User Relationship Classification of Facebook Messenger Mobile Data using WEKA

Amber Umair[1], Priyadarsi Nanda[1], Xiangjian He[1], and Kim-Kwang Raymond Choo[2]

[1]School of Electrical and Data Engineering, University of Technology Sydney, Australia,
[2]Department of Information Systems and Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249-0631, USA

amber.umair@student.uts.edu.au

Abstract. Mobile devices are a wealth of information about its user and their digital and physical activities (e.g. online browsing and physical location). Therefore, in any crime investigation artifacts obtained from a mobile device can be extremely crucial. However, the variety of mobile platforms, applications (apps) and the significant size of data compound existing challenges in forensic investigations. In this paper, we explore the potential of machine learning in mobile forensics, and specifically in the context of Facebook messenger artifact acquisition and analysis. Using Quick and Choo (2017)'s Digital Forensic Intelligence Analysis Cycle (DFIAC) as the guiding framework, we demonstrate how one can acquire Facebook messenger app artifacts from an Android device and an iOS device (the latter is , using existing forensic tools. Based on the acquired evidence, we create 199 data-instances to train WEKA classifiers (i.e. ZeroR, J48 and Random tree) with the aim of classifying the device owner's contacts and determine their mutual relationship strength.

Keywords: Mobile Forensics, Social network information forensics, Weka

1 Introduction

Online social networks are a source of information, for example to profile an individual or group, to understand consumer sentiments on a particular topic, to detect an ongoing event (e.g. earthquake), to stay in touch (e.g. Facebook's Safety Check feature), etc [1]. In other words, such information can also be useful in a forensic investigation for both criminal cases and civil litigation. However, mobile device and app forensics is constantly playing catching up due to rapid changes in mobile device technologies [2, 3]. Compounding the challenge is the different formats used to store data on different devices [4, 5]. Unsurprisingly, mobile device and app forensics is an active research area. For example, the authors in [6] forensically examined 20 popular Android instant messaging apps and demonstrated how one can reconstruct message content, in different extent, from 16 of these 20 apps. Other researchers have also shown that a range of artifacts relating to user activities (e.g. login, uploading, downloading, deletion, and the sharing of files) can be recovered from a mobile forensic investigations [7–9].

Facebook messenger is another popular application (app) where a Facebook user can have text, voice or video conversations with one or more other Facebook users (e.g. one-to-one or one-to-many conversations); thus, this is the focus of this paper.

Contribution 1: Specifically, we seek to demonstrate the artifacts that can be obtained from such an app when installed on an Android device and an iOS device. We use the Digital Forensic Intelligence Analysis Cycle (DFIAC) [10] to guide the forensic investigation and use existing commercial forensic tools (i.e. FTK access data, SQLite, IPhone Analyzer) to acquire the forensic artifacts from both devices. The original DFIAC model comprises the following steps:

1. Commence(Scope/Tasking)
2. Prepare
3. Evaluate and Identify
4. Collect/Preserve/Collate
5. Analyze
6. Inference Development
7. Present, Complete / Further Tasks identified

In [10], the authors exported the metadata reports from mobile devices, and the CSV, XLS and SLSX reports were collated and manually combined into a spreadsheet. Then, the spreadsheet was converted in Pajek format for analysis. To highlight the interconnections from the acquired data, a graph (e.g. Fruchterman reingold 2D link chart) can then be created and the information analyzed, for example to identify links between individuals in seemingly disparate cases. In this paper, we limit our investigation scope to messages from only the Facebook messenger app. For example in our iOS case study, the data was acquired from a real-world suicide incident, and we are able to determine the victim's relationship strength with other contacts based on factors such as number of messages exchanged in a day or week, and time and day of the messages.

Contribution 2: We also seek to demonstrate the utility of using machine learning to classify the device owners contacts with respect to relationship strength, from the obtained forensic artifacts. Thus, in step 6 of DFIAC (i.e. Inference Development), we train three WEKA Classifiers (i.e. ZeroR, J48 and Random tree) to efficiently classify the messenger contacts of the phone owner and determine their mutual relationship strength.

