Neural Visualization of Android Malware Families

Alejandro González[1], Álvaro Herrero[1], and Emilio Corchado[2]

[1]Department of Civil Engineering, University of Burgos, Spain
Avenida de Cantabria s/n, 09006 Burgos, Spain
agr0095@alu.ubu.es, ahcosio@ubu.es

[2]Departamento de Informática y Automática, Universidad de Salamanca
Plaza de la Merced, s/n, 37008 Salamanca, Spain
escorchado@usal.es

Abstract. Due to the ever increasing amount and severity of attacks aimed at compromising smartphones in general, and Android devices in particular, much effort have been devoted in recent years to deal with such incidents. However, scant attention has been devoted to study the interplay between visualization techniques and Android malware detection. As an initial proposal, neural projection architectures are applied in present work to analyze malware apps data and characterize malware families. By the advanced and intuitive visualization, the proposed solution provides with an overview of the structure of the families dataset and ease the analysis of their internal organization. Dimensionality reduction based on unsupervised neural networks is performed on family information from the Android Malware Genome (Malgenome) dataset.

Keywords: Android Malware, Malware Families, Artificial Neural Networks, Exploratory Projection Pursuit

1 Introduction

Since the first smartphones came onto the market in the late 90s, sales on that sector have increased constantly until present days. Among all the available operating systems, Google's Android is the most popular mobile platform [1]. The number of Android-run units sold in Q4 2015 worldwide raised to 325.39 million out of 403.12 million units, that is a share of 80.71%. It is not only the number of devices but also the number of apps; those available at Google Play (Android's official store) constantly increase, up to more than 2.1 million that are available nowadays [2]. With regard to the security issue, Android became the top mobile malware platform as well [3] and it is forecast that the volume of Android malware will spike to 20 million during 2016 when it was 4.26 million at the end of 2014 and 7.10 million in first half of 2015 [4]. This operating system is an appealing target for bad-intentioned apps, mainly because of its open mentality, in contrast to iOS or some other operating systems.

Smartphone security and privacy are nowadays major concerns. In order to address these issues, it is required to understand the malware and its nature. Otherwise,

it will not be possible to practically develop an effective solution [5]. According to this idea of gaining deeper knowledge about malware nature, present study is focused on the analysis of Android malware families. To do so, Malgenome (a real-life publicly-available) dataset [6] has been analyzed by means of several neural visualization models. From the samples contained in such dataset, several alarming statistics were found [5], that motivate further research on Android malware. That is the case of the 36.7% of the collected samples that leverage root-level exploits to fully compromise the security of the whole system or the fact that more than 90% of the samples turn the compromised phones into a botnet controlled through network or short messages.

To characterize malware families, this study proposes the use of neural models able to visualize a high-dimensionality dataset, further described in section 2. Each individual from the dataset (a malware app) encodes the subset of selected features using a binary representation (details on section 4). These individuals are grouped by families and then visualized trying to identify patterns that exist across dimensional boundaries in the high dimensional dataset by changing the spatial coordinates of family data. The idea is to obtain an intuitive visualization of the malware families to draw conclusions about the structure of the dataset.

Neural visualization techniques have been previously applied to massive security datasets, such as those generated by network traffic [7], SQL code [8], honeynets [9], or HTTP traffic [10]. In present paper, such methods are applied to a new problem, related to the detection of malware.

Up to now, a growing effort has been devoted to detect Android malware [11]. Machine learning [12], [13] has been applied to differentiate between legitimate and malicious Android apps, as well as knowledge discovery [14], and weighted similarity matching of logs [15] among others. Although some visualization techniques have been applied to the detection of malware in general terms [16], few visualization-based proposals for Android malware detection are available at present time. In [17] Pythagoras tree fractal is used to visualize the malware data, being all apps scattered, as leaves in the tree. Authors of [18] proposed graphs for deciding about malware by depicting lists malicious methods, needless permissions and malicious strings. In [19], visualization obtained from biclustering on permission information is described. Behavior-related dendrograms are generated out of malware traces in [20], comprising nodes related to the package name of the application, the Android components that has called the API call and the names of functions and methods invoked by the application. Unlike previous work, Android malware families are visualized by neural models in present paper. Up to the authors knowledge, this is the first time that neural projection models are applied to visualize Android malware.

