Pertanika J. Sci. & Technol. 25 (S): 143 - 150 (2017)

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Android Malware Detection using Deep Belief Network

Wael Farouk Elsersy and Nor Badrul Anuar*

Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia

ABSTRACT

Over the last few years, the Android smartphone had faced attacks from malware and malware variants, as there is no effective commercial Android security framework in the market. Thus, using machine learning algorithms to detect Android malware applications that can fit with the smartphone resources limitations became popular. This paper used state of the art Deep Belief Network in Android malware detection. The Lasso is one of the best interpretable $\ell 1$-regularisation techniques which proved to be an efficient feature selection embedded in learning algorithm. The selected features subset of Restricted Boltzmann Machines tuned by Harmony Search feature reduction with Deep Belief Network classifier was used, achieving 85.22% Android malware detection accuracy.

Keywords: Android malware detection, Deep belief network, Feature learning, Machine learning algorithms

INTRODUCTION

In the last decade, Android became the most popular operating environment for smart devices, which makes it a very attractive target for malware attackers. As an open source nature, it has an ability to re-package benign applications with malicious code, and thus, Android malware became a real threat (Fang, Han, & Li, 2014; Faruki et al., 2015). Hence, many researchers have started investigating the behaviour of several Android malware attacks using machine learning techniques (Abdulla & Altaher, 2015; Das, Liu, Zhang, & Chandramohan, 2016). Some studies introduced static malware analysis using permission based and/or source code analysis to collect malware features (Kang, Jang, Mohaisen, & Kim, 2015; Talha, Alper, & Aydin, 2015). Other researchers employ dynamic analysis in extracting malware features. They install and run the application on the Android sandbox to capture and analyse

ARTICLE INFO

Article history:
Received: 15 August 2016
Accepted: 18 May 2017

E-mail addresses:
wfarouk@siswa.um.edu.my (Wael Farouk Elsersy),
badrul@um.edu.my (Nor Badrul Anuar)
*Corresponding Author

ISSN: 0128-7680

network traffic, SMS messages, dynamically loading other applications of malicious code and trace the log files to extract the sensitive feature in a stochastic binary format (Afonso, de Amorim, Grégio, Junquera, & de Geus, 2015; Spreitzenbarth, Schreck, Echtler, Arp, & Hoffmann, 2015). Android malware detection problem is simply a binary classification task, which determines if Android application is malware or benign. The classifier refers to algorithms that carry the classification task. Costa et al. (2015) employed Optimum-Path Forest (OPF) and Support Vector Machine (SVM) on a dataset of manually collected applications. He introduced Restricted Boltzmann Machine with Harmony Search tuning (RBM-HS) to learn vital features in an unsupervised manner. However, he concluded that the RBM is not an efficient unsupervised feature learning procedure as it could not perform with either OPF or SVM classifiers due to the use of the 9 manually selected features out of all the collected 152 features.

Therefore, this study introduces Deep Belief Network (DBN) classification for Android malware detection, in addition to ℓ1 regularisation technique for feature learning in a highly interpretable manner. It also evaluates different regularisation techniques with deep network and K-NN classifiers. The rest of the paper is organised as follows: Section 2 examines the DBN and RBM while Section 3 presents the experimental details, dataset, methodology, and results. Section 4 concludes the paper and recommends future research areas.

The DBN is a generative multi-layered deep learning model, the top two layer is undirected RBM and the bottom layers construct a directed Sigmoid Belief Network (SBN) (Hinton, Osindero, & Teh, 2006). There are no interactions between units of the same layer as shown in Figure 1. The top two hidden layers are trained using labeled data which is normalised by the probability distribution of $h^{(2)}$ and $h^{(3)}$. These two layers will prepare a good representation of the original data into a form that can be easily computationally explored or exploited. After learning the model weights, the DBN classification employs a simple sigmoid function, i.e. logistic regression classification as seen in section 2.2.

