ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 7, Issue 2, pp: (23-31), Month: October 2020 – March 2021, Available at: www.paperpublications.org

Steganalysis on Android Applications using Generative Adversarial Networks

[1]Coulibaly Fatoumata Siga, [2]Zhang Jie Fu

[1]School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China

[2]School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China

Abstract: **Stegomalware is analyzed using a steganographic algorithm which helps the programmers of malware with avenues for concealing malicious payloads. These algorithms have proven to be very efficient because most of the stegomalware deployed in the Android markets has yet to be identified, detected, or taken down. Encoding schemes such as Steganography provide traces of the text's calculation as an outcome of changes needed to embed the text information. This is as a result of digital assailants rising rapidly, and they need to curb them, i.e., recognizing whether the malicious object has or has not been embedded into the data. To do so, a Steganalysis approach is carried out to substantiate the real and forged Information, and the analysis will determine whether the data embedded is malicious or not. To establish if an application has a secret embedded payload, you have to determine the size of the payload in terms of length and establish the real payload. The concept of steganalysis is vital, holding the fact that the application of steganalysis on these *stegamalware* has proven to be efficient. In our research paper, we are using the generative adversarial network to analyze stegamalware in our android applications so that we can deliver secure applications free from malware. In our paper, we will use a secure generative adversarial network for Steganalysis. The model is secure and pontifically efficient in stegamalware generation, which is easily detectable by the discriminator and can substantiate the stegamalware and something else that isn't malware. We have evaluated the effectiveness and efficiency of the generative adversarial network from applications data trained for steganalysis. The promising results demonstrate that the model is useful, and its classification is very accurate.**

Keywords: **PSO-particle swarm optimization, CNN-convolutional neural networks, GA-Genetic Algorithm, ACO-Ant colony optimization, GANS-generative adversarial network, SSGAN-Secure Steganography Generative Adversarial network, SGANs-secure Generative adversarial network, NN-Neural networks, API-Application programming interface, EC-Evolutionary computation, EA-Evolutionary algorithm.**

1. INTRODUCTION

Our research is essentially pertinent to the steganalysis of a mobile application using generative adversarial networks. What has necessitated this paper is simply because of various malware which typically affects our mobile devices. Nonetheless, numerous clients aimlessly award consent to obscure applications and accordingly subvert the consent framework's motivation. As a result, malicious applications are hardly constrained by the Android permission system in practice [1]. In 2002, many malware was detected, tallied to four hundred and eighty-two thousand five hundred and seventy-nine in a month. Day by day, this malware became stealthier and used advanced obfuscation techniques that made them practically indistinguishable [5]. For instance, Droidkungfu is also referred to as Trojan behavior. It could overcome all the malware detectors in a backdoor mechanism, affecting attaching itself to the device. It affects your personal Information, hence having the possibility of controlling the device; this kind of malware was discovered in advance by the United States of America researcher lab and reported to Google [9]. The mode in which DroidKungFu attacks the subject is by encrypting itself into the folder's assets and then decrypting itself when the applications run. Except for the

DroidKungFu variant called Ginger Master, it is another delegated case of cell phone malware that purposely attempts to conceal itself by loading its malicious payload using a PNG format JPG images inside the asset's directory. Digital assailant makes use of Steganography in code applications, making them difficult to detect through static analysis. Some malware programmers accomplish their task using a steganography mechanism to conceal some executable files in the app assets' hierarchical structure. Stegamalware is analyzed using a steganographic algorithm, which helps the programmers of malware with avenues for hiding malicious payloads, rather than others using the currently used mechanism. This method has proven to be very efficient because most of the stegomalware deployed on the Android markets has been identified, detected, and taken down[16]. A Steganalysis approach is carried out to substantiate real and forged information. To establish if a channel has a secret embedded payload, finally select the payload's size in terms of length and establish the real payload. The concept of steganalysis is vital, holding the fact that the application of steganalysis on this stegomalware is proven to be proficient.

2. LITERATURE REVIEW

In our research, we shall focus on the topography of stegomalware that has been on the rise. In the past, researchers have done a lot of work and have registered great success in malware analysis. This has been well facilitated by the advancement in machine learning. Malware has a ravaging effect hence promoted a lot of researchers to request more funding to have more innovative solutions to this problem.

