generator. The response from the discriminator is then fed back into the generator, which improves its generation based on the feedback from the discriminator. This game, depicted in Figure 1, continues until the output from the generator converges and the discriminator cannot distinguish between the generator and the real data [14].

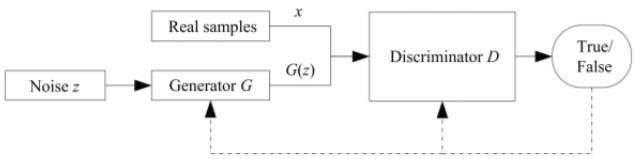

Fig. 1. Generative Adversarial Network architecture [15].

The original idea of GAN as proposed by Goodfellow et al. works well for continuous data (e.g. images or audio). However, when dealing with text or categorical data, it tends to not work as well, and text based mobile log data falls into the discrete category. To this end, some improvements to GAN have been made. The Wasserstein GAN (WGAN) was introduced in 2017 and utilizes a different distance measurement than the original GAN to measure the distance between outputs. This approach solves some of the divergence and training issues that the original GAN framework suffered from [16]. Some additional improvements have been made on the WGAN to improve its functionality on discrete data. Improved Training of Wasserstein GANs discusses the limitations that WGAN produces with weight clipping and demonstrate some of its flaws. To alleviate this problem, they propose a new method of ~~a~~ gradient penalty to maintain the 1-Lipschitz property of the Wasserstein GAN without the instability produced by weight clipping. In their evaluation of this method, they found that this improvement on WGAN helped improve its performance on discrete data.

V. Conclusions and Future Work

Android devices are a major focus of research but there is a lack of available up to date datasets for conducting research on mobile devices. In this paper we have explored different types of tools available for collecting data from mobile devices. Some of the available datasets for research have been discussed along with their pros and cons. Additionally, we have discussed a possible solution to the lack of datasets in the form of synthetically generated data and have discussed the idea of utilizing a Generative Adversarial Network to generate this data.

Future work in this area will be to develop datasets for mobile research. In the GAN arena, future work to improve the evaluation metrics for similarity will help further the ability of GANs to develop synthetic text effectively and augment existing mobile data to effectively enhance the mobile device research process.

References

[1] X. Xia, C. Qian, and B. Liu, "Android security overview: A systematic survey," *2016 2nd IEEE International Conference on Computer and Communications, ICCC 2016 - Proceedings*, pp. 2805–2809, 2017.

[2] E. Welbourne and E. Tapia, "CrowdSignals: A Call to Crowdfund the Community's Largest Mobile Dataset," in *UbiComp '14*, Seattle, WA, 2014. [Online]. Available: http://dx.doi.org/10.1145/2638728.2641309

[3] A. Nandugudi, A. Maiti, T. Ki, F. Bulut, M. Demirbas, T. Kosar, C. Qiao, S. Y. Ko, and G. Challen, "PhoneLab: A Large Programmable Smartphone Testbed," in *SENSEMINE13*, Roma, Italy, 2013. [Online]. Available: www.phone-lab.org

[4] "Logcat command-line tool — Android Developers." [Online]. Available: https://developer.android.com/studio/command-line/logcat

[5] "Log — Android Developers." [Online]. Available: https://developer.android.com/reference/android/util/Log

[6] "Write and View Logs with Logcat — Android Developers." [Online]. Available: https://developer.android.com/studio/debug/am-logcat.html

[7] Y. Hirabe, Y. Arakawa, and K. Yasumoto, "Logging all the touch operations on Android," in *2014 7th International Conference on Mobile Computing and Ubiquitous Networking, ICMU 2014*. IEEE, 1 2014, pp. 93–94. [Online]. Available: http://ieeexplore.ieee.org/document/6799073/

[8] R. Spolaor, E. D. Santo, and M. Conti, "DELTA: Data Extraction and Logging Tool for Android," *IEEE Transactions on Mobile Computing*, vol. 17, no. 6, pp. 1289–1302, 6 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8067446/

[9] H. Falaki, R. Mahajan, and D. Estrin, *SystemSens: A Tool for Monitoring Usage in Smartphone Research Deployments*, 2011. [Online]. Available: http://systemsens.cens.ucla.edu.

[10] J. Grover, "Android forensics: Automated data collection and reporting from a mobile device," *Digital Investigation*, vol. 10, pp. S12–S20, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.diin.2013.06.002

[11] J. Kramer, "DroidSpotter: A Forensic Tool for Android Location Data Collection and Analysis," Ph.D. dissertation, Iowa State University, 2013. [Online]. Available: https://lib.dr.iastate.edu/etd/13407

[12] N. Eagle and A. Pentland, "Reality mining: Sensing complex social systems," *Personal and Ubiquitous Computing*, vol. 10, no. 4, pp. 255–268, 2006.

[13] N. Kiukkonen, J. Blom, O. Dousse, D. Gatica-Perez, and J. Laurila, "Towards rich mobile phone datasets: Lausanne data collection campaign," in *Proc. ICPS*, Berlin, 2010. [Online]. Available: https://pdfs.semanticscholar.org/25d8/9cc6b650ee7ea094f6eb5f58ab8ac0802f65.pdf

[14] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," Tech. Rep., 2014. [Online]. Available: http://www.github.com/goodfeli/adversarial

[15] "Generative Adversarial Network Architecture — Download Scientific Diagram." [Online]. Available: https://www.researchgate.net/figure/Generative-Adversarial-Network-Architecture_fig1_321865166

[16] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 1 2017. [Online]. Available: https://arxiv.org/pdf/1701.07875.pdf