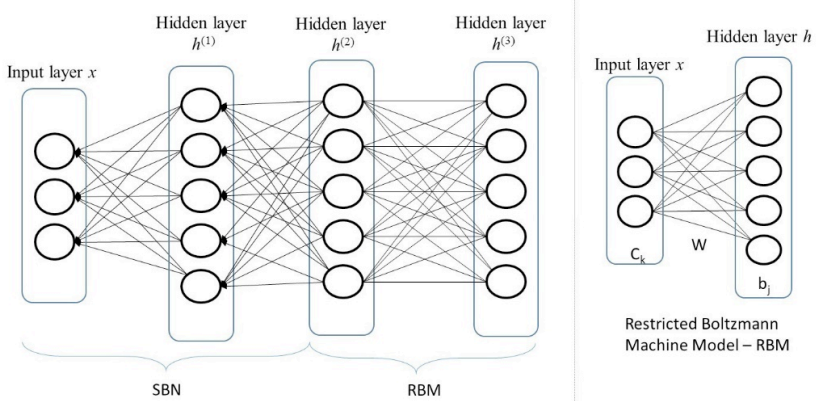

Figure 1. Deep Belief Network graphical model

Restricted Boltzmann Machine is basically an unsupervised feature learning algorithm designed for pattern recognition (Ripley, 1996) and spam detection (da Silva et al., 2016). The RBM, on the other hand, is undirected probabilistic graphical model containing a layer of observable variables in input layer *x* and a single hidden layer *h* with latent binary variables as shown in Figure 1. The RBM defines the distribution over the input layer that involves latent variables and corresponds to binary hidden units. This joint distribution is represented by an energy function, which infers the most probable variable and drives the energy function to the minimum.

$$E(x,h) = -\sum_i \sum_k W_{i,k}\, h_i x_k - \sum_k c_k x_k - \sum_i b_i h_i \quad [1]$$

Where *x* is binary state of the input units, *h* is binary state of the hidden units, *W* is weights between input layer *x* and hidden layer *h, c* is the input layer bias that expresses the preference over the values of *x* and likewise, *b* is the hidden layer bias. If the bias value is positive, it means the preference of corresponding layer variable is most likely to be selected. The tuning parameter selection requires practical experience in selecting the fine-tuning parameters. Studies have looked at ways to tackle this issue (Papa, Rosa, Marana, Scheirer, & Cox, 2015) by optimising energy function using harmony search optimisation (Costa et al., 2015).

The RBM model selection and fine tuning are the main challenges of RBM, which is basically an optimisation task to achieve the minimum square error. Harmony Search (HS) is a meta-heuristic optimisation technique for fine tuning the RBM (Papa et al., 2015). HS employs two parameters to achieve the optimal model, the first parameter is Harmony Memory Considering Rate (HMCR) which is responsible for suggesting a new model based on the previous model and Pitch Adjusting Rate (PAR) parameter to skip the local minimum traps.

The constructive feature learning process in RBM layers is followed by the directed Sigmoid Belief Network layers using the conditional distribution which uses logistic regression classification (Hinton et al., 2006).

$$p\left(h_j^{(1)} = 1 \mid \boldsymbol{h}^{(2)}\right) = \sigma\left(\boldsymbol{b}^{(1)} + \boldsymbol{W}^{(2)^T}\, \boldsymbol{h}^{(2)}\right) \quad [2]$$

$$p\left(x_i = 1 \mid \boldsymbol{h}^{(1)}\right) = \sigma\left(\boldsymbol{b}^{(0)} + \boldsymbol{W}^{(1)^T}\, \boldsymbol{h}^{(1)}\right) \quad [3]$$

MATERIALS AND METHODS

This section explains the methodology used in assessing the robustness of the DBN in Android malware detection in addition to the effectiveness of the RBM feature learning (Costa et al., 2015; da Silva et al., 2016; Papa et al., 2015) and Lasso feature shrinkage and selection techniques.

This study uses the 414 applications from DroidWare[1] dataset which was collected manually by Costa in 2015 (Costa et al., 2015). The original dataset consists of 152 permissions based features, each feature represents the list of granted permissions to each application. The dataset consists of 278 benign (good) apps that is collected from Google Play[2] and 121 Android malware apps, which was collected from the Virstotal[3] website.