The rest of this paper is organized as follows: the applied neural methods are described in section 2, the setup of experiments for the Android Malware Genome dataset is described in section 3, together with the results obtained and the conclusions of the study that are stated in section 4.

2 Neural Visualization

This work proposes the application of unsupervised neural models for the visualization of Android malware data. Visualization techniques are considered a viable approach to information seeking, as humans are able to recognize different features and to detect anomalies by means of visual inspection. The underlying operational assumption of the proposed approach is mainly grounded in the ability to render the high-dimensional traffic data in a consistent yet low-dimensional representation. In most cases, security visualization tools have to deal with massive datasets with a high dimensionality, to obtain a low-dimensional space for presentation.

This problem of identifying patterns that exist across dimensional boundaries in high dimensional datasets can be solved by changing the spatial coordinates of data. However, an a priori decision as to which parameters will reveal most patterns requires prior knowledge of unknown patterns.

Projection methods project high-dimensional data points onto a lower dimensional space in order to identify "interesting" directions in terms of any specific index or projection. Having identified the most interesting projections, the data are then projected onto a lower dimensional subspace plotted in two or three dimensions, which makes it possible to examine the structure with the naked eye.

2.1 Principal Component Analysis

Principal Component Analysis (PCA) is a well-known statistical model, introduced in [21], that describes the variation in a set of multivariate data in terms of a set of uncorrelated variables each, of which is a linear combination of the original variables. From a geometrical point of view, this goal mainly consists of a rotation of the axes of the original coordinate system to a new set of orthogonal axes that are ordered in terms of the amount of variance of the original data they account for.

PCA can be performed by means of neural models such as those described in [22] or [23]. It should be noted that even if we are able to characterize the data with a few variables, it does not follow that an interpretation will ensue.

2.2 Maximum Likelihood Hebbian Learning

Maximum Likelihood Hebbian Learning [24] which is based on Exploration Projection Pursuit. The statistical method of EPP was designed for solving the complex problem of identifying structure in high dimensional data by projecting it onto a lower dimensional subspace in which its structure is searched for by eye. To that end, an “index” must be defined to measure the varying degrees of interest associated with each projection. Subsequently, the data is transformed by maximizing the index and the associated interest. From a statistical point of view the most interesting directions are those that are as non-Gaussian as possible.

2.3 Cooperative Maximum Likelihood Hebbian Learning

The Cooperative MLHL (CMLHL) model [25] extends the MLHL model, by adding lateral connections between neurons in the output layer of the model. Considering an *N*-dimensional input vector (*x*), and an *M*-dimensional output vector (*y*), with W_{ij} being the weight (linking input *j* to output *i*), then CMLHL can be expressed as defined in equations 1-4.

1. Feed-forward step:

$$y_i = \sum_{j=1}^{\mathbf{N}} W_{ij} x_j, \forall i \tag{1}$$

2. Lateral activation passing:

$$y_i(t+1) = \left[y_i(t) + \tau(b - Ay) \right]^+ \tag{2}$$

3. Feedback step:

$$e_j = x_j - \sum_{i=1}^{M} W_{ij} y_i, \forall j \tag{3}$$

4. Weight change:

$$\Delta W_{ij} = \eta . y_i . sign(e_j) | e_j |^{p-1} \tag{4}$$

Where: η is the learning rate, τ is the "strength" of the lateral connections, b the bias parameter, p a parameter related to the energy function and A a symmetric matrix used to modify the response to the data. The effect of this matrix is based on the relation between the distances separating the output neurons.

3 Experiments & Results

As previously mentioned, some neural visualization models (see Section 2) have been applied to analyze Android malware. Present section introduces the analyzed dataset as well as the main obtained results.

3.1 Malgenome Dataset

The Malgenome dataset [5], coming from the Android Malware Genome Project [6], has been analysed in present study. It is the first large collection of Android malware (1,260 samples) that was split in malware families (49 different ones). It covered the majority of existing Android malware, collected from the beginning of the project in August 2010.

Data related to many different apps from a variety of Android app repositories were accumulated over more than one year. Additionally, malware apps were thoroughly characterized based on their detailed behavior breakdown, including the installation, activation, and payloads.

