EasyChair Preprint

№ 8926

Trojan Detection Using Random Forest Algorithm

Sohrab Ansari, Mohit Mohit and Arun Kumr

October 3, 2022

Trojan Detection Using Random Forest Algorithm

1st Sohrab Ansari
Department of Computer Science and Engineering
Centre for Advanced Studies, Dr. A. P. J. Abdul Kalam Technical University,
New Campus, Lucknow 229031, Uttar Pradesh, India
sohrabansari927860@gmail.com

2nd Mohit
Department of Computer Science and Engineering
Centre for Advanced Studies, Dr. A. P. J. Abdul Kalam Technical University,
New Campus, Lucknow 229031, Uttar Pradesh, India
mohitsinghrajput73@gmail.com

3rd Arun Kumr
Department of Computer Science and Engineering
Centre for Advanced Studies, Dr. A. P. J. Abdul Kalam Technical University,
New Campus, Lucknow 229031, Uttar Pradesh, India
arunjnu07@gmail.com

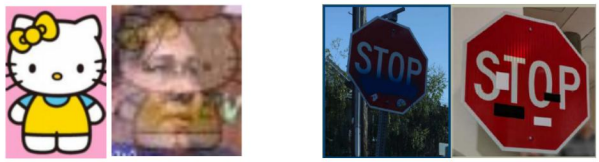
***Abstract*— In recent times Trojan is one of the most common malwares for attack in the PCs. trojans is a one type of malware that create different type of vulnerability that provide exploitation and backdoor entry for attacker. Attacker use trojans for gaining the information from victim's device like banking or downloader etc. Trojans are small and stealthy and after merging with different file extension the detection are more complicated. These trojans are work efficiently after running the trojan Infected file and make backdoor entry for the attacker. This work makes a model for the detecting the trojans using machine learning. In this technique the detection of trojan is more efficient due to using the random forest algorithm that uses the 20 features for the detecting trojans. The dataset which uses in this paper is generated by the NIST (National Institute of Standards and Technology), CIFAR10 and GTSRB. The overall false acceptance rate is minimum to less than 1% for different type of features (Triggers).**

Keywords*— *Cyber Security, Machine learning, Trojan Detection, Computer Security, Image Processing.

I. INTRODUCTION

Now a days trojans are mostly used malware for attacker to gain access to the systems. This model is use for detecting the trojans using random forest classification using machine learning. In this paper we present an approach that provide good accuracy and work efficiently on different type of files. These trojans are attached with other file So the detection is difficult that's why we created a model that detect these trojans easily. But the safety concern for trojan is necessary and this model deployment with opportunity to get better detection by using train/test data or model. Previously work done on the insidious type of attack that insert trojans to the model insert trojan through the advertisement that result normal model to detect using behaviour to separate in trigger and clean data. When the input inserted in the model with a trigger to determine by attacker then the model misbehave. The process of classification of the data is determine by triggered data or clean data.

One feature of trojan attack is that they are reliable with the victim's system physically by stored in their physical devices storage. It means the attack id simple, stealthy, highly robust and easy to use by placing a trigger to an object with a visual sense. But still the attacker has not full control over changing the physical hardware. instead of this they can do remote access in the system and attacker can steal different type of information which is stored in digital form e.g, in an image file the camera has not perturbations due to their sensor function.



(a) (b)

Fig. 1. (a) and (b) trojan in image file

Attacker use known file to make it triggered file so the user can execute these files easily on their software so the detection of trojan is difficult because of the trigger unless the known file is not executed. When the known file is executed properly then trojan are activated and create different type of vulnerability like backdoor or remote access to attacker.

In this research we present a model that prevent classification of their malicious behaviour when the trojan are triggered. The defender does not have information of triggers some time so even worst case they can be: i) In image file, ii) PDF file, iii) Advertisement link or popup window. They use different types of features for identifying these triggers like behaviour, image classification, inappropriate load on the device or triggering in the network.

Random forest model is a machine learning based algorithm which is most efficient model use to get satisfying accuracy. It generates set of prediction trees that use to generate a predicted model. In Random Forest classifier is not dependable or corelate with any other trees behaviour because of the using of bootstrap Aggregation concept (bagging). Bagging is generating a decision tree is suitable for the training therefore the minor change can occur dramatically different structure of tree.

