Perspectives in Communication, Embedded-Systems and Signal-Processing (PiCES) – An International Journal
ISSN: 2566-932X, Vol. 4, Issue 4, July 2020
Part of the Proceedings of the 1st All India Paper writing Competition on Emerging Research - PaCER 2020

Detection of Malware in Cloud Computing using Sparse Autoencoders

Doddi Srilatha
School of Computing & Information Technology,
REVA University, Bengaluru, India,
doddisrilatha@gmail.com

Gopal Krishna Shyam
School of Computing & Information Technology,
REVA University, Bengaluru, India,
gopalkrishnashyam@reva.edu.in

Abstract: Cloud computing is a most inclining world view that gives conveyance of physical and intelligent assets as administrations over the Internet on request. Numerous malwares focus at customized personal computers (PCs) in cloud condition to obtain secret data and obstacle the cloud appropriation by organizations and clients. In this paper, we consider a way to deal with shielding the cloud from being assaulted by nearby PCs. Because of this issue, in view of the Windows Application Programming Interface (API) calls are removed from the Portable Executable (PE) files, we propose a novel Behavior-based Machine Learning Framework (BMLF) using Sparse Autoencoder (SpAE) which is worked in cloud stage for detection of malware. In the proposed BMLF, first we develop conduct graphs to give effective data of malware practices utilizing extricated through. We at that point utilize SpAEs for removing elevated level highlights from conduct graphs. The layers of SpAEs are embedded in a steady progression and the last layer is associated with an additional classifier. The design of SpAEs is 5,000-2,000-1000. The experimental results show that the proposed BMLF yields the semantics of more elevated level noxious practices and increments the normal detection accuracy by 2%.

Keywords: Cloud Computing; Malware Detection; Machine Learning; Behavior based Machine Learning Framework (BMLF); Sparse Autoencoders (SpAEs)

I. INTRODUCTION

At present, cloud computing is a innovative technology that provisions physical and logical resources as services to the end users for necessary operation of data processing and management tasks. Organizations such as Amazon, Google, IBM, Microsoft, Facebook and Yahoo! are investing on data centers to offer cloud services that aim to utilize virtualization capabilities[1]. Cloud frameworks pull in numerous clients with its attractive features such as flexibility to access the services, easy to use, pay as per use, simple registration process etc. Undoubtedly, this increased flexibility in the cloud encounters a number of security attacks.

As mentioned in [2] cloud service models suffer from a large number of security threats that break the security enclosed within the inherent features of cloud such as rapid elasticity, flexibility to access and service transparency. Hence, a security is great challenging issue that exists in cloud is associated with the efficient detection of abnormal behaviors such as perpetrate web fraud, steal personal information, and for many other abuse and nefarious activities caused either by legitimate or malicious intent. In particular, the malware detection is a most challenging issue since malwares results in starting point for launching of Distributed Denial of Service (DDoS) assaults in cloud [3].

Malware is a trend that tends to increase and will remain as the greatest security threat faced by computer users. Symantec report [4] demonstrated that new malwares have developed by 36% before in 2015 with all out examples surpassing 430 million. Everyday lives can be caused risk because of exponential development of malware threats.

Customized PCs acquire a great deal of assaults cloud condition. Malware assaults PCs and utilizations, the tainted PCs to assault other associated gadgets in cloud condition. For example, Mirai can taint windows and use the hosts to contaminate different gadgets. The tainted windows can take private data and change the affected systems into a botnet to dispatch another DDoS assault. Numerous existing customized PCs' malware assaults may likewise reach out to other cloud. Lamentably, there are no perfect answers for keep away from Mirai and other cloud dangers. One methodology intends to debilitate these dangers by ensuring the security of conventional PCs in cloud condition.

Because of rapid development of malware in the information technology, the knowledge of new or unknown malware detection based on machine learning methods is an important challenge for researchers.

Malware can be detected basically with two techniques. Signature based and behavior based detection techniques[5]. The popular anti-malware software's such as Kaspersky, Symantec, and Comodo etc. use signature-based detection in order to guard legal users from the hackers, The signatures may be operation codes (opcode), the sequences of byte codes, system calls etc. However, this method can be easily escaped by hacking through

encryption, polymorphism and obfuscation techniques such. This technique works effective in detecting well-known malicious objects However, they have two serious shortcomings. First, they can't detect new and quickly changing nature of portable malware, and second, they require an enormous amount of time from skilled security experts to develop the signature database. To address these challenges, there's a new kind of malware detection tools based on modeling normal behavior

Behavior-based detection systems do not test the programs against a list of known attacks. Instead of, they observe all unknown programs of malicious behaviors. This kind of detection works effectively against malware attacks, even a zero-day attack.