Collected malware was split in families, that were obtained by "carefully examining the related security announcements, threat reports, and blog contents from existing mobile antivirus companies and active researchers as exhaustively as possible and diligently requesting malware samples from them or actively crawling from existing official and alternative Android Markets" [5]. The defined families are: ADRD, AnserverBot, Asroot, BaseBridge, BeanBot, BgServ, CoinPirate, Crusewin, DogWars, DroidCoupon, DroidDeluxe, DroidDream, DroidDreamLight, DroidKungFu1, DroidKungFu2, DroidKungFu3, DroidKungFu4, DroidKungFuSapp, DoidKungFuUpdate, Endofday, FakeNetflix, FakePlayer, GamblerSMS, Geinimi, GGTracker, GingerMaster, GoldDream, Gone60, GPSSMSSpy, HippoSMS, Jifake, jSMSHider, Kmin, Lovetrap, NickyBot, Nickyspy, Pjapps, Plankton, RogueLemon, RogueSPPush, SMSReplicator, SndApps, Spitmo, TapSnake, Walkinwat, YZHC, zHash, Zitmo, and Zsone. Samples of 14 of the malware families were obtained from the official Android market, while samples of 44 of the families came from unofficial markets.

The dataset to be analyzed consists of 49 samples (one for each family) and each sample is described by 26 different features derived from a study of each one of the apps. The features are divided into six categories, as can be seen in Table 1.

Table 1. Features describing each one of the malware families in the Malgenome dataset.

Category #1: Installation		**Category #3: Privilege escalation**	
1	Repackaging	14	exploid
2	Update	15	RATC/zimperlich
3	Drive-by download	16	ginger break
4	Standalone	17	asroot
Category #2: Activation		18	encrypted
5	BOOT	**Category #4: remote control**	
6	SMS	19	NET
7	NET	20	SMS
8	CALL	**Category #5: financial charges**	
9	USB	21	phone call
10	PKG	22	SMS
11	BATT	23	block SMS
12	SYS	**Category #6: personal information stealing**	
13	MAIN	24	SMS
		25	phone number
		26	user account

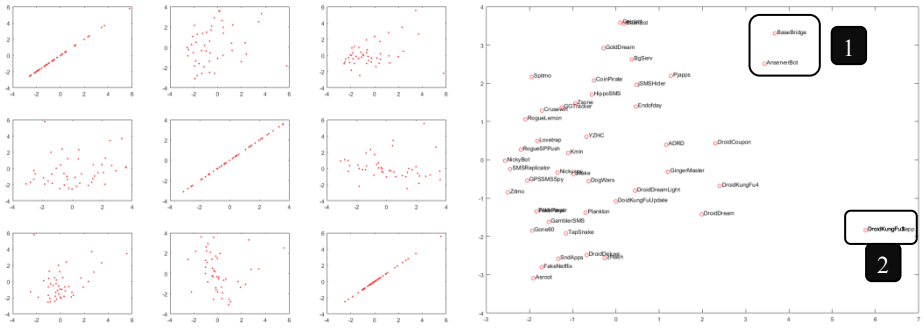
The features describing each family take the values of 0 (if that feature is not present in that family) or 1 (if the feature is present).

3.2 Results

For comparison purposes, three different projection models have been applied, whose results are shown below.

PCA Projection

Fig. 1 shows the principal component projection, obtained by applying PCA to the previously described data. Fig. 1.a corresponds to the scatterplot matrix, where the three principal components are shown pairwise; those pairs in the main diagonal of the matrix do not provide with interesting information as the same component is shown in both axes (1-1, 2-2 and 3-3). Fig 1.b corresponds to the projection obtained by combining the two principal components.



a) Scatterplot matrix b) Labeled component 1-2

Fig. 1. PCA projection of Malgenome families.