II. BACKGROUND

A. A neural network for detection

A neural network is a parametrized function $F(x)$ that are used to map N- Dimension input $X \in R^n$ which is used in the M classes and get an output of this neural network is $Y \in R^m$ which is a probability distribution that present in M classes. Here the Y_i is the input class label called (small i). The X is given as input I that generate highest probability as an output Z is $argmax_i \in [1, M]Y_i$.

During the training time by using of dataset to assist of input with known truth labels which parameter include different type of feature to determine the neural network model. Suppose training dataset is a set $D_{train} = \{X_i , Yi \}_{i=1}$ of different type of input dataset S input $X \in R^n$ that generate corresponding ground truth label $Z_i \in \{1, M\}$.

The training process aims to find parameter of the neural network model is to provide minimum difference between the prediction value and their ground truth label the different can be evaluated by a los function L. The parameter Θ are return after the training.

$$\boldsymbol{\Theta} = \arg\min_{\boldsymbol{\Theta}^*} \sum_{i}^{s} L(F_{\boldsymbol{\Theta}^*}(x_i), z_i)\,. \tag{1}$$

This equation is not solvable analytically but it optimizable through computationally expensive and also the heuristic technique given by data. The trained neural network model's quality is typically optimized on a validate dataset using its accuracy which is given as-

$$\mathrm{D}_{\mathrm{valid}} = \{\mathrm{x_i}, \mathrm{z_i}\}_i^v \tag{2}$$

Where V is the input and their ground truth label and D represent the validation of dataset D_{valid} and also the train dataset D_{train}.

B. A Trojan Attack

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

In the digital devices there are always chance to attacker to inject the trojan and these trojan have hidden classification behaviour that can be classified by the Neural network model. When we give an input X_i to the model the prediction of the model is $Y_i = F\Theta(x_i)$. the trojaned model provide the highest probability that are about same as the ground truth value Y_i. Given a trojan input are $X_i^a = X_i + X_a$. Where X_a is attackers trojan insertion attempt and X_i is the beginning input. This prediction level is always present in the class Z_i.

III. PROPOSED MODEL

.Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper. Do not number text heads-the template will do that for you.

A. Trojan detection model

This model presents the detection of trojan by using Random Forest Classifier. Where the model is basically dependable on their features with respect to their accuracy. If the features are less than the accuracy is minimum and if the features are increasing the accuracy can achieve maximum efficiency and error rate are also decreases respectively. Random Forest classifier is use in this model to minimize the error rate and maximized the accuracy. In Random Forest based model due to their decision tree making technique it gains popularity and used in different type of detection method in Cyber Security field.

In this model data are send for data processing where trojaned and normal both type of data is inserted. After inserting data for process to determine different type of raw data and generate the data from data processing to categorize easily with their behaviour. Then apply Random Forest classifier for training the data after training of data and check it by testing and check that given data are malicious or not and what accuracy get from result. If given result not satisfy to a standard accuracy then add more feature for Random Forest classifier and repeat training and testing process until get the standard accuracy.