Better developments have now come into place as they have allowed more excellence and better protection against malware to be made available to business and technology owners.

In fact, Stolfo.M [11] attempted to detect malware using machine learning. He used executable files and checked for the possible feature characteristics to classify the kind of malware. This famous guy made use of a naïve Bayes classifier; for feature selection, and he ended up with a very accurate and significant system during that time, which could not be compared to any antivirus.

Across various analyses made using machine learning, many researchers have demonstrated that machine learning has a great impact on malware detection; among the featured tested included executable files, which encompassed gathering n-grams from the bytes. Text classification was initially made by discovering n-grams. This technique was employed by Assaleh.M [5] for malware classification. The generated sequence of bytes was dependent on the executable files. Their continued repetitive occurrence inside the file was considered a real attribute. The handwork of Assaleh.M [5] has assisted many researchers as it gave them the power to pre-establish features, which helped them classify the malware.

Over the many years, because of positives with classifiers, more features for malware classification have been brought into books. The number of classifiers, namely SVM, MNN, and ANN, performs their work at varying degrees; hence, classification accuracy can be compromised between them, which is more accurate?.

To obtain a more simplistic and accurate classifier. New researchers are trying to up better classifiers such as generative adversarial networks that work better for them. The classifier addresses all the shortcomings of another classifier in malware classification. The classifier tries to distinguish the real malware and fake one so that it can classify them accordingly. Although exhibiting many success factors, the generative adversarial network needs to be reinforced with another classifier for malware detection. This will ensure that the results are accurate as they can be compared to other classifiers reducing error rates.

3. METHODOLOGY

The generative adversarial network uses game theory and jointly combines with the unsupervised approach to train a particular model of our interest. The model is trained iteratively to improve the output of each model. GAN tries to train samples while the discriminative models try to match the real samples and fake samples.

In this paper, we apply GAN to analyze our malware from various samples to use them in section four of our research. Based on the previous review, GAN has successfully done important jobs in image generation tasks. In earlier days, though the work of GANs received a lot of praise, it had some shortcomings, such as it was hard to train, and there lacked association between sample quality and convergence of the loss function

3.1 Initialization Strategies

3.1.1 Using particle swarm optimizer strategy

The strategy was advanced by Kennedy &Eberhart [7], which uses concepts of flocking birds to explain the behavior of the particles. The presence of swarms makes it appear as a candidate solution to the optimization problems. To achieve the best optimal solution, the vectors need to roam through many times in the space area.

3.1.2 Uniform random initialization

The starting position of the particles is achieved by using Uniform distribution in the following manner: **x~ U (xmin, xmax) d** where **x** represents the search space dimensionality and **xmax** denotes upper limits while **xmin** denotes lower search space limits. To generate the position of the particle while being generated using a pseudo-random number generator.

3.1.3 Sobol sequences

Sobol sequence has improved particle swarm optimization performance by applying a quasi-random number generator [10]. This will help to generate the initial position of the particle.

This kind of random number generator exhibits some low discrepancy; at any particular time and is given a set of points. The importance of this generator does not produce clusters or gaps, hence allowing the swarm to be evenly distributed inside the search space, giving them better coverage to provide a better implementation of the Sobol sequence[6].

3.1.4 Centroidal voronoi tesselations

This strategy divides the subspace into smaller spaces called the cells. The interesting point is that those small cells should be uniformly distributed because they are of the same size. The cells are implemented by the use of a set of generators. Closer generators produce cells at all points in the search. In this way, we will partition search subspace into corresponding subspaces and place it close to the nearest generator.

3.1.5 Non-linear simplex method

The non-simplex form was attributed to Nelder & Mead [14] as the Minimization function technique. Theorem demonstrated that if a figure has n-dimensional simplex, then its vertices are equal to n+1.

3.2 Pbest and gbest update mechanism

The influence of a swarm's reaction in the process of evolution all depends on the shareable information of PSO. Using a traditional approach to update the particle depends much on fitness value, which is selecting attributes when classifying some characteristics. We say particle has an updated position if the current situation is better than the fitness of the previous location.

Traditional particle updating mechanisms exhibit a number of challenges while dealing with particles. A better feature is described as when the classification performance of the particle's new position is considered as the same as where the number of attributes is few. In the traditional approach, where classification is the same, pbest is not likely to be updated.