Paper's Roadmap: We will now explain how the remaining of this paper is structured. In Section 2, we present our case study, as well as our experimental setup along with the tools used. Section 3 explains how we can use machine learning to determine the device owners closest contacts or friends. The last section concludes this paper.

2 Case Studies

In this section, we will describe our two case studies, namely: an Android device (see Section 2.1) and an iOS device (see Section 2.2). We also remark that our case study Section 2.2 used the backup image from the iPhone of a real-world victim.

2.1 Android Device Case Study

Table 1 summarizes the equipment used in this case study.

Table 1. Experimental Setup

Equipment	Version	Purpose
Samsung Galaxy S3	Android Version 4.3	Test device
ADB Android Debug Bridge	Android Studio 2.3.2.	Android IDE
One Root	Version 1.0	Gain super user access
Root Checker	Version 6.1.7	Verify root access
Forensic Toolkit (FTK)	FTK Imager 3.1.2.0	Disk imaging program
Dell Laptop	Intel Core i7 Windows 10 Ent	Phone images Analysis

Device Preparation: To facilitate the creation of a physical image of the Samsung Galaxy S3 device, we root the device to gain super user privileges and verify root access using the freely available One Root and Root Checker software. Android Debug Bridge (ADB) is installed on the laptop so that we can issue shell commands to the device by connecting it using a data cable.

Test Data Creation: We then create the test data by installing Facebook app on the device. We also proceed to create a test Facebook user ID and undertake the following user activities on the device:

- Sign In. (Login Id and password entered via Facebook application)
- Remove phone number
- Add Henry gray as friend
- Upload post Time is flying
- Message sent to Henry via messenger app Hi Henry, Any Plans for the weekend.
- Comment on own post And I cant do anything about it.

Imaging of phone memory: To examine the device's image, we acquire the physical (i.e. bit-for-bit) image of the device's storage, and we know that the device's memory partitions contain user specific data and are of potential forensic interest.

– */system - mmcblk0p9* is where read-only memory (ROM) is installed. Within the '/system' are a number of important folders that a user cannot normally access. For example, Location /system/app all where key ROM applications are located. Things like the device app and the messaging app /system/bin are where important binaries, which allow Android to execute the required commands, etc.
– */data - mmcblk0p12* contains information about the installed app, such as SMS and emails. Key directories here are /data/app and /data/data, which are generally wiped when a device is set to the factory default.
– */cache - mmcblk0p8* stores the temporary system data for everyday tasks, designed to expedite the system's access to apps.

Example artifacts of what we obtain from using FTK are depicted in Figures 1 to 5.