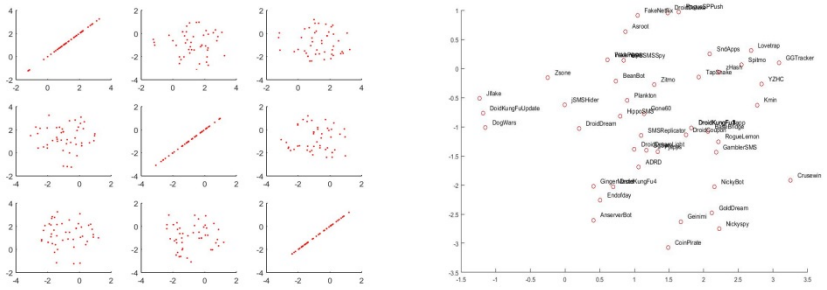
In Fig 1.b it can be seen that most of the malware families are grouped in a main group (left side of the figure) while just a few families can be identified away from this cluster (groups 1 and 2). Group 1 gathers two families (BaseBridge and AnserverBot), that are the only two families in the dataset that combine repackaging and update installation. Group 2 gathers four families (DroidKungFu1, DroidKungFu2, DroidKungFu3 and DroidKungFuSapp) that are the only ones in the dataset presenting the encrypted privilege escalation.

Additionally, this projection let us identify that some families are projected at the very same place. By getting back to the data we have realized that these families take the very same values for all the features. This is the case of Walkinwat and FakePlayer on the one hand and for DroidKungFu1, DroidKungFu2, DroidKungFu3 and DroidKungFuSapp on the other hand. It means that, by taking into account the features in the analysed dataset, it will not be possible to distinguish Walkinwat from FakePlayer malware or any of the 4 mentioned variants of DroidKungFu malware.

MLHL Projection

Fig. 2 shows the MLHL projection of the analyzed data. As in the case of PCA, Fig. 2.a represents the obtained scatterplot matrix and Fig. 2.b shows the projection on the

two main components. MLHL projection shows the structure of the data in a way that a kind of ordering can be seen in the dataset. However, as it is more clearly shown in the CMLHL projection (Fig. 3), MLHL is not further described.



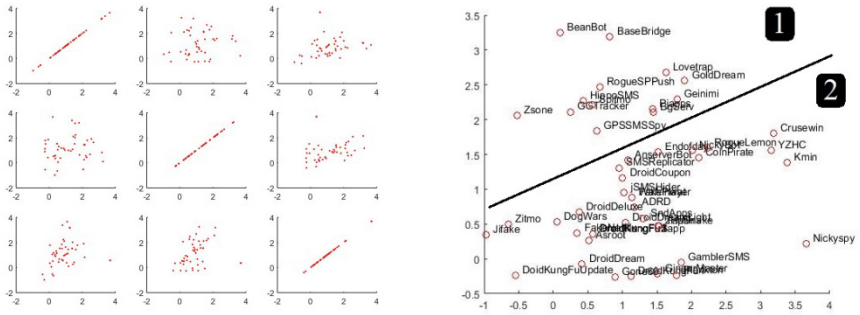
a) Scatterplot matrix b) Labeled component 1-2

Fig. 2. MLHL projection of Malgenome families.

The parameter values of the MLHL model for the projections shown in Fig. 2 are: Number of output dimensions: 3. Number of iterations: 100, learning rate: 0. 2872, p: 0.4852.

CMLHL Projection

When applying CMLHL to the analysed dataset, the projection shown in Fig. 3 has been obtained. As in previous figures, Fig. 3.a represents the obtained scatterplot matrix and Fig. 3.b shows the projection on the two main components. As expected, CMLHL obtained a sparser projection, revealing the structure of the dataset in a clearer way.



a) Scatterplot matrix b) Labeled component 1-2

Fig. 3**.** CMLHL projection of Malgenome families.

The parameter values of the CMLHL model for the projections shown in Fig. 3 are; Number of output dimensions: 3. Number of iterations: 100, learning rate: 0.0406, p: 1.92, τ: 0.44056.

In Fig 3.b it is easy to visually identify at least two main groups of data, labeled as 1 and 2. It means that families in each one of these groups are similar in a certain way. Group 1 gathers all the families with dangerous SMS activity, as SMS activation and SMS financial charges are present in all the families in Group 1. On the other hand, none of the families in this group present any of the following features: USB or PKG activation, and user-account information stealing. This group is also characterized by the almost complete absence of privilege escalation, as only one of those features (RATC/Zimperlich) is present in only one of the families (BaseBridge). Regarding group 2, none of the families in Group 2 present phone-call financial charges.