The procedure for detecting and finding a malware can be done by two types of analysis namely, static and dynamic analysis.

The static analysis extracts features directly from the byte n-grams, string signatures, opcode frequency distribution and control flow graph. The benefit of this analysis is that it could follow all probable execution paths and it consumes fewer resources, but it is sensible to encryption, garbage code insertion, and code obfuscation techniques.

Dynamic analysis refers to executing malicious samples in a virtual or separate environment (e.g., a sandbox), and then capture the various behaviors of the samples during the execution. Dynamic features are the behavior of the sample execution, instruction traces, process monitoring, network monitoring, registry monitoring, function call monitoring, system change detection, information flow tracking, the creation and destruction of processes. Since the malware malicious behaviors during runtime can't be obscured, dynamic features provide more reasonable information than the static features. Dynamic analysis is used to identify unknown samples and thus reduces the costs of analysis.

Numerous machine learning techniques such as Support Vector Machine (SVM), Decision Tree (DT), K-Nearest Neighbor (KNN), and Invariant Miner(IM) are normally utilized in malware detection[6][7]. Existing Artificial Intelligence (AI) calculations gain the conduct includes conceivably from the malware. Unfortunately, many AI calculations execution dependent on the removed highlights exactness, thus, it is hard to decoct conduct includes definitively for improving malware recognition execution. In addition, requires high skills to process feat extraction. In this way, existing AI calculations are as yet unfit to identify malware efficiently.

AI is a part of software engineering that endeavors to gain significant level highlights straight forwardly from the first information. AI is effective to examine elevated level highlights of tests by methods for multilayer deep learning, and it has been generally utilized in picture handling, visual acknowledgment, object location, computer vision and so on [8], [9].

This paper acquaints a strategy with shield cloud from being assaulted by nearby PCs. In this paper, we assemble a Behavior-based Machine Learning Framework (BMLF), which uses Sparse Autoencoders (SpAEs) and customary AI calculations for malware detection. SpAE is one of the AI models that comprises of numerous layers of inadequate Autoencoders [10][11]. We use SpAEs model to separate elevated level highlights from conduct diagrams and afterwards do order by the additional classifiers such as DT, IM, and SVM. The proposed BMLF is executed in cloud stage as showed in Figure 1.

The major contributions in this paper are:

1. We develop a novel Behavior-based Machine Learning Framework (called BMLF) by searching SpAEs model with conduct graphs for malware detection. The proposed BMLF means to get further semantics in conduct graphs instead of API call groupings (e.g., n-gram) and
2. In proposed BMLF, we explore an AI model of SpAEs to consequently procure significant level portrayals of malware practices. The experimental results show that proposed technique decocts unique highlights seriously and help to achieve high accuracy in malware detection.

The remaining sections of the paper are as follows. Section 2 deals with related works on malware detection. Section 3 discusses about proposed malware detection framework. Section 4 presents results of proposed malware detection framework and comparison of results with existing techniques and finally section 5 deals with conclusion.

II. RELATED WORK

The malicious behavior of the insiders generally creates the critical problems throughout the systems [12]. Besides, the intrusion detections were found to suffer from numerous factors such as less accuracy rate and more false positives and dynamic threat scenarios. These confronts were found to be rare while are considered to be extremely important as these parameters creates critical problems as the malicious behavior of the insiders try to evolve the computing technology to avoid being caught [13] .

Besides, the motivation of the intrusion in the cloud computing services were comprised of complex contextual combinations of activities which are either authorized or legitimate as these are performed in distinct combinations and are found to be diverse based on the context [14].

Therefore, from the aforementioned it was observed that considerable information is required to discriminate between the intrusions and legitimate activities. Thus, appropriate knowledge concerning the context explaining the anomalous activities in terms of user's behavior is necessary and should incorporate the information