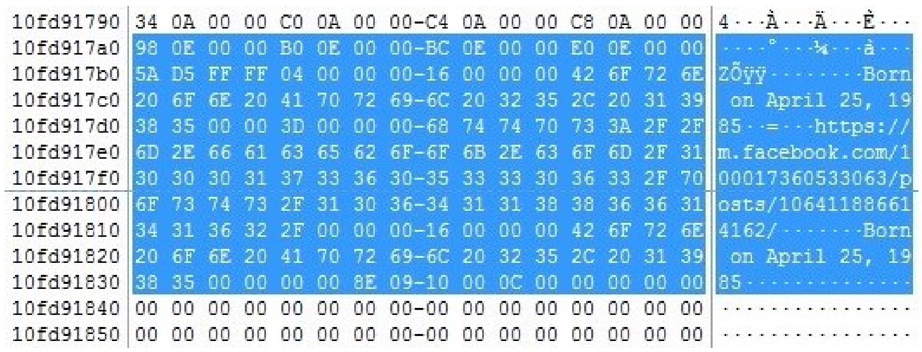

Fig. 1. User's birthday

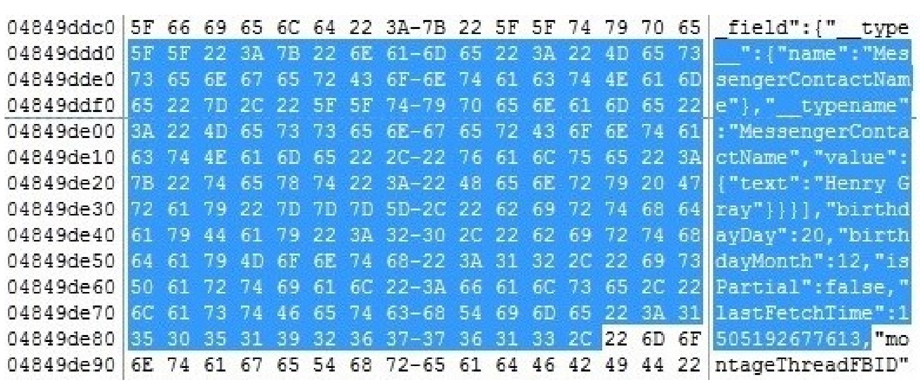

Fig. 2. User contact's birthday

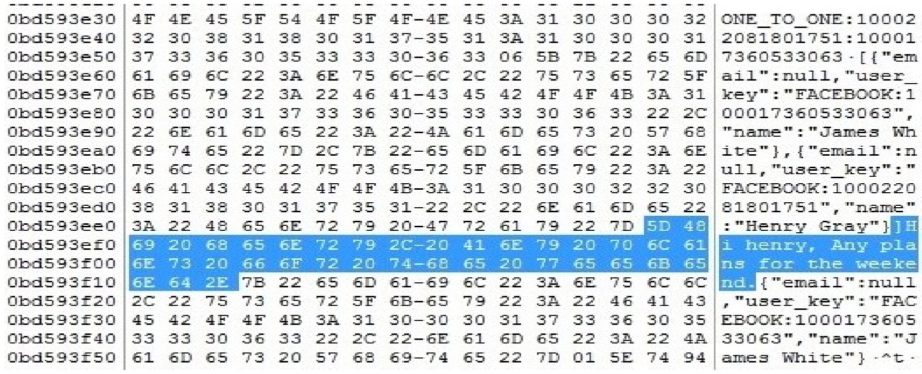

Fig. 3. Private Facebook messages

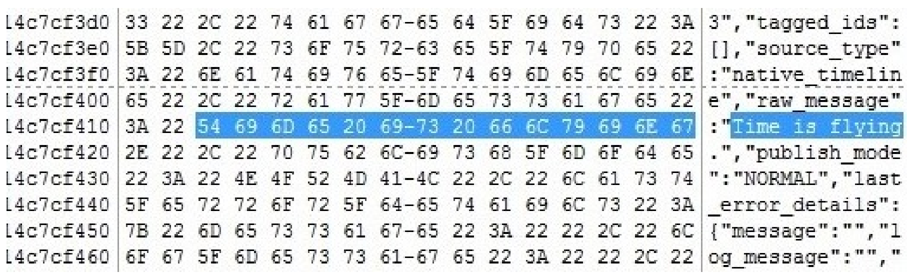

Fig. 4. Facebook status update and Comments

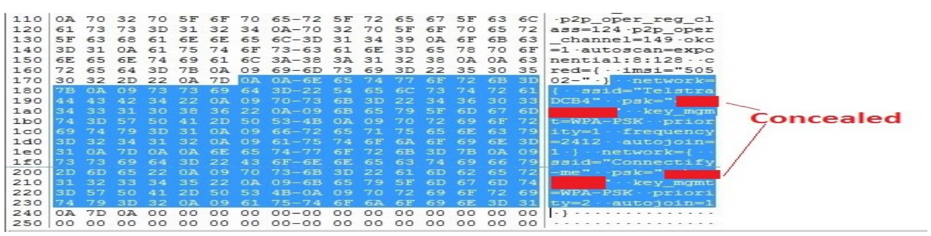

Fig. 5. WIFI and connectify details

2.2 iOS Device Case Study

The device of a teenager who had committed suicide was made available to the researchers for this research, in order to facilitate the determination of the motive and other factors relevant to the investigation. One of the evidence sources is the victim's iPhone backup files obtained from the victim's laptop. Therefore, artifacts were collected from the victim's iPhone 6 (iTunes version 12.0.1.26) backup. As the data is from an ongoing investigation, we anonymize the information to prevent the identification of the case or the individual(s) involved.

