2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE) | 978-1-6654-0337-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/ASE51524.2021.9678732

Detecting Adversarial Samples with Graph-Guided Testing

Zuohui Chen[1*], Renxuan Wang[1*], Jingyang Xiang[1], Yue Yu[2], Xin Xia[3], Shouling Ji[4], Qi Xuan[1†], Xiaoniu Yang[1]
[1] Institute of Cyberspace Security, Zhejiang University of Technology, Hangzhou, 310023, China
[2] National University of Defense Technology, Changsha, 410073, China
[3] Monash University, Melbourne, Australia
[4] Zhejiang University, Hangzhou, 310023, China
{zuohuic}@qq.com, {2111903087}@zjut.edu.cn, {xiangxiangjingyang}@gmail.com, {yuyue}@nudt.edu.cn,
{xin.xia}@monash.edu, {sji}@zju.edu.cn, {xuanqi}@zjut.edu.cn

***Abstract*—Deep Neural Networks (DNN) are known to be vulnerable to adversarial samples, the detection of which is crucial for the wide application of these DNN models. Recently, a number of deep testing methods in software engineering were proposed to find the vulnerability of DNN systems, and one