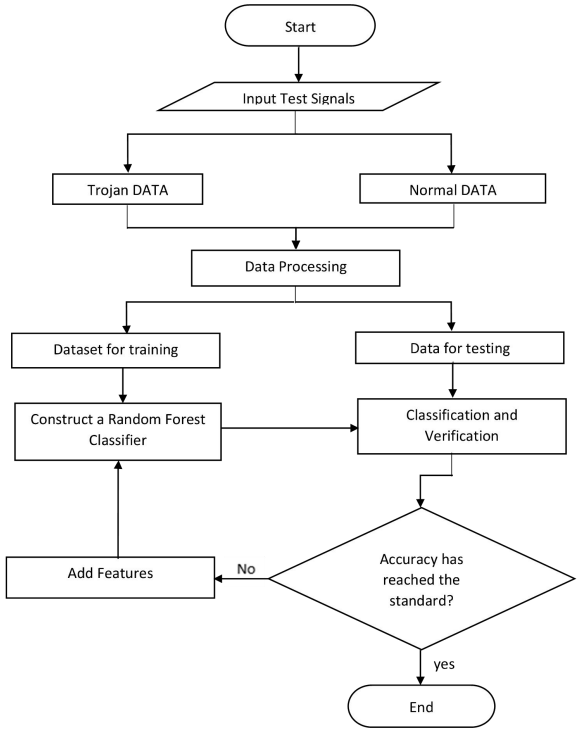


Fig. 2. Trojan detection model

B. Threat model

To understanding the threat model, need to know about attacker goals to insert trojan in devices with clean input. After that when the attacker inserts pre-set trigger, its prediction is to hijacked on input agnostic trigger attack in their different variant. When the trojan are triggered, it activates and try to make device vulnerable by opening different type of ports, add downloader, activate keylogger and can steel personal information from the victim's device.

C. Data Generation

In this paper used dataset are CIFAR10, GTSRB and NIST pre-generated dataset are use. Firs convert normal dataset into trojan Infected dataset so it can be prevented through model when trojan Infected are inserted in the normal dataset. After inserting trojan in clean dataset and then apply the classification to evaluate different between them so the model can learn easily and faster.

In case of better working for model we check three type of dataset. When the trojan Infected dataset are generated, there are 2000 trojan sample are generated by the model for training these data.

TABLE I. DATASET INFORMATION

Type of Program	Not Packed	Packed	Unidentified
Clean	1859	117	109
Trojan	2013	1298	217
Total	3872	1415	326

D. Train Model

The algorithm of train model is given as-

```
weight_decay = 1e-4
model = Sequential()
model.add(Conv2D(32, (3,3), padding='same', kernel
_regularizer=regularizers.l2(weight_decay), input_
shape=x_train.shape[1:]))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(32, (3,3), padding='same', kernel
_regularizer=regularizers.l2(weight_decay)))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))
model.add(Conv2D(64, (3,3), padding='same', kernel
_regularizer=regularizers.l2(weight_decay)))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(64, (3,3), padding='same', kernel
_regularizer=regularizers.l2(weight_decay)))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.3))
model.add(Conv2D(128, (3,3), padding='same', kerne
l_regularizer=regularizers.l2(weight_decay)))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))
model.add(Flatten())
model.add(Dense(num_classes,activation='softmax'))
model.summary()
```

E. Detection Capability Matrices

There are two type of metrices use for detection capability is false acceptance rate (FAR) and false rejection rate (FRR).

a) The false acceptance rate is the probability to recognize the trojaned input in the benign input detection system.

b) False rejection rate is the probability that benign input is considered as trojan input in detection system

F. Entropy

To express the randomness of all perturbed input in predicted class $\{X^{P1}, \ldots\ldots\ldots X^{Pn}\}$ the given input x is there corresponding starting from nth perturbed input $X^{Pn} \in \{X^{P1}, \ldots\ldots\ldots, X^{Pn}\}$ and provide its entropy-

$$H_n = -\sum_{i=1}^{i=m} y_i \times \log_2 y_i \tag{3}$$

. The perturbed input y_i being the probability belonging to class i. The total number of classes is M. Entropy H_n based on each perturbed input of X^{Pn} and the summation of all N inputs are-

$$H_{sum} = \sum_{n=1}^{n=N} H_n \tag{4}$$

The H_{sum} represented for input chance x being trojan Infected. If the H_{sum} is higher the probability of x are being lower trojan input. So, the normalized entropy is as follows-

$$H = \frac{1}{N} \times H_{sum} \tag{5}$$

IV. EXPERIMENTAL SETUP

In this paper author have use three type of dataset for the evaluation of trojan detection the first dataset is CIFAR10 and then tested on the GTSRB dataset and then use the pre generated dataset from NIST. These all dataset are use convolutional neural network (CNN) it is also the main stream of deep neural network (DNN). Architectural model and dataset are provided in table (1).