Tools used to obtain the artifacts from the iPhone are FTK Access Data, SQLite Forensic Explorer, Firefox SQLite manager and IPhone Analyzer 2.1.70. Password was not required to extract the personal data from the backup, which included contact numbers, call logs, phone messages, Facebook messenger data/ chats, notes, phone reminders /alarms, pictures, videos and audios. The iPhone Analyzer was able to pull out all details from the backup data, without requiring any passcode. Moreover, it was also able to export all data in the way it was organized on the victims phone. Figures 6 and 7 are a snapshot of what could be obtained from the phone's backup. For example, call logs and messages could be easily seen, browsed and exported. We concealed the phone numbers to protect the identity of phone owner. For the same reason, snapshots from other messages artifacts are not shown.

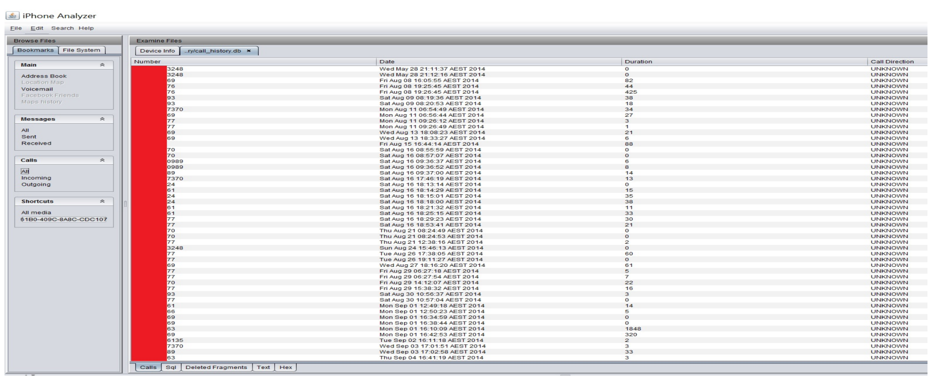

Fig. 6. iPhone Analyzer

Call logs and messages can be easily seen browsed and exported. The phone numbers are concealed to protect the identity of phone owner. Similarly messages artefacts snapshot is not shown.

3 Using Weka

In our case studies presented in the preceding section, one challenge we face is the difficulty in quickly pinpointing the more important evidences due to the different data formats, number of social apps on a device, etc. In addition, a real-world user will have possibly a number of identities for different social network accounts, a significantly larger number of contacts, etc. Thus, an investigation triage phase needs to be sufficiently robust.