The K-NN classifier detects Android application with malicious intentions. It uses Euclidean distance *K* to classify the apps with respect to the nearest *k* neighbors. Class library (Ripley, 1996) in *R* for K-NN and deepnet library for DBN are engaged. The K-NN classifier is preferred by researchers in low dimension classification techniques. In this study, we hired K-NN to validate DBN classifier with different regularisation algorithms LASSO and RBM-HS.

Lasso, also known as the ℓ1 norm, is a very high-pitched and interpretable feature shrinkage algorithm which was developed by Rob in 1996 (Tibshirani, 1996). It is very effective in feature selection by means of absolute regularisation penalty. The idea behind Lasso is to remove the irrelevant features from the dataset by driving feature coefficient values to zeros and this explains the Lasso high interpretability. Let us have n samples of applications, p is the total number of features, X and Y is the classification label of being malware of benign mobile application. The best coefficients $\beta_o, \beta_1, \beta_2, \ldots, \beta_p$ are the values that minimise the residual sum of squares RSS:

$$RSS = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 \quad [4]$$

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j| \quad [5]$$

where $\lambda \geq 0$ is a tuning parameter, Lasso introduced a shrinkage penalty, ℓ1 which is controlled by tuning parameter λ. The coefficient values are scattered when λ is very small while the coefficients penalised gradually to zero as λ increases. The optimum value of the fine-tuning parameter λ is achieved when the MSE is minimum.

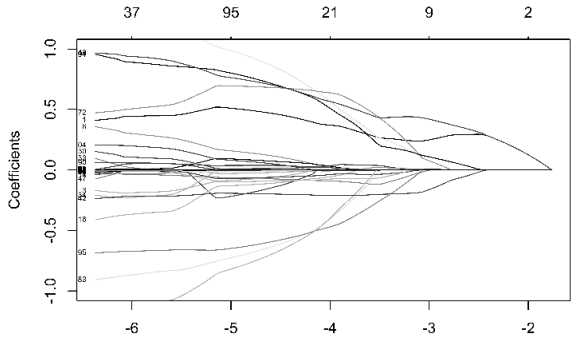

Figure 2. Feature shrinkage using Lasso Regression

[1]https://github.com/RECOVI/DroidWare.git
[2]https://Play.google.com/store
[3]http://www.virustotal.com/

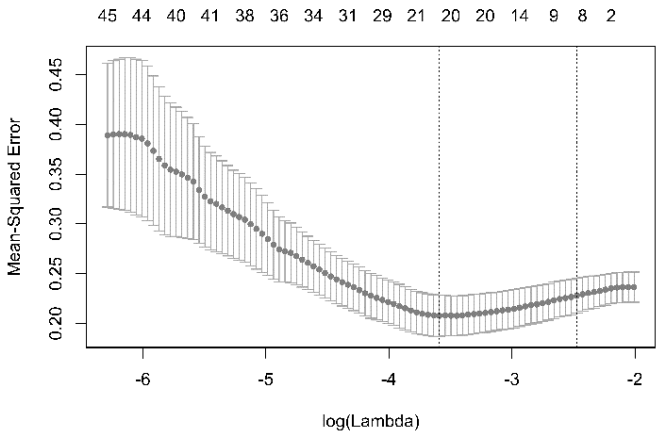

Figure 3. Mean Squared Error against shrinkage penalty

RESULTS AND DISCUSSION

We conducted three experiments to evaluate the robustness of the DBN and K-NN in Android malware detection with different feature learning techniques. The first experiment conducted is feature learning using Lasso as shown in Figure 2. The second experiment, the K- NN classifier is induced to the ORIGINAL dataset and then with regularised subsets learned by Lasso and RBM-HS. The third experiment introduced the DBN classifier to classify the same datasets as with K-NN.