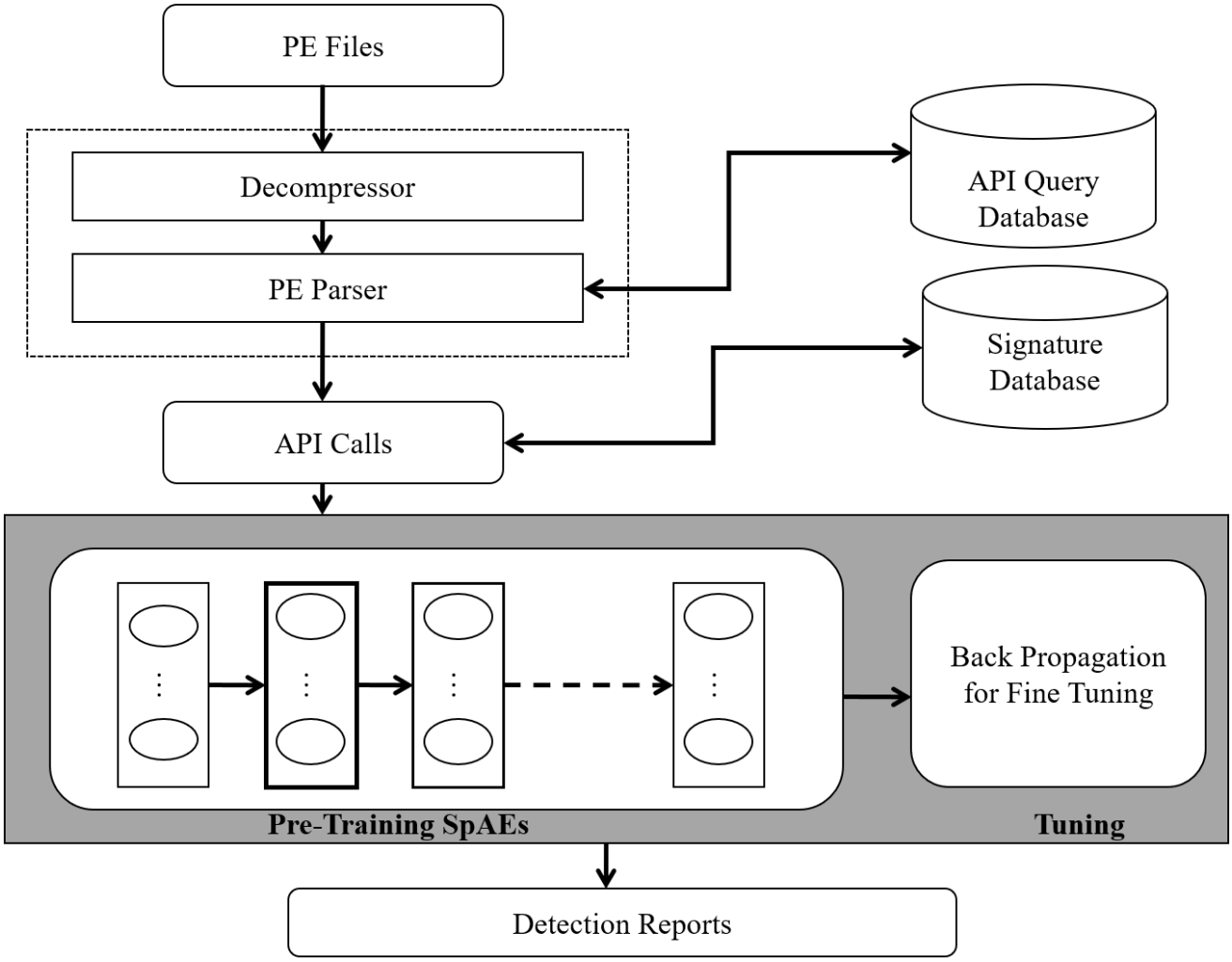

Fig 1. Architecture of SpAE

regarding the employee personnel records, charts of project allotment, working hours and locations. Furthermore, from the recent study it was noticed that the information of individuals was protected due to the ethical considerations and privacy needs from disclosure.

In this manner, it is vital that the engineers ought to guarantee applications with the end goal that the protected cloud could be executed notwithstanding creating and planning of a safe programming application. The information took care of by the cloud should be ensured by considering security instruments to such an extent that it will have the option to give information trustworthiness, secrecy, and power in the trade or correspondence inside the cloud site to clients and the other way around. As the issues were found to rudimentary it is important to consider the safety efforts all through the formative periods of the product. Furthermore, the assurance of the cloud against noxious or non-vindictive assaults has become an essential worry as these were conceivably critical dependent on the financial outcomes concerning the dangers engaged with the distributed computing.

The intrusion detection is considered as a challenging issue in cloud to identify illegitimate activities. Intrusion detection is the distinguishing proof of a pernicious conduct against a framework. The key test is to dependably distinguish legal users and attackers. There are two customary methods utilized by IDS, these are knowledge-based IDS[14] and behavior-based IDS[15].