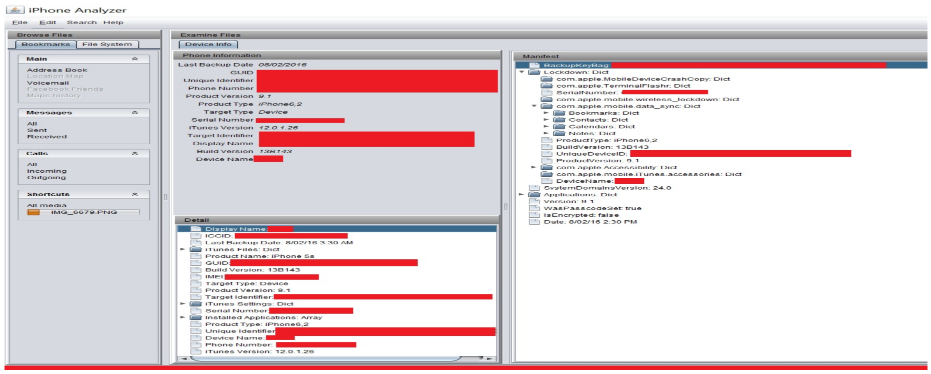

Fig. 7. iPhone Analyzer call details

We posit the importance of identifying strongly connected contacts of the device owner during a triage phase, for example by analyzing the social networking messaging app and their content. Therefore, to classify the contacts with respect to their relationship strength, we extract the data features from Facebook messenger app in our case studies. The focus is on the number of messages exchanged with a certain contact. Moreover, message exchange during certain times of the day / week (e.g. weekend) may be given a higher weight in determining relationship strength, depending on the context. In order to test the effectiveness of our approach, we analyze the message dataset of 199 instances which represent the message communication pattern of a user with his/her contacts.

Weka (Waikato Environment for Knowledge Analysis) [11] is used to determine the best performing classifiers among ZeroR [12], J48 Decision Tree [13] and Random Tree algorithms [14]. Specifically, we evaluate their performance on our dataset, based on the following key performance indicators: number of correctly identified instances, False Positive Rate(FP), Recall and F-measure.

- The correctly identified instances are the accurately classified instances, which indicate the precision of a classifier.
- FP measure denotes the number of examples predicted positive that are actually negative.
- Recall / sensitivity is the fraction of relevant instances that have been retrieved over the total amount of relevant instances.
  - Recall=True Positive / (True Positive + False Negative)
- F-measure is a measure of a test's accuracy. It is the harmonic mean of recall and precision.
  - F-measure = 2 * Recall * Precision / (Recall +Precision)

The features / attributes of our dataset are presented in Table 2. The J48 decision tree is the Weka implementation of the standard C4.5 algorithm. It starts from the training data, builds a predictive model in a tree structure. Its goal is to achieve optimal classification with a minimal number of decisions. The end nodes are the targets/classes.

Table 2. Attribute Details

Attribute	Description
User	Phone owners Facebook contact/ friend id.A,B,C,D,E
Wavg	Weekly messages exchanged. Can be less than or greater than 320. (64 msgs/day X 5 days= 320)
Weekend	Messages exchange on weekends 0-No messaging 1-Messaging on Saturday or Sunday 2-Messaging on both Saturday and Sunday
Relationship	Relationship type with phone owner W-Weak M-Medium S-Strong

Random Tree Classifier is a supervised machine-learning classifier based on constructing a multitude of decision trees, choosing random subsets of variables for each tree, and using the most frequent tree output as the overall classification. We use this classifier, as it is known to correct for the J48 decision tree classifier over-fitting issue. In this method, a number of trees are grown (i.e. a forest). Variation among the trees is introduced by projecting the training data into a randomly chosen subspace before fitting each tree. Testing this algorithm on test data resulted in reduced correctly classified instances but the tree structure revealed more detailed decisions on the data attributes as shown in Figure 8.

```
wavg < 319.5 : W (92/0)
wavg >= 319.5
|   weekend < 1.5
|   |   weekend < 0.5 : W (42/0)
|   |   weekend >= 0.5
|   |   |   user = A
|   |   |   |   wavg < 320.5 : W (1/0)
|   |   |   |   wavg >= 320.5 : M (12/0)
|   |   |   user = B : M (7/0)
|   |   |   user = C : M (1/0)
|   |   |   user = D
|   |   |   |   wavg < 321 : W (3/0)
|   |   |   |   wavg >= 321 : M (4/0)
|   |   |   user = E : W (0/0)
|   weekend >= 1.5 : S (37/0)

Size of the tree : 16
```

Random Tree

```
weekend <= 1
|   wavg <= 320: W (78.0)
|   wavg > 320
|   |   weekend <= 0: W (35.0)
|   |   weekend > 0: M (24.0)
weekend > 1
|   wavg <= 319: W (25.0)
|   wavg > 319: S (37.0)

Size of the tree :        9
```

J48 Tree

Fig. 8. Random Tree and J48 Tree

To evaluate performance of J48 decision tree classifier and random tree classifier, we compare their outputs to that of the ZeroR Classifier. ZeroR is the simplest classification algorithm and is based on frequency table. This classifier relies on the target/class only and ignores the features. It is useful for determining the baseline of a model. We analyze the data by using the following three test options using ZeroR, Decision Tree and Random Tree.