Table 1
Results considering K-NN classifier

Rounds	ORIGINAL	RBM-HS	Lasso-DS
1	73.90%	65.00%	75.00%
2	77.30%	78.00%	72.17%
3	76.50%	78.00%	79.10%
4	77.00%	65.00%	75.60%
5	80.00%	68.00%	72.00%
6	76.50%	69.00%	75.60%
7	74.70%	69.00%	71.30%
8	85.20%	63.00%	77.30%
9	72.10%	65.00%	80.00%
10	79.10%	69.00%	80.80%
Avg.	77.23%	68.90%	75.89%

The ORIGINAL dataset randomly formed 10 round each round is a random selection of 70% of the data for the training and 30% for test-ing. Lasso and RBM-HS learned features data subsets are also randomised to create different 10 rounds. Using Lasso regression, as shown in Figure 3, Lasso features subset formed a subset dataset called Lasso-DS.

Table 1 shows the accuracy results when applying K-NN classification to the ORIGINAL dataset, RBM-HS reduced dataset and Lasso-DS reduced feature dataset. Each dataset is randomly split into training and testing data into 10 rounds as referred to DroidWare (Costa et al., 2015) dataset in section 3.1. Although RBM-HS did not show robust accuracy results with K-NN classifier, Lasso with its selected features, shows a very close accuracy to the results obtained when applying K-NN to the ORIGINAL dataset with only 1.34% difference.

Table 2
Result considering DBN classification

Rounds	ORIGINAL	RBM-HS	Lasso-DS
1	83.10%	85.00%	84.80%
2	87.90%	89.00%	83.05%
3	86.60%	89.00%	87.00%
4	84.40%	82.50%	85.30%
5	87.90%	84.00%	83.50%
6	84.00%	84.50%	85.30%
7	83.10%	84.50%	83.10%
8	90.00%	84.00%	86.10%
9	82.70%	82.50%	87.40%
10	85.70%	84.50%	86.60%
Avg.	85.54%	84.95%	85.22%

Table 2 shows the result of testing DBN classifier using the ORIGINAL dataset in 10 rounds which gave a better accuracy than K-NN as shown in Table 1. It is also far better than OPF and SVM (Costa et al., 2015). The big achievement of applying DBN to Lasso feature dataset emphasise that the accuracy rate is similar to when applying DBN as a classifier to all the 153 features ORIGINAL dataset with 0.32% difference. Table III compares between OPF, SVM, K-NN and DBN classifiers in the situation of using ORIGINAL dataset and when introducing RBM-HS feature learning algorithm. The average accuracy results of OPF and SVM classifiers is referred to as the algorithms used by Costa in 2015 (Costa et al., 2015). Table 3 shows that using DBN classifier with pre-learned features by means of Lasso gave better model compared with applying the same classifier to RBM-HS feature learning reduced dataset.

Table 3
Average accuracy results of OPF, SVM, K-NN, and DBN

	OPF	SVM	K-NN	DBN
ORIGINAL	72.42%	81%	77.23%	85.54%
RBM-HS	57.50%	68.50%	68.90%	84.50%
Lasso-DS	-	-	75.89%	85.22%

Table 3 also shows that Lasso initialisation works better with deep network classifiers. Currently, the present authors are working on other search and regularisation techniques to enhance the accuracy of the deep network. Lasso steadily provides more robust accuracy than RBM-HS because HS as a meta-heuristic optimisation technique which depends on random variable selection HMCR and PAR, and it cannot always guarantee a global optimal. Hence, HS may fall into local optima in which the least means the error is not reached.

CONCLUSIONS

Android has become the most popular smartphone environment in the last decade, which attracts the attackers to develop a huge number of malware variants every day. Using different machine learning regularization techniques can reduce the required amount of data from the malware analysis. This paper employed Lasso feature shrinkage algorithm that is efficient and has interpretability feature selection. The Deep Belief classifier with Lasso and RBM-HS regularisation techniques were employed to evaluate the performance of DBN classification with K-NN, SVM, and OPF classification.