From a deeper analysis of such groups, some subgroups can be distinguished and are identified in Fig. 4. Additionally, the families located in each one of these groups are listed in Table 2.

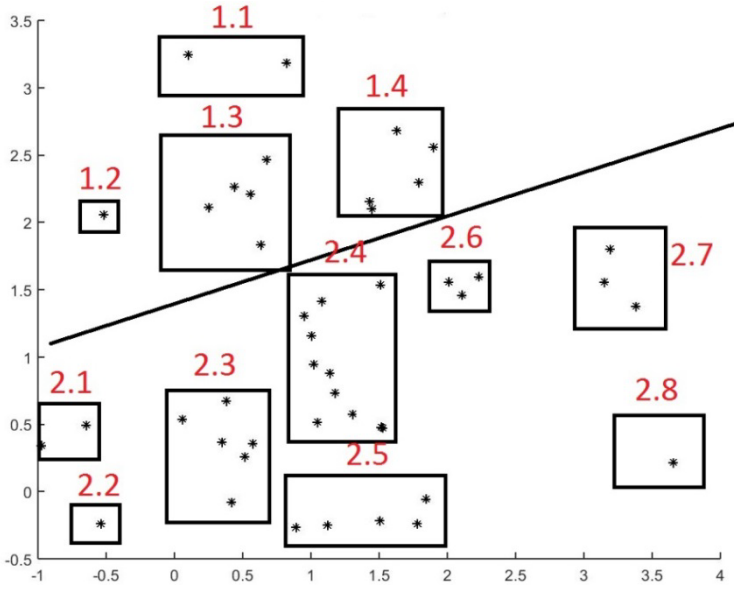


Fig. 4. CMLHL projection of Malgenome families with identified subgroups.

Table 2. Families allocation to subgroups defined in CMLHL projection.

Subgroup	Families
1.1	BaseBridge, BeanBot
1.2	Zsone
1.3	GGTracker, GPSSMSSpy, HippoSMS, RogueSPPush, Spitmo
1.4	BgServ, Geinimi, GoldDream, Lovetrap, Pjapps
2.1	Jifake, Zitmo

2.2	DroidKungFuUpdate
2.3	Asroot, DogWars, DroidDeluxe, DroidDream, DroidKungFu1, DroidKungFu2, DroidKungFu3, DroidKungFuSapp, FakeNetflix
2.4	ADRD, AnserverBot, DroidCoupon, DroidDreamLight, Endofday, FakePlayer, jSMSHider, SMSReplicator, SndApps, TapSnake, Walkinwat, zHash
2.5	DroidKungFu4, GamblerSMS, GingerMaster, Gone60, Plankton
2.6	CoinPirate, NickyBot, RogueLemon
2.7	Crusewin, Kmin, YZHC
2.8	Nickyspy

All the variants of DroidKungFu malware are located in the bottom-left side of the projection (groups 2.2, 2.3, and 2.5). Jifake and Zitmo are gathered in the same subgroup (2.1) as they are the only two families in group 2 presenting the drive-by download installation feature.

4 Conclusions and Future Work

From the projections in section 3, it can be concluded that neural projection models are an interesting proposal to visually analyse the structure of a high-dimensionality dataset in general terms. More specifically, when studying Android malware families, neural projections let us gain deep knowledge about the nature of such apps. Similarities and differences of the studied families are identified thanks to the obtained projections.

After the analysis of the CMLHL projection and the associated allocation of families in groups, it can be said that a coherent ordering is shown, consistent with the seminal characterization of Malgenome dataset [6].

In future work, some other neural visualization models will be applied to the same dataset to better understand the nature of Android malware.

References

1. http://www.statista.com/statistics/266219/global-smartphone-sales-since-1st-quarter-2009-by-operating-system/
2. http://www.appbrain.com/stats/stats-index
3. Micro, T.: The Fine Line: 2016 Trend Micro Security Predictions. (2015)
4. http://www.trendmicro.com/vinfo/us/security/news/mobile-safety/mind-the-security-gaps-1h-2015-mobile-threat-landscape
5. Yajin, Z., Xuxian, J.: Dissecting Android Malware: Characterization and Evolution. In: 2012 IEEE Symposium on Security and Privacy, pp. 95-109. (Year)
6. http://www.malgenomeproject.org/
7. Corchado, E., Herrero, Á.: Neural Visualization of Network Traffic Data for Intrusion Detection. Applied Soft Computing 11, 2042–2056 (2011)
8. Pinzón, C.I., De Paz, J.F., Herrero, Á., Corchado, E., Bajo, J., Corchado, J.M.: idMAS-SQL: Intrusion Detection Based on MAS to Detect and Block SQL Injection through Data Mining. Information Sciences 231, 15-31 (2013)