– Option 1: With K- fold cross validation(K=199)
– Option 2: With 66% Split data
– Option 3: With test data

3.1 Option 1: Classifiers with K- fold cross validation(K=100, 150, 199):

For K-fold, data is decomposed into K-blocks. Then, for K = 1 to X, the Kth block is made the test block and the rest of the data become the training data. Classifier is trained, tested, and then K is updated. Theoretically, the higher the number of folds, less biased results are achieved [15]. It is important that K¡=X, where X=no. of instances. In our dataset analysis, we use three different values of K=X=100, 150 and 199 to achieve unbiased results. ZeroR provides the baseline 69.3% accuracy for the model when used with K-fold cross validation for all three values of K (100, 150, 199). J48 classifier outperforms with a perfect correctly identified instances. Moreover, J48 classifier results remain consistent for all three values of K. The results with J48 also appears optimistic, therefore the same data are used with the random tree classifier, which results in 98.9% correctly identified instances with K=199. Similarly, other performance indicators like FP, Recall and F-measure are more realistic when using Random Tree. The changes in K value vary between the results of Random Tree classifier from 0.5% to 1%.

Table 3 summarizes the results with K-fold cross validation for all three classifiers.

Table 3. Test Option 1: With K- fold cross validation(K=100, 150, 199)

Classifier	K	Correctly classified	FP	Recall	F-measure
ZeroR	100	69.30%	0.693	0.693	0.568
	150	69.30%	0.693	0.693	0.568
	199	69.30%	0.693	0.693	0.568
J48	100	100%	0	1	1
	150	100%	0	1	1
	199	100%	0	1	1
Random Tree	100	98.40%	0.024	0.985	0.984
	150	99.40%	0.011	0.995	0.995
	199	98.90%	0.023	0.99	0.99

3.2 Option 2: Classifiers With Split Data (50%, 66%, 80%)

Initially, we tested the classifiers on Weka default split value of 66%. By splitting the data of 199 instances in 66% means that 66% of data (131 instances) were used as training and 34% (68 instances) as test.

In this test option, our classifiers show significantly decrease in precision as compared to the K-fold cross validation, but J48 and Random tree still performs with an above 90% accuracy rate. We also analyze the behavior of all three classifiers by splitting the data in 50% and 80%. J48 and Random tree achieve accuracy rates of 100% and 97.50% respectively, at 80% of data splitting. However, ZeroR achieves the highest accuracy (69.30%) at 66% data split and lowest accuracy (62.50%) at 80% split data. Table 4 summarizes the results of all three classifiers with 50%, 66% and 80% split data.

3.3 Option 3: Classifiers With Test Data

In the third test option, we provide a separate test data to Weka, to check the performance of our dataset. In this test option, Random tree classifier results improves by 0.5% as compared to option 1 (K-folds) and 6.8% as compared to option 2 (split data). Therefore, on an average the performance of the Random Tree classifier improves by 3.65% when a new/unknown test data is introduced. The performance of ZeroR and J48 is almost identical to the first test (K-folds) – see Table 5.

Table 4. Test Option 2: With Split Data (50%, 66%, 80%)

Classifier	% split	Correctly classified	FP	Recall	F-measure
ZeroR	50%	67.70%	0.677	0.677	0.546
	66%	69.30%	0.693	0.647	0.49
	80%	62.50%	0.625	0.625	0.481
J48	50%	95.95%	0.085	0.96	0.957
	66%	94.12%	0.101	0.941	0.937
	80%	100%	0	1	1
Random Tree	50%	95.95%	0.085	0.96	0.957
	66%	92.60%	0.105	0.926	0.922
	80%	97.50%	0.042	0.975	0.974

Table 5. Test Option 3: With Test Data

Classifier	Correctly classified	FP	Recall	F-measure
ZeroR	69.3%	0.693	0.693	0.568
J48	100%	0	1	1
Random Tree	99.4%	0.001	0.995	0.995