In knowledge-based IDS, search for information of attacks based on knowledge accumulated from known attacks. These systems stores attack descriptions, typically a signature that can be matched to attack manifestations. These systems score high accuracy, and low false positive rate, but cannot detect unknown attacks (or new or zero-day attacks).

In behavior-based IDS, search for deviations from a model of normal behavior based upon observations of a system during a known normal state. These systems are designed to meet new requirements such as identifying previously unknown attacks. The use of these systems is depending on the false alarm rate (FAR) that they output, which is based on false-positive count (i.e. raising a

notification for legitimate traffic) and false-negative count (i.e. the failure to notify an intrusion attacks).

Pannu et al. [16] presented anomaly detection that uses SVMs for validating cloud dependability assurance. An unlabelled monitoring datasets are used, which are processed by a SVM algorithm.

Han et al. [17] proposed a defense method against co-resident attacks on virtual machines in cloud computing and prevention of these attacks by applying clustering and constructed multiple semi-supervised SVMs. This approach distinguishes attackers and normal users, and classifies all users into three classes such as low, medium, and high risk by using semi-supervised learning techniques. The attacker's overall cost is increased dramatically by one to two orders of magnitude due to attackers are forced to behave similar to legal users.

Sarat Akasapu [18] implemented detection of DoS attacks through feature-based selection method to choose the subset of significant features on KDD dataset then the selected features are given as an input to the classification models such as RF, DT, KNN, and NB etc. and also evaluated the accuracy results of these classifiers.

Another a huge number of similar defense mechanisms against malwares which are based on shallow and deep learning techniques are summarized as follows:

Sang Ni et al. [19] proposed a malware classification using Simhash and convolutional neural network (MCSC) algorithm that uses static features such as opcode sequence and locality-sensitive hashing (LSH) using major block selection then convert the dis-assembled malware codes into gray code images based on simhash and then identifies malwares by a CNN. The MCSC performs better in malware categorization, even at small sample size.

Anderson et al. [20] used Support Vector Machine to detect the malware based on graphs constructed from instruction traces, n-grams, markov-chain, spectral kernels, and Gaussian kernel demonstrated good results, but with a high complexity cost.

Angelos et al. [21] introduced an Ensemble Empirical Mode Decomposition (E-EMD) algorithm for malware detection in the cloud based on individual network (or system-specific) features for every virtual machine (VM) that runs on a given physical host and also captured volume-based network traffic features from every VMs' network The E-EMD gives a 90% accuracy.

Hardy et al. [8] proposed a stacked Autoencoders (SAEs) deep learning model based on API calls extracted from the PE Files. This system is performed on Comodo Cloud Security Center dataset. This model uses a greedy layer wise training for unsupervised automatic feature learning; thereafter follow supervised learning for fine tuning the parameters.

Galal et.al [5] used a behavior-based malware detection system using support vector machine, decision tree, random forests, and classifiers based on the API traces (high level features) of malwares, that were captured in controlled virtual environment and experimental results express that the decision trees perform better and give 97.19 % accuracy.

Zhenlong et al. [22], assemble an android malware location motor (Droid Detector) in view of Deep Belief Networks (DBN), yet the technique can accomplish 96.8% accuracy by breaking down the highlights of required consents, delicate APIs, and dynamic practices (13 application activities).

III. THE PROPOSED MALWARE DETECTION FRAMEWORK

This section highlights the proposed Behavior-Based Machine Learning Framework (BMLF). The proposed BMLF comprises of two modules: conduct graphs development and SpAE-based malware detection.

A. Framework Overview

The proposed framework is made out of neighborhood PCs and cloud stage (CS) module. The proposed BMLF is actualized in CS's identifiers, which is the primary module for development of conduct graphs and malware detection. In the proposed BMLF, each program is spoken to a conduct graphs which consists of numerous API calls. Programming interface calls chart coordinates API calls with working framework assets. After the conduct graphs are developed, CS changes the conduct highlights into two fold vectors and afterward utilizes these vectors as contribution to the SpAEs. There are 3 layers in the proposed SpAEs model. The engineering of the SpAEs is the last output layer's information are utilized as the yield of the additional classifier (i.e., DT, SVM, and IM). The point of the proposed BMLF is to get familiar with the semantics of the elevated level malicious practices and recognize malware viably.