To address this limitation, a new pbest and gbest updating mechanism has been proposed to take care of it. This new mechanism classification anticipates the first thing to be considered in the feature subset is the classification performance; however, the feature also needs to be considered. In particle update, PSO takes place in two dimensions.

(i) If the performance of classification of the particle's current position is better than pbest, then it is anticipated that pbest will get replaced and occupy the current position. For this situation, the number of attributes will be assumed.

(ii) If the performance of classification of the particle's current position equals pbest and the feature attributes are few, the previous pbest will occupy the new position.

Updates of other particles take place by comparing gbest with pbest in regard to the close particles. The inclusion of roman ii in updating the mechanism will be okay and will avoid some challenges associated with the traditional updating mechanism of PSO.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 7, Issue 2, pp: (23-31), Month: October 2020 – March 2021, Available at: www.paperpublications.org

The mechanism intentionally has the tendency of choosing pbest or gbest as the best attribute subset resulting from either better classification performance or similar class performance with a smaller number of attributes. This assisted various algorithms in eliminating some unnecessary features and making a data subset with prolific classification performance. Fewer feature attributes appear to be the leader of all particles and the swarm at large.

3.3 Representation of solutions

In the solution generation stage, we evaluate each algorithm accordingly. The research uses a Sobol sequence strategy on the line that cuts across the Centre of search space. Sobol will be used in this research to act as a benchmark function across the Centre of search.

The benchmark functions were made in a manner such that the initialization strategy in solution representation will not be biased. The PSO was applied in the generation of representation of solution by generating updates close to the topology. The pbest or gbest were to update them on the condition they had to remain within the search space. This ensures swarm particles are maintained inside space.

3.4 Candidate solution generation strategies (Csgs)

The candidate's solution generation strategy is achieved when we generate some dimensional points in the search space. Candidates are generated by selecting a random point on a specific line, assuming that the line must pass across the search space Centre. The direction of the line is determined through a linear combination of several vectors in the space.

3.5 Diagram of the proposed method

The use of swarm initialization techniques for sure will tend to initialize the swarm in the particular subspace of the more prominent search space. What happens in particle swarm optimization not only swarms move anyhow from the initial subspace but also make swarm exploit the required space. But should focus on exploring the expanse search space.

4. EXPERIMENTS AND DISCUSSION

This chapter introduces various experiments that need to be done to discern how malware is embedded in our mobile devices and analyze them quantitatively. The experiments were carried out using Tensor Flow and Keras, machine learning libraries for our stegomalware dataset. We analyzed the data below from the permission of android manifest files of APK files from different devices.

Table 1: Illustration of the various network permissions

Manifest permission	Ranks
Manifest permission to allow the user to read the SMS(READ_SMS)	1.915
Permission for complete boot (RECEIVER_BOOT_COMPLETE	0. 0001
Manifest permission for sending messages (SEND_SMS)	1.874
permission for reading phone state (READ_PHONE_STATE)	0.791
WIFI Access states permission for the android operating system (ACCESS_WIFI_STATE)	0.209
Permission for making a call (CALL_PHONE)	0..903

Android permission refers to authentication required by the device to access a certain service. The experimental data on manifest permissions above illustrate malicious payloads in terms of rank. The rank means that the bigger the number, the higher the number of malware detected by our model(GAN). The read SMS permission (android.permission.read_sms) was used by various programmers of malware to read and access people's devices. More importantly, send SMS (android.permission.send_sms) was another avenue used by the assailant to send harmful data that triggered accessing the owner device without owner know-how and launch an attack from a remote point. Android permission such as boot had a low prevalence of low malware caseload, meaning attackers hardly used them.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 7, Issue 2, pp: (23-31), Month: October 2020 – March 2021, Available at: www.paperpublications.org