4 Conclusion and Future Work

In this paper, we studied the potential of using machine learning classifiers to facilitate mobile forensics, specifically in terms of Facebook messenger artifact triaging. Specifically, after acquiring forensic artifacts from an Android device and an iOS device, we created 199 data-instances and trained three WEKA Classifiers (i.e. ZeroR, J48 and Random tree). This was done so that we were able to classify the device owner's contact classification into weak, medium and strong (i.e. determine their mutual relationship strength). Our analysis with the three test options and three different classifiers revealed that J48 appeared to highly biased or overfitted to the provided dataset, and Random tree achieved optimal performance in all three test options with increased accuracy when tested with a different test dataset.

Future work includes extending this work to other classifiers as well as using a broader range of datasets.

Acknowledgments

The first author is supported by the Australian Government Research Training Program Scholarship.

References

1. Umair, A., Nanda, P., He, X.: Online social network information forensics: A survey on use of various tools and determining how cautious facebook users are? In: 2017 IEEE Trustcom/BigDataSE/ICESS. (Aug 2017) 1139–1144
2. Barmpatsalou, K., Cruz, T., Monteiro, E., Simoes, P.: Current and future trends in mobile device forensics- a survey. ACM Computing Surveys **51** (2018) 46
3. Dezfouli, F.N., Dehghantanha, A., Eterovic-Soric, B., Choo, K.K.R.: Investigating social networking applications on smartphones detecting facebook, twitter, linkedin and google+ artefacts on android and ios platforms. Australian Journal of Forensic Sciences **48**(4) (2016) 469–488
4. Anglano, C., Canonico, M., Guazzone, M.: Forensic analysis of telegram messenger on android smartphones. Digital Investigation **23** (2017) 31–49
5. Marturana, F., Me, G., Berte, R., Tacconi, S.: A quantitative approach to triaging in mobile forensics. In: 2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications. (Nov 2011) 582–588
6. Walnycky, D., Baggili, I., Marrington, A., Moore, J., Breitinger, F.: Network and device forensic analysis of android social-messaging applications. Digital Investigation **14** (2015) S77 – S84 The Proceedings of the Fifteenth Annual DFRWS Conference.
7. Daryabar, F., Dehghantanha, A., Choo, K.K.R.: Cloud storage forensics: Mega as a case study. Australian Journal of Forensic Sciences **49**(3) (2017) 344–357
8. Cahyani, N.D.W., Ab Rahman, N.H., Glisson, W.B., Choo, K.K.R.: The role of mobile forensics in terrorism investigations involving the use of cloud storage service and communication apps. Mobile Networks and Applications **22**(2) (2017) 240–254
9. Yang, T.Y., Dehghantanha, A., Choo, K.K.R., Muda, Z.: Windows instant messaging app forensics: Facebook and skype as case studies. PLOS ONE **11**(3) (03 2016) 1–29
10. Quick, D., Choo, K.K.R.: Pervasive social networking forensics: intelligence and evidence from mobile device extracts. Journal of Network and Computer Applications **86** (2017) 24–33
11. Azuaje, F.: Witten ih, frank e: Data mining: Practical machine learning tools and techniques 2nd edition. BioMedical Engineering OnLine **5**(1) (Sep 2006) 51

12. Lee, K., Palsetia, D., Narayanan, R., Patwary, M.M.A., Agrawal, A., Choudhary, A.: Twitter trending topic classification. In: 2011 IEEE 11th International Conference on Data Mining Workshops. (Dec 2011) 251–258
13. Patil, T.R., Sherekar, S.: Performance analysis of naive bayes and j48 classification algorithm for data classification. International Journal of Computer Science and Applications **6**(2) (2013) 256–261
14. Breiman, L.: Random forests. Machine learning **45**(1) (2001) 5–32
15. Refaeilzadeh, P., Tang, L., Liu, H.: Cross-validation. In: Encyclopedia of database systems. Springer (2009) 532–538