CS gives a boundless extra room. Identifiers in CS are answerable for recognizing getting to information or records got from Local Computer (LC). For getting to data, CS develops conduct graphs and afterward changes the API calls charts into paired vectors which are utilized as contribution to SpAEs models for malware detection. For suspicious records, CS executes tests in Cuckoo Sandbox and afterward extricates API calls from sandbox's checking documents. A short time later, CS controls the checking documents same route as the filtering data. After identification, CS offers input to LC. The developments of conduct graphs are shown in Figure 2.

As to develop a deep learning system, we apply SpAEs model which comprises various layers of Sparse Autoencoder to separate highlights [23][24].An Autoencoder (AE) has three layers namely: input layer, hidden layer, and output layer. An Autoencoder attempts to utilize the encoder to describe and feed the information into a hidden layer and utilize the decoder to outline hidden layer's information into a output layer, so as the output is

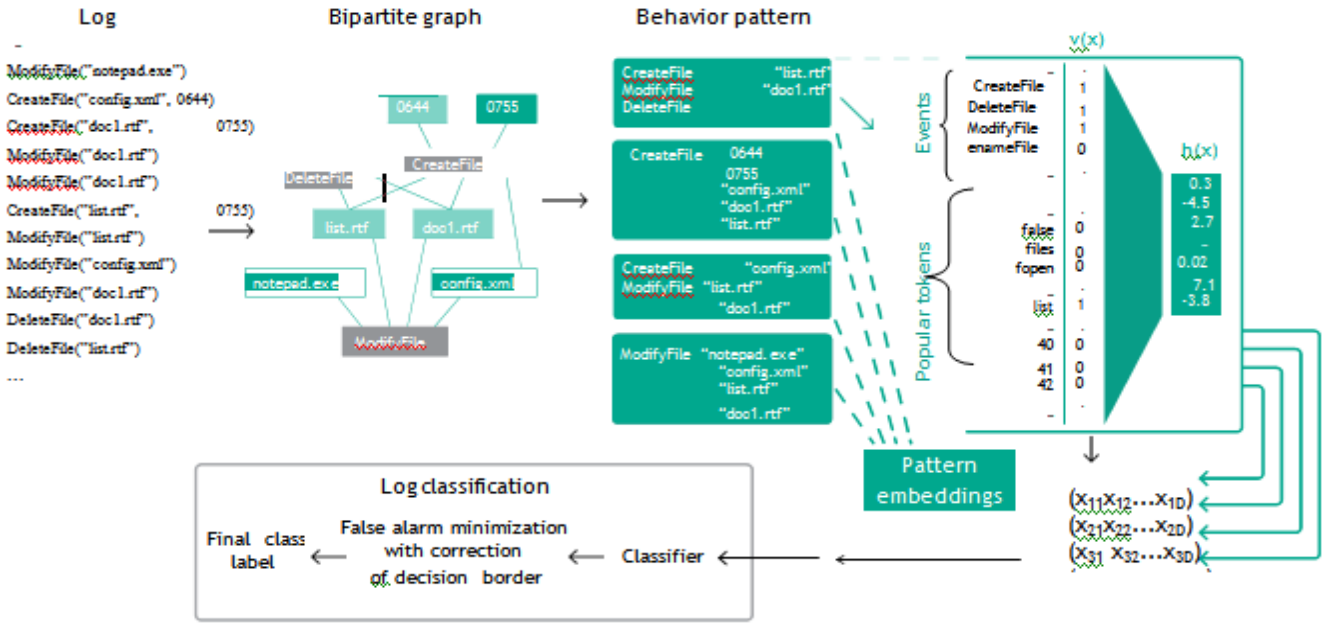

Fig 2. Developments of conduct graphs

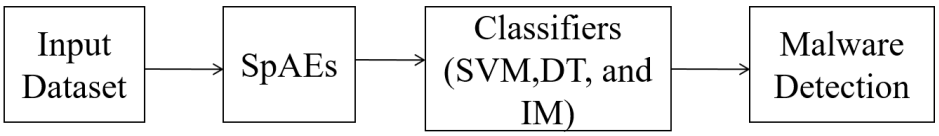

Fig 3. Sparse Auto Encoders-Based Malware Detection System

compressed form of input information. To put it plainly, an AE endeavors to get familiar with the scanty portrayal of the input and reproduce the information.