9. Herrero, Á., Zurutuza, U., Corchado, E.: A Neural-Visualization IDS for Honeynet Data. International Journal of Neural Systems 22, 1-18 (2012)
10. Atienza, D., Herrero, Á., Corchado, E.: Neural Analysis of HTTP Traffic for Web Attack Detection. In: Herrero, Á., Baruque, B., Sedano, J., Quintián, H., Corchado, E. (eds.) International Joint Conference, vol. 369, pp. 201-212. Springer International Publishing (2015)
11. Arshad, S., Khan, A., Shah, M.A., Ahmed, M.: Android Malware Detection & Protection: A Survey. International Journal of Advanced Computer Science and Applications 7, 463-475 (2016)
12. Cen, L., Gates, C.S., Si, L., Li, N.: A Probabilistic Discriminative Model for Android Malware Detection with Decompiled Source Code. IEEE Transactions on Dependable and Secure Computing 12, 400-412 (2015)
13. Sanz, B., Santos, I., Laorden, C., Ugarte-Pedrero, X., Nieves, J., Bringas, P.G., Álvarez Marañón, G.: MAMA: Manifest Analysis for Malware Detection in Android. Cybernetics and Systems 44, 469-488 (2013)
14. Teufl, P., Ferk, M., Fitzek, A., Hein, D., Kraxberger, S., Orthacker, C.: Malware Detection by Applying Knowledge Discovery Processes to Application Metadata on the Android Market (Google Play). Secur. Commun. Netw. 9, 389-419 (2016)
15. Jang, J.-w., Yun, J., Mohaisen, A., Woo, J., Kim, H.K.: Detecting and Classifying Method based on Similarity Matching of Android Malware Behavior with Profile. SpringerPlus 5, 1-23 (2016)
16. Wagner, M., Fischer, F., Luh, R., Haberson, A., Rind, A., Keim, D.A., Aigner, W.: A Survey of Visualization Systems for Malware Analysis. In: EG Conference on Visualization (EuroVis)-STARs, pp. 105-125. (Year)
17. Paturi, A., Cherukuri, M., Donahue, J., Mukkamala, S.: Mobile Malware Visual Analytics and Similarities of Attack Toolkits (Malware Gene Analysis). In: Collaboration Technologies and Systems (CTS), 2013 International Conference on, pp. 149-154. (Year)
18. Park, W., Lee, K.H., Cho, K.S., Ryu, W.: Analyzing and Detecting Method of Android Malware via Disassembling and Visualization. In: 2014 International Conference on Information and Communication Technology Convergence (ICTC), pp. 817-818. (Year)
19. Moonsamy, V., Rong, J., Liu, S.: Mining Permission Patterns for Contrasting Clean and Malicious Android Applications. Future Generation Computer Systems 36, 122-132 (2014)
20. Somarriba, O., Zurutuza, U., Uribeetxeberria, R., Delosières, L., Nadjm-Tehrani, S.: Detection and Visualization of Android Malware Behavior. Journal of Electrical and Computer Engineering 2016, (2016)
21. Pearson, K.: On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine 2, 559-572 (1901)
22. Oja, E.: Principal Components, Minor Components, and Linear Neural Networks. Neural Networks 5, 927-935 (1992)
23. Fyfe, C.: A Neural Network for PCA and Beyond. Neural Processing Letters 6, 33-41 (1997)
24. Corchado, E., MacDonald, D., Fyfe, C.: Maximum and Minimum Likelihood Hebbian Learning for Exploratory Projection Pursuit. Data Mining and Knowledge Discovery 8, 203-225 (2004)
25. Corchado, E., Fyfe, C.: Connectionist Techniques for the Identification and Suppression of Interfering Underlying Factors. International Journal of Pattern Recognition and Artificial Intelligence 17, 1447-1466 (2003)