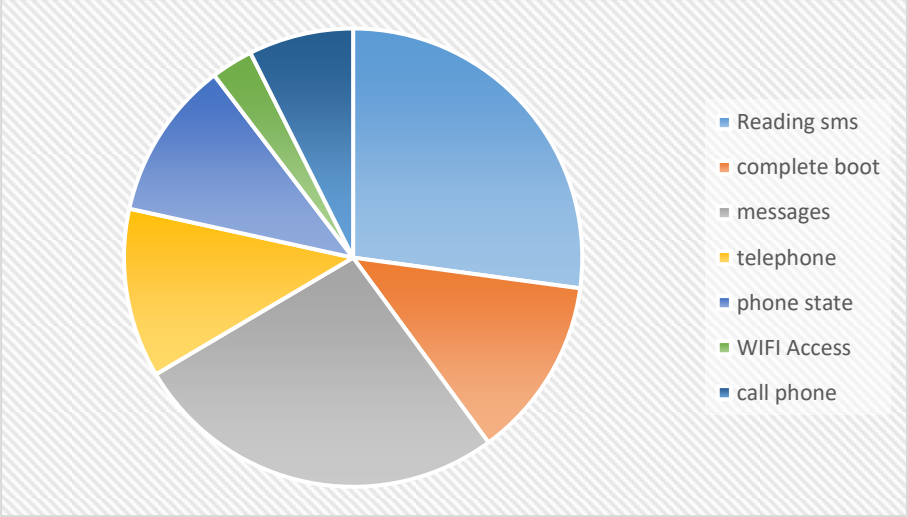

Figure 1: Chart of the various permissions

The pie chart above shows that digital assailants pronely used messaging applications to execute malicious code in the process of controlling our devices. The Android operating system was observed to be low in the ranks, perhaps because the android system is more secure. The research establishes that a manifest file was targeted more by a digital assailant. That is why we carried out a vigorous experiment using our generative adversarial networks on how stegomalware are embedded in our APK files. Steganalysis of this application demonstrated that mobile applications needed to be programmed with level security to avoid future problems. Another recommendation all applications, before published in the app store, should be decomposed, and the integrity of the application needs to be checked to ensure it meets the quality.

4.1 Parameter settings

Table 2: Parameter settings

Parameter	value	Description
Start size	21	The perceived agent will perform a uniform random policy for this number of steps before learning.
Buffer size	500000	Buffer capacity memory
Size of minibatch	16	Number of training cases
interval	10	The frequency at which the network is updated
Discounted factor	1	The discount factory is used in the Learning model.
Initial value x	**1**	The initial value for greedy exploration
Final value y	**0.3**	The final value in the greedy exploration
Test interval	10	Steps carried out for agent evaluation

4.2 Data Set Description

The malicious files used in the experiment were classified either as malicious or not harmful. Those executable program from the Android phones directories were classified as Android folders and their total in numbers was 10153 and those containing malicious dataset runs up to a total of 12361. In order to achieve stability and robustness of the repetitive experiments all the dataset under trial were parsed and the set which was not realistic was analyzed and filtered out.

After the experiments the following samples remained were 12145 and 9056. However, initially, the data used in the experiment was identified using its own original characteristics and the labels accorded unto them. So, they are required to be converted into vectors before sending them to the Generative Adversarial Network model.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 7, Issue 2, pp: (23-31), Month: October 2020 – March 2021, Available at: www.paperpublications.org

In this paper we shall make use of python programming language while performing machine learning. The android directory files will specifically be analyzed using a python toolkit. The python language was used to write the script, which will actually be used to do extraction and identification of characteristic of a particular sample.

The analyzed sample by the model is generated as a 1 x 204-dimensional vector. Tensor flow will be used to extract the desired selection features, the ideal process of extraction will be as follows:

(i) Convert Android asset images to hexadecimals files that can generate sequence of byte

(ii) Handle hexadecimal file using slide window method to produce selected files.

(iii) To select the desired features, we use DF criterion to get the feature with the highest DF values. The python script eventually has the possibility of generating a very large dimensional vector.

(iv) The joint vector is given a label, for stegomalware it is denoted with 1 while for non-malware is denoted using 0.

(v) Finally, for purposes of training, we store the file which possess original features in spreadsheet format (CSV format).

4.3 Convergence performance on the two methods in the training sets

In this experiment, we made use of steganalysis to generate an image by using Generative adversarial networks. The input is denoted as P(z) for a fixed value. We tested experiments 1 to 3 for stegomalware, where we determined the convergence performance of three data experiments to ascertain the classification accuracy of our model, and the following results were obtained.