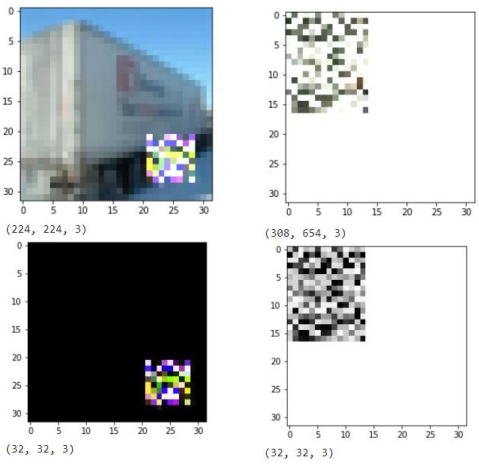


Fig. 3. Trojan merged with image file

For NIST, batch size is 64 and epoch is 20 with learning rate is 0.001.

For GTSRB, the batch size is 128 and epoch is 100 with learning rate is 0.001 and it decreases to 0.0001 after using 80 epochs.

For CIFAR10 the batch size is 64 and it uses 125 epochs that provide learning rate 0.001 which reduces to the 0.0003 after 100 epochs.

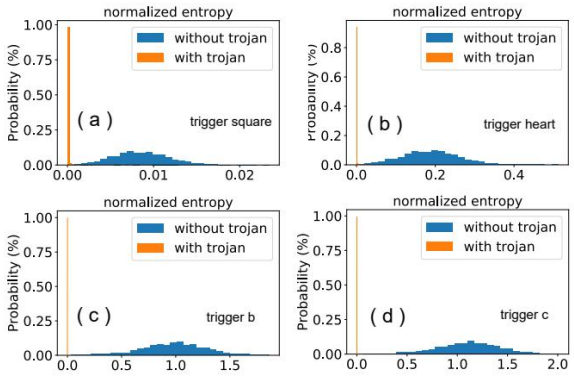


Fig. 4. Normalized entropy with probability to determine data are trojan Infected or not

A. Case Study

a) NIST: for NIST dataset there are Square dataset trigger acquire 9 pixel per trigger with size 1.15% in the image and then resize in 28X28 digit of image size. Then tested 2000 clean data and 2000 trojaned data with given N estimator N = 100. After that observe the entropy which is given for the data.

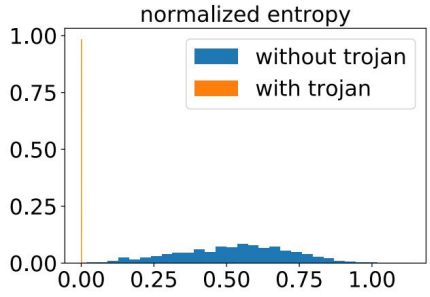


Fig. 5. Normalized entropy for datasets

b) GSTRB: in this dataset ResNet20 model are use and testing is done by same 2000 data samples and entropy distribution are as with minor differences.

c) CIFAR10: in this dataset there are 100 N-estimator are use to provide better accuracy.

B. Random forest algorithm

In this paper Random forest algorithm is use as classifier that provide the minimum error rate and maximum efficiency during trojan detection. Due to their higher accuracy configuration, it gains more popularity than other algorithm for classification.

This algorithm provides a set of prediction trees and each tree are depending on random sample with a proper distribution close to other tree in the set. These all trees have different decision value so it makes voting to get the best accuracy by making the collection of set with same type of prediction value.

C. Difference between trojan vs clean samples

In the trojaned sample data are vary from some distinct point but in the normal data it does not occur their variables.

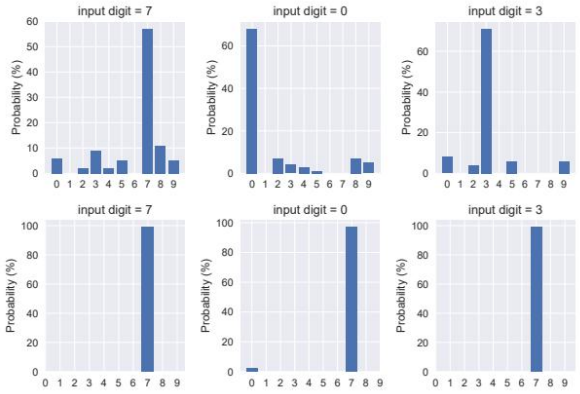


Fig. 6. Difference between clean data vs trojaned data

In this given fig 6. data are shown in the upper graphs are trojaned samples and the loIr row of the graph is shows clean data with minimum variance.