The RBM-HS was not very effective when it was engaged with OPF; however, Lasso regularises the features from 152 features in the ORIGINAL dataset to 8 features with one standard error far from least MSE. Lasso had shown a precise feature selection results when it initialised DBN classifier in addition to showing a reasonable accuracy with K-NN classifiers unlike RBM-HS with OPF, K-NN or SVM. The study has proven that Lasso proves more flexibility in feature learning than RBM-HS feature learning. This RBM-HS behavior might be due to its nature of being a generative meta-heuristic which cannot guarantee global optima of RBM fine tuning model while Lasso is more discriminative. We look forward to fine tuning the RBM expanding other algorithms and expand the ORIGINAL dataset to include more information and effective features to enhance the performance of the detection model.

REFERENCES

Abdulla, S., & Altaher, A. (2015). Intelligent approach for android malware detection. *KSII Transactions on Internet and Information Systems, 9*(8), 2964-2983.

Afonso, V. M., de Amorim, M. F., Grégio, A. R. A., Junquera, G. B., & de Geus, P. L. (2015). Identifying Android malware using dynamically obtained features. *Journal of Computer Virology and Hacking Techniques, 11*(1), 9-17.

Costa, K. A. P. d., Silva, L. A. d., Martins, G. B., Rosa, G. H., Pereira, C. R., & Papa, J. P. (2015). Malware detection in android-based mobile environments using Optimum-Path Forest. IEEE 14th International Conference on Machine Learning and Applications (ICMLA).

da Silva, L. A., da Costa, K. A. P., Ribeiro, P. B., de Rosa, G. H., & Papa, J. P. (2016). Learning spam features using restricted boltzmann machines. *Iadis-International Journal on Computer Science and Information Systems, 11*(1), 99-114.

Das, S., Liu, Y., Zhang, W., & Chandramohan, M. (2016). Semantics-based online malware detection: Towards efficient real-time protection against malware. *IEEE Transactions on Information Forensics and Security, 11*(2), 289-302.

Fang, Z., Han, W., & Li, Y. (2014). Permission based android security: Issues and countermeasures. *Computers and Security, 43*, 205-218.

Faruki, P., Bharmal, A., Laxmi, V., Ganmoor, V., Gaur, M. S., Conti, M., & Rajarajan, M. (2015). Android security: A survey of issues, malware penetration, and defenses. *Communications Surveys and Tutorials, IEEE, 17*(2), 998-1022.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation, 18*(7), 1527-1554.

Kang, H., Jang, J. W., Mohaisen, A., & Kim, H. K. (2015). Detecting and classifying android malware using static analysis along with creator information. *International Journal of Distributed Sensor Networks*.

Papa, J. P., Rosa, G. H., Costa, K. A., Marana, N. A., Scheirer, W., & Cox, D. D. (2015). On the model selection of Bernoulli Restricted Boltzmann Machines through harmony search. *Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference* (pp. 1449-1450). ACM.

Papa, J. P., Rosa, G. H., Marana, A. N., Scheirer, W., & Cox, D. D. (2015). Model selection for Discriminative Restricted Boltzmann Machines through meta-heuristic techniques. *Journal of Computational Science, 9*, 14-18.

Papa, J. P., Scheirer, W., & Cox, D. D. (2015). Fine-tuning deep belief networks using harmony search. *Applied Soft Computing*.

Ripley, B. D. (1996). Pattern recognition via neural networks. A volume of Oxford Graduate Lectures on Neural Networks, title to be decided. Oxford University Press. Retrieved from http://www. stats. ox. ac. uk/ripley/papers. html.

Spreitzenbarth, M., Schreck, T., Echtler, F., Arp, D., & Hoffmann, J. (2015). Mobile-Sandbox: Combining static and dynamic analysis with machine-learning techniques. *International Journal of Information Security, 14*(2), 141-153.

Talha, K. A., Alper, D. I., & Aydin, C. (2015). APK Auditor: Permission-based Android malware detection system. *Digital Investigation, 13*, 1-14.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B(Methodological)*, 267-288.