Table 3: Trained data set of stegomalware

Experimental conditions	Classification Accuracy
Data experiments 1	0.85
Data Experiments 2	0.78
Data experiments 3	0.74

The trained data set produced the above classification accuracy from the three experiments so that we compare our data with another classifier to ascertain its accuracy viability.

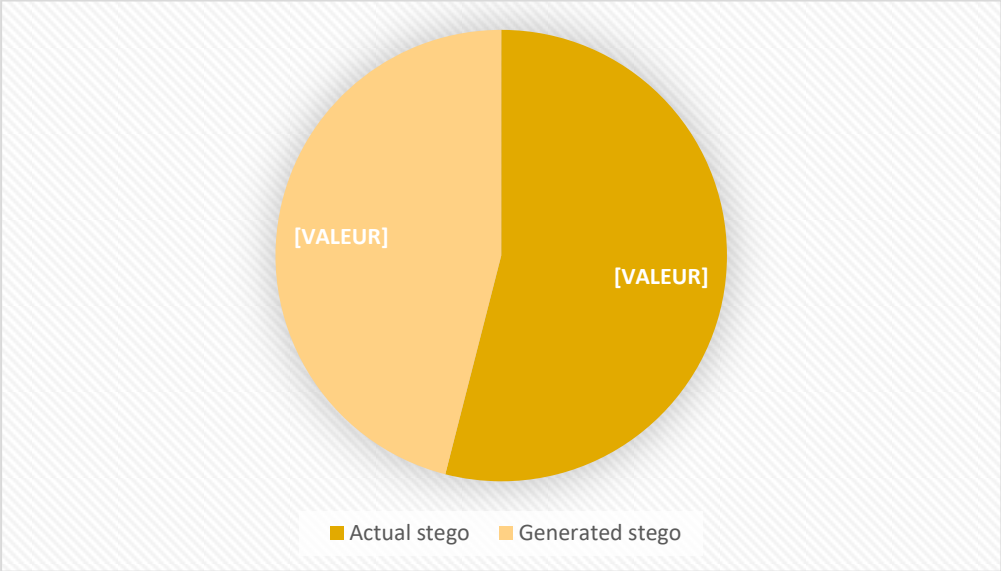

Figure 2: Chart showing the accuracy Steganalysis of trained using SSGANs

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 7, Issue 2, pp: (23-31), Month: October 2020 – March 2021, Available at: www.paperpublications.org

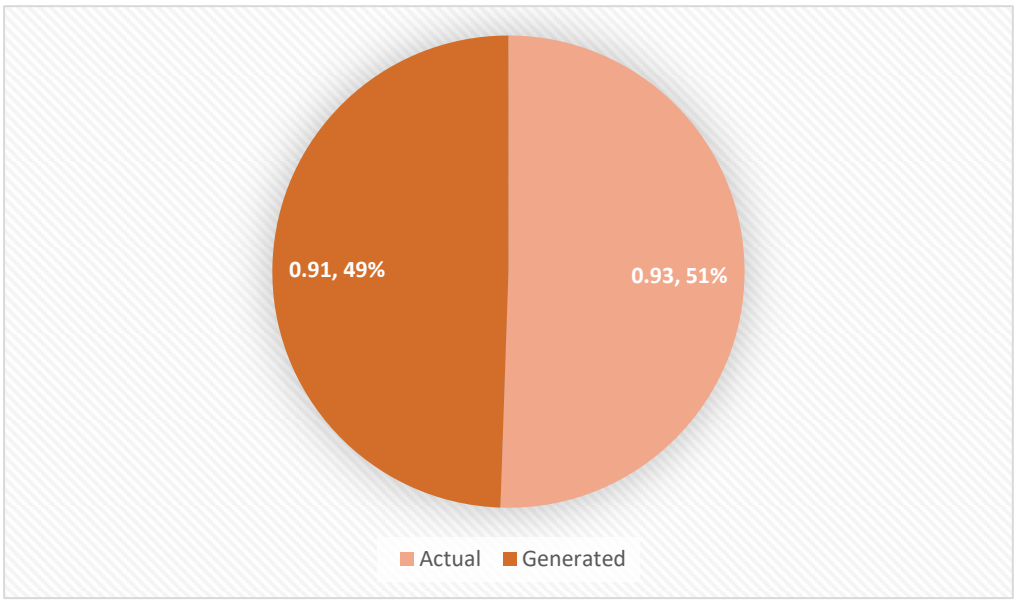

Figure 3: Chart showing the accuracy Steganalysis of trained using SSGANs

5. EXPERIMENTAL RESULTS AND ANALYSIS

The size of each batch of training was set to 150 and equivocally dimension set to 150.The stopwatch was stopped after every 130seconds, repetitive training gave us dataset of size 5030 from suspected malicious application.