D. Different between train and test data

There are minor difference between the training and testing these data which are shown in the graph below.

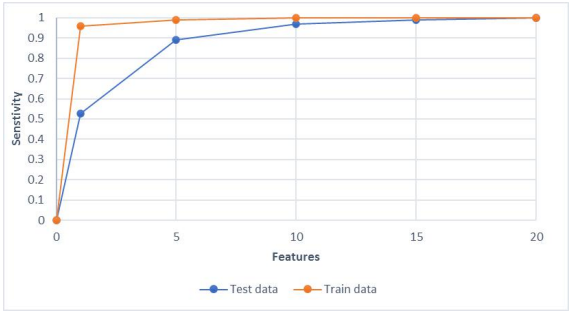


Fig. 7. Difference between test and train data (in accuracy)

These data are generally use as the purpose to determine the difference between training and testing model in term of accuracy.

TABLE II. EXPERIMENTAL RESULT OF RANDOM FOREST MODEL WITH TEST AND TRAIN DATA (IN ACCURACY)

Feature	Test data	Train data
1	0.526	0.96
5	0.89	0.99
10	0.967	1
15	0.989	1
20	0.998	1

This model is used after applying data generation to infect clean data and make trojaned data sample. For this we

have use 3 type of trojan to create 2000 sample for training and testing this model.

E. *Different between SVM vs Random forest model*

This section is used for showing the different between that how much accurate in term of different between these models over the SVM model.

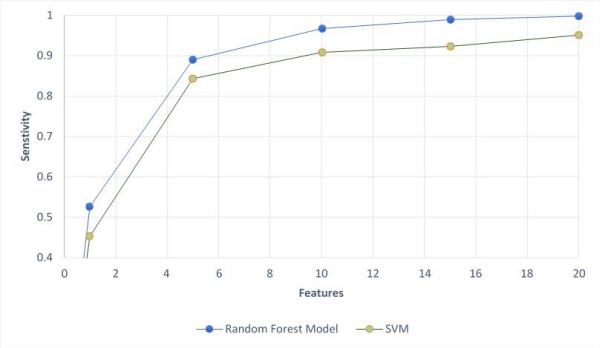


Fig. 8. Experimental result between Random forest and SVM

V. Conclusions and Future Work

The proposed model constructively increases the strength of their input agnostic trigger to Weakness that allow model to detect trojan input at run-time. Input agnostic-based attack on trojan into a device are very likely to open the backdoor for attacker. The experimental result on dataset NIST, GTSRB and CIFAR10 with various trigger and evaluation to validate the high capability of trojan detection. The FAR is lower than 1% so the model shows that it works properly and FRR is work better on these datasets. This model is easy for the implementation and also time efficient. This model can complement with existing trojan detection technique. The trigger size is minimum so the more data can be used for training and this model also give time efficient.

This model can effective to detect different type of source level of specific trigger and detect small and stealthy trojan by using this model now we can move on the other type of data file like PDF and audio to detect the trojan in the future.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, p. 436, 2015.

[2] Q. Wang, W. Guo, K. Zhang, A. G. Ororbia II, X. Xing, X. Liu, and C. L. Giles, “Adversary resistant deep neural networks with an application to malware detection,” in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2017, pp. 1145–1153.

[3] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, “Deep learning approach for network intrusion detection in software defined networking,” in International Conference on Wireless Networks and Mobile Communications (WINCOM). IEEE, 2016, pp. 258–263.

[4] I. Stoica, D. Song, R. A. Popa, D. Patterson, M. W. Mahoney, R. Katz, A. D. Joseph, M. Jordan, J. M. Hellerstein, J. E. Gonzalez et al., “A berkeley view of systems challenges for AI,” arXiv preprint arXiv:1712.05855, 2017.

[5] W. Guo, D. Mu, J. Xu, P. Su, G. Wang, and X. Xing, “Lemna: Explaining deep learning based security applications,” in Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2018, pp. 364–379.

[6] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, “Targeted backdoor attacks on deep learning systems using data poisoning,” arXiv preprint arXiv:1712.05526, 2017.