B. Sparse Auto Encoders-Based Malware Detection

***Algorithm 1**: Auto Encoder preparation in malware detection*

Input: Dataset D with n preparing tests:

$D_i =< R_i, C_i >$, where $i \in \{1, ..., n\}$

Output: Parameter set $\theta = \{W, b\}$

Introduce (W, b);

While computing error E hasn't met or the assigned emphasis hasn't came to **do**

For each info D_i **do**

Compute activation Y_i at the hidden layer, and get an output Z_i at the output layer;

End

Compute activation error E(D,z); Back propagate the error through the net and update parameter set $\theta = \{W, b\}$;

End

The hidden layers are prepared individually from bottom to top. In the proposed SpAEs model, the primary layer gets 1,25,164-sized unique information and prepares just as an AE. After finish of preparing in an AE, the hidden layer of 60,000-sized highlights created in the principal hidden layer is used as the contribution to another AE which is included top of the current AE. The new AE achieves the current AE's yield as its information and prepared similarly. By and large, the concealed layer's information are utilized as the contribution of the layer and prepared essentially as an AE. At last, the last hidden layer's output is the whole SpAEs model's output.

***Algorithm 2**: SpAE preparation in malware detection*

Input: Dataset D with n preparing tests:

$D_i =< R_i, C_i >$, where $i \in \{1, ..., n\}$

h: number of hidden layer; k_j :number of neurons for each layer, where $j \in \{1, ..., h\}$

Output: Parameter sets θ_s

For each layer ($L \in \{1, ..., h\}$) in SpAEs

Do Use Algorithm 1 to prepare the Auto Encoder at each layer;

End

Instate (W_{h+1}, b_{h+1}) at the classifier layer; compute the names for each preparation test D_i; Perform Back Propagation (BP) in an administered approach to tune the parameter sets θ_s all things considered.

At the point when all the preparation layers completed, the SpAEs model believers the 1, 25, 164-sized unique highlights into 100-sized elevated level highlights. These 100-sized significant level highlights are viewed as the new introductions of an executable program document. The proposed SpAEs model plans to decrease the quantity of the highlights and depict the highlights in a reduced elevated level articulation.

When the primary layer in line is pre-trained, it tends to be utilized as a contribution to the following AE. We adjust the profound neural system in the wake of being pre-trained in line and put last layer's initiation to the additional classifier to line. In line, speaks to the DT, speaks to KNN, speaks to NB, and speaks to SVM. Individually train the additional classifiers and yield the class mark (malware or favorable example).

IV. ASSESSMENT AND EXPERIMENT RESULTS

In this section, first we outline the dataset i.e. utilized for assessment and assessment strategy. To assess the viability of our strategy, we at that point look at the proposed BMLF (SpAE) with some shallow models which comprises of DT, IM, SVM and profound model SAE.

A. Dataset and Evaluation Method

We test the assessment with a dataset containing 1,25,164 instances, where 65,164 are malware instances, and the other 60,000 are benign instances. In this experiment, we utilize 10- fold cross-validation technique [25] in malware detection. In10- fold cross-validation technique, the first dataset is randomly separated into equivalent measured parts. For10- fold cross-validation technique, we utilize 6500 examples for preparing and 3250 examples for testing in each trial.

We assess the proposed malware detection technique by utilizing accuracy based on true positive (the positive example is effectively recognized as the positive example), true negative (the negative example is accurately distinguished as the negative example), false positive (the negative example is mistakenly distinguished as the positive example), and false negative (the positive example is erroneously distinguished as the negative example).

B. Analysis and Evaluation Results

We inspect the examinations in two viewpoints: shallow learning models and deep learning models. We conducted eight tests. The analyses incorporate some shallow learning models and Stacked Autoencoder (SAE) and SpAE-based deep learning models.

The shallow learning models select conduct includes by meaning data trade and choose certain properties and afterward utilize these highlights to anticipate the marks of the samples. Data gathering is utilized to signify data trade and choose certain properties of the element can bring to the framework. DT, IM, and SVM are used as Shallow learning models for evaluation.

We train deep learning models which incorporate the proposed SpAE-DT, SpAE-IM, and SpAE-SVM. Three hidden layers deep learning models executed on Keras. We feed 1,25,164 instances to SpAEs model and convert them to 1000-sized highlights. The cluster size for the profound inclining models is 2,000. The SpAE-based frameworks (SpAE-DT, SpAE-IM, and SpAE-SVM) are prepared with 100 epochs.