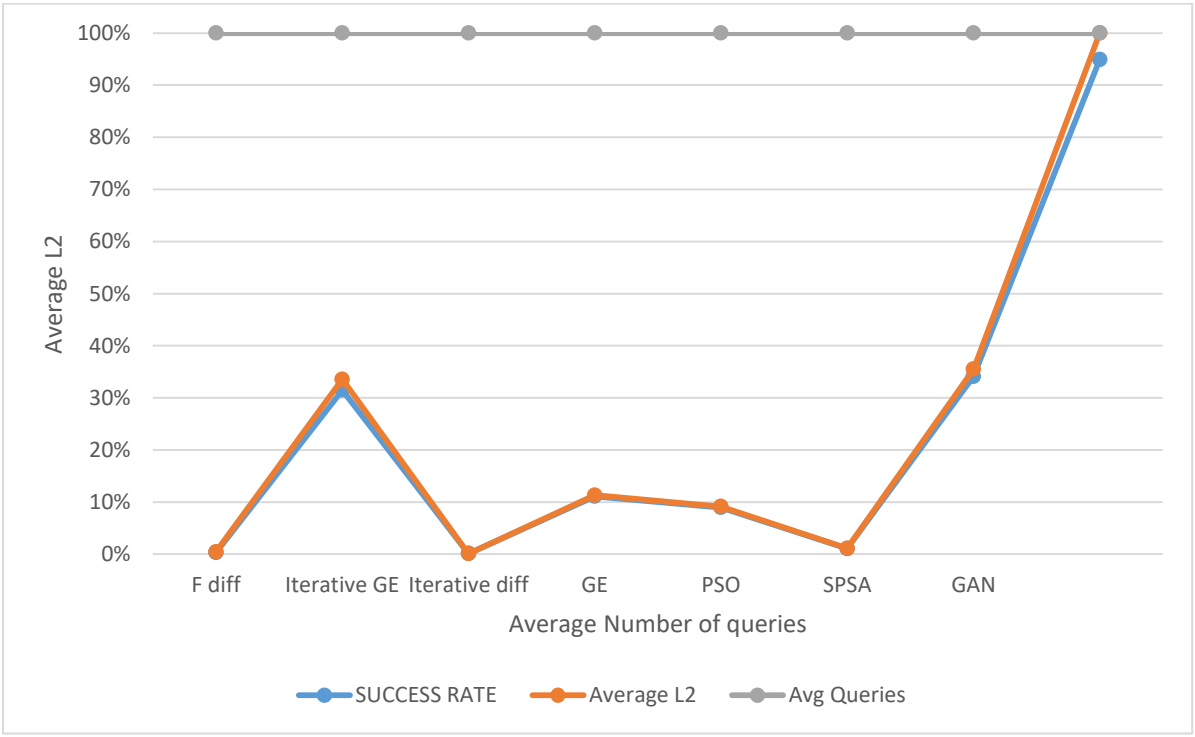

Figure 4: Line graph representing the possibility attacker attacking various classifiers

Based on the data we have from various network permissions, it is evident there is the possibility of malware being embedded into our mobile application by use of manifest file permission, where an attacker embeds the malicious code into the manifest file. Our classifier (GAN) for checking harmful application distinguished applications with malware; hence the ranks on the following table:

Table 5: Comparison of GAN with other classifiers

Accuracy / Classifier	5	6	7	8
KNN	95.21%	97.69%	96.19%	98.50%
DECISION TREE	96.47%	96.47%	96.02%	97.55%
R. FOREST	96.02%	95.95%	98.13%	97.15%
NAÏVE BAYES	95.91%	94.30%	96.33%	99.23%
GAN	**95.32%**	**96.85%**	**99.30%**	99.12%

Various methods for classifying and training data were accounted for so as to establish which method produced a better result and compare it with our General adversarial network. NAÏVE BAYES demonstrated similarity in classification with GAN; hence Table 5 illustrates the classification accuracy of experiments carried out using GAN. The parameter was set with a high level of magnitude to achieve dignified and significant results. The methodology of the generative adversarial network model should be adopted to any mobile application to test it genuinely before being published in the play store.