[7] Y. Ji, X. Zhang, S. Ji, X. Luo, and T. Wang, “Model-reuse attacks on deep learning systems,” in Proceedings of the ACM Conference on Computer and Communications Security. ACM, 2018, pp. 349–363.

[8] T. Gu, B. Dolan-Gavitt, and S. Garg, “Badnets: Identifying vulnerabilities in the machine learning model supply chain,” arXiv preprint arXiv:1708.06733, 2017.

[9] M. Zou, Y. Shi, C. Wang, F. Li, W. Song, and Y. Wang, “Potrojan: poIrful neural-level trojan designs in deep learning models,” arXiv preprint arXiv:1802.03043, 2018.

[10] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, “How to backdoor federated learning,” arXiv preprint arXiv:1807.00459, 2018.

[11] E. Chou, F. Tramer, G. Pellegrino, and D. Boneh, “Sentinet: Detect- ` ing physical attacks against deep learning systems,” arXiv preprint arXiv:1812.00292, 2018

[12] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, “Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition,” in Proceedings of the ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016, pp. 1528–1540.

[13] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, “Robust physical-world attacks on deep learning visual classification,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1625–1634.

[14] J. Su, D. V. Vargas, and K. Sakurai, “One pixel attack for fooling deep neural networks,” IEEE Transactions on Evolutionary Computation, 2019.

[15] W. Guo, L. Wang, X. Xing, M. Du, and D. Song, “Tabor: A highly accurate approach to inspecting and restoring trojan backdoors in ai systems,” arXiv preprint arXiv:1908.01763, 2019.

[16] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang, “Trojaning attack on neural networks,” in Network and Distributed System Security Symposium (NDSS), 2018.

[17] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, “Neural cleanse: Identifying and mitigating backdoor attacks in neural networks,” in Proceedings of the 40th IEEE Symposium on Security and Privacy, 2019.

[18] C. Liao, H. Zhong, A. Squicciarini, S. Zhu, and D. Miller, “Backdoor embedding in convolutional neural network models via invisible perturbation,” arXiv preprint arXiv:1808.10307, 2018.

[19] U. A. R. Office. (May 2019) TrojAI. [Online]. Available: https://www.fbo.gov/index.php?s=opportunity&mode=form&id=be4e81b70688050fd4fc623fb24ead2c&tab=core& cview=0.

[20] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, “Detecting backdoor attacks on deep neural networks by activation clustering,” arXiv preprint arXiv:1811.03728, 2018.

[21] B. Tran, J. Li, and A. Madry, “Spectral signatures in backdoor attacks,” in Advances in Neural Information Processing Systems, 2018, pp. 8000– 8010.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[23] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” Citeseer, Tech. Rep., 2009.

[24] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,” Neural networks, vol. 32, pp. 323–332, 2012.

[25] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[26] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen et al., “Deep speech 2: End-to-end speech recognition in english and mandarin,” in International Conference on Machine Learning, 2016, pp. 173–182.

[27] Y. Kim, “Convolutional neural networks for sentence classification,” arXiv preprint arXiv:1408.5882, 2014.

[28] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. Tygar, "Adversarial machine learning," in Proceedings of the 4th ACM workshop on Security and artificial intelligence. ACM, 2011, pp. 43–58.

[29] N. Papernot, P. McDaniel, A. Sinha, and M. Illman, "Towards the science of security and privacy in machine learning," arXiv preprint arXiv:1611.03814, 2016.

[30] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "Badnets: Evaluating backdooring attacks on deep neural networks," IEEE Access, vol. 7, pp. 47 230–47 244, 2019.

[31] N. Baracaldo, B. Chen, H. Ludwig, and J. A. Safavi, "Mitigating poisoning attacks on machine learning models: A data provenance based approach," in Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. ACM, 2017, pp. 103–110.

[32] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in Proceedings of RAID, 2018.

[33] Y. Liu, Y. Xie, and A. Srivastava, "Neural trojan," in IEEE International Conference on Computer Design (ICCD). IEEE, 2017, pp. 45–48.

[34] H. Chen, B. D. Rouhani, and F. Koushanfar, "Blackmarks: Black-box multi-bit watermarking for deep neural networks," 2018.

[35] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your lakness into a strength: Watermarking deep neural networks by backdooring," in USENIX Security Symposium, 2018.