The assessment of the performance of malware detection technique is achieved with the employment of accuracy based on the number of True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). The results of the proposed technique with various hidden layers and number of neurons in each layer are shown in Table 2 and 3. The best execution of malware detection technique is that of the SpAE-DT model. In the proposed SpAE-DT model, is as high as 96%. The superior exhibitions were gotten from the SpAE-DT, SpAE-IM, and SpAE-SVM model, which demonstrate that the highlights gained from the SpAEs model assistance to improve the presentation contrasted and conventional order.

The results of proposed technique and other learning techniques are shown in table 4 and 5. The testing dataset results delineate their proposed profound learning model accomplishes accuracy with 95.5%. Our proposed BMLF dependent on SpAEs and conduct graphs accomplishes 96.6% accuracy in malware detection. It may be very well seen from Table 3 that our proposed SpAE-DT improves the accuracy in malware detection. It is significant for mining the deep semantic connections in conduct graphs.

Measure	Description
TP	Number of instances classified as malwares effectively
TN	Number of instances classified as benign effectively
FP	Number of instances classified as malwares erroneously
FN	Number of instances classified as s benign erroneously

Table 1. Performance Measures in Malware Detection

where, TP: true positive, TN: true negative, FP: false positive, and FN: false negative.

Hidden Layers	Neurons	TP	FP	TN	FN	Accuracy
2	[50 50]	21,859	1,434	21,066	641	0.9539
2	[100 100 100]	22,142	1,460	21,040	358	0.9596
3	[50 50 50]	22,110	1,295	21,205	390	0.9626
3	[100 100 100]	22,035	953	21,547	465	0.9685
4	[100 100 100]	22,150	1,178	21,322	350	0.9660
5	[100 100 100]	22,055	977	21,523	445	0.9684

Table 2. The Evaluation of Different SpAE-based Deep Neural Network Systems during Training

Hidden Layers	Neurons	TP	FP	TN	FN	Accuracy
2	[50 50]	2,368	161	2,339	132	0.9414
2	[100 100]	2,391	185	2,315	109	0.9412
3	[50 50 50]	2,396	170	2,330	104	0.9452
3	[100 100 100]	2,396	114	2,386	104	0.9564
4	[100 100 100]	2,408	147	2,353	92	0.9550
5	[100 100 100]	2,384	128	2,372	116	0.9512

Table 3. The Evaluation of Different SpAE-based Deep Neural Network Systems during Testing

Method	TP	FP	TN	FN	Accuracy
DT	21,338	1,781	20,719	1,162	0.9346
SVM	21,610	1,576	20,924	890	0.9452
IM	21,532	9,940	12,560	968	0.7576
SAE	21,630	1,560	20,940	870	0.9460
SpAE	22,035	953	21,547	465	0.9685

Table 4. Comparison between Shallow Learning and Deep Learning Methods during Training

Method	TP	FP	TN	FN	Accuracy
DT	2,264	163	2,337	236	0.9202
SVM	2,305	152	2,348	195	0.9306
IM	2,332	1,541	959	168	0.6582
SAE	2,357	292	2,208	143	0.9130
SpAE	2,396	114	2,386	104	0.9564

Table 5. Comparison between Shallow Learning and Deep Learning Methods during Training

V. CONCLUSION

In this paper, we proposed a novel Behavior-based Machine Learning Framework (BMLF) based on conduct graphs for malware detection in cloud environment. By adding shallow learning classifiers and Sparse Autoencoder, we get ideal malware detection. The experimental results show that SpAE-based models can learn further dynamic semantics highlights and help to improve the normal accuracy of the malware detection by 2%. We conclude that extra works in SpAE model can be applied in malware detection.

REFERENCES

[1] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud," ACM SIGCOMM Comput. Commun. Rev., vol. 39, no. 1, pp. 68–73, 2008.

[2] G. K. Shyam and S. Doddi, "Machine vs Non-Machine Learning Approaches to Cloud Security Solutions: A Survey.", Journal of Engineering Science and Technology Review, vol.12.no.3, pp.51-63, 2019.

[3] A. K. Marnerides, M. R. Watson, N. Shirazi, A. Mauthe, and D. Hutchison, "Malware analysis in cloud computing: Network and system characteristics," 2013 IEEE Globecom Work. GC Wkshps 2013, no. December, pp. 482–487, 2013.

[4] Symantec, "Internet security threat report," Netw. Secur., vol. 21, no. 2, pp. 1–3, 2016.

[5] H. S. Galal, Y. B. Mahdy, and M. A. Atiea, "Behavior-based features model for malware detection," J. Comput. Virol. Hacking Tech., vol. 12, no. 2, pp. 59–67, 2016.