6. CONCLUSION

In this paper, we discussed the implementation of Steganalysis on Android applications using General Adversarial Networks. A major contribution of this paper is the procedure of analyzing the applications, decoding them to understand their behavior then begin to apply GAN on the steganographic applications. We examined the information from the permission of android show records of APK documents from various gadgets and discovered that most malwares find their way through there. Thereby, we inspected a range of methods not only GAN and applied our data to several methods which include Naïves Bayes, R. Forest, Decision Trees, KNN. As we predicted, GAN came out to be the most precise of all. For long Steganographic applications have been a huge problem in the android field but now Deep learning has long proven to be very efficient when it come to its application to Cybersecurity. Nonetheless, the algorithm can be enhanced so that the differentiation between stego-malwares and usual applications can be more explicit.

REFERENCES

[1] (Chen. Y,2016) Efficient ant colony optimization for image feature selection.

[2] (D. P, 2014). A method for stochastic optimization.'[Online]. Available: https://arxiv.org/abs/1412.6980

[3] H. Park, Y. Yoo, and N. Kwak. (2018). ``MC-GAN: Multi-conditional generative adversarial network for image synthesis." [Online]. Available: https://arxiv.org/abs/1805.01123

[4] Honglak Lee. Generative adversarial text for image synthesis. arXiv preprint arXiv:1605.05396, 2016.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. pp. 2672–2680, 2014.

[6] Mielikainen, Jarno. LSB matching revisited. IEEE signal processing letters, 13(5):285–287, 2015.

[7] Kennedy & Eberhart (2015) Improved binary particle swarm optimization using catfish effect for feature selection, Expert Systems with Applications.

[8] M.A. Hall, Correlation-based feature selection for machine learning, The University of Waikato, 2018

[9] Nelder & Mead (2014), Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, IEEE Transactions on Pattern Analysis and Machine Intelligence.

[10] Schoemann and Engelbrecht (2014) A new local search-based hybrid genetic algorithm for feature selection, Neurocomputing.

[11] (Stolfo.M, 2018) Detection of LSB steganography via sample pair analysis, in Proc. Int.Workshop

[12] Shi, Y. Q., Sutthiwan, P., and Chen, L., "Textural features for steganalysis," in [Information Hiding], 63–77, Springer (2013).

[13] T. Qiu, R. Qiao, and D. Wu, ``EABS: An event-aware backpressure scheduling scheme for emergency Internet of Things," IEEE Trans. Mobile Comput., vol. 17, no. 1, pp. 72_84, Jan. 2018.

[14] Joe and Kuo.M (2010) Aspider-Web-like transmission mechanism for emergency data in vehicular ad hoc networks," IEEE Trans. Veh. Technol., vol. 67, no. 9, pp. 8682_8694, Sep. 2018.

[15] T. Qiu, H. Wang, K. Li, H. Ning, A. K. Sangaiah, and B. Chen, ``SIGMM: A novel machine learning algorithm for spammer identification in industrial mobile cloud computing," IEEE Trans. Ind. Information., vol. 15, no. 4, pp. 2349_2359, Apr. 2019. doi: 10.1109/TII.2018.2799907.

[16] R. Meng, Q. Cui, and C. Yuan, ``A survey of image information hiding algorithms based on deep learning," Comput. Model. Eng. Sci., vol. 117, no. 3, pp. 425_454, 2018.

[17] T. Salimans et al., ``Improved techniques for training GANs," in Proc. Adv.Neural Inf. Process. Syst., 2016, pp. 2234_2242.

[18] Parsopoulos.M, 2015, GANs trained by a two time-scale renewal rule converge to local equilibrium.

[19] (Qi. P, 2017) Time-location-frequency'-aware Internet of Things service selection based on historical records.

[20] (Penny. B, 2018) Using high-dimensional image models to perform highly undetectable steganography.