[6] M. Fan et al., "Android malware familial classification and representative sample selection via frequent subgraph analysis," IEEE Trans. Inf. Forensics Secur., vol. 13, no. 8, pp. 1890–1905, 2018.

[7] Z. Lin, F. Xiao, Y. Sun, Y. Ma, C. C. Xing, and J. Huang, "A secure encryption-based malware detection system," KSII Trans. Internet Inf. Syst., vol. 12, no. 4, pp. 1799–1818, 2018.

[8] W. Hardy, L. Chen, S. Hou, Y. Ye, and X. Li, "DL4MD: A Deep Learning Framework for Intelligent Malware Detection," Proc. Int. Conf. Data Min., pp. 61–67, 2016.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9351, pp. 234–241, 2015.

[10] E. Protopapadakis, A. Voulodimos, A. Doulamis, N. Doulamis, D. Dres, and M. Bimpas, "Stacked Autoencoders for Outlier Detection in Over-the-Horizon Radar Signals," Comput. Intell. Neurosci., vol. 2017, no. i, 2017.

[11] L. Vareka and P. Mautner, "Stacked autoencoders for the P300 component detection," Front. Neurosci., vol. 11, no. MAY, pp. 1–9, 2017.

[12] O. M. Al-Jarrah, M. Al-Ayoub, and Y. Jararweh, "Hierarchical detection of insider attacks in cloud computing systems," Int. J. Inf. Comput. Secur., vol. 9, no. 1–2, pp. 85–99, 2017.

[13] S. Browne, W. Golden, and M. Lang, "Association for Information Systems AIS Electronic Library (AISeL) Contextualising the Insider Threat: A Mixed Method Study Recommended Citation Contextualising the Insider Threat: A Mixed Method Study," vol. 10, 2016.

[14] A. Carlin, M. Hammoudeh, and O. Aldabbas, "Intrusion Detection and Countermeasure of Virtual Cloud Systems - State of the Art and Current Challenges," Int. J. Adv. Comput. Sci. Appl., vol. 6, no. 6, pp. 1–15, 2015.

[15] J. Jang-Jaccard and S. Nepal, "A survey of emerging threats in cybersecurity," J. Comput. Syst. Sci., vol. 80, no. 5, pp. 973–993, 2014.

[16] H. S. Pannu, J. Liu, and S. Fu, "AAD: Adaptive anomaly detection system for cloud computing infrastructures," Proc. IEEE Symp. Reliab. Distrib. Syst., pp. 396–397, 2012.

[17] Y. Han, T. Alpcan, J. Chan, C. Leckie, and B. I. P. Rubinstein, "A Game Theoretical Approach to Defend Against Co-Resident Attacks in Cloud Computing : Preventing Co-Residence Using Semi-Supervised Learning," vol. 11, no. 3, pp. 556–570, 2016.

[18] S. Akasapu, "An Integrated Approach for detecting DDoS attacks in Cloud Computing," no. June, pp. 258–261, 2017.

[19] S. Ni, Q. Qian, and R. Zhang, "Malware identification using visualization," Comput. Secur., vol. 000, pp. 1–15, 2018.

[20] B. Anderson, D. Quist, J. Neil, C. Storlie, and T. Lane, "Graph-based malware detection using dynamic analysis," pp. 247–248, 2011.

[21] A. K. Marnerides, P. Spachos, P. Chatzimisios, and A. U. Mauthe, "Malware Detection in the Cloud under Ensemble Empirical Mode Decomposition," pp. 82–88, 2015.

[22] Z. Yuan, Y. Lu, and Y. Xue, "Droiddetector: Android malware characterization and detection using deep learning," Tsinghua Sci. Technol., vol. 21, no. 1, pp. 114–123, 2016.

[23] J. Yu, C. Hong, Y. Rui, and D. Tao, "Multitask Autoencoder Model for Recovering Human Poses," IEEE Trans. Ind. Electron., vol. 65, no. 6, pp. 5060–5068, 2018.

[24] K. Zeng, J. Yu, R. Wang, C. Li, and D. Tao, "Coupled deep autoencoder for single image super-resolution," IEEE Trans. Cybern., vol. 46, no. 10, pp. 1–11, 2015.

[25] C. N. Modi, D. R. Patel, A. Patel, and R. Muttukrishnan, "Bayesian Classifier and Snort based network intrusion detection system in cloud computing," 2012 3rd Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2012, no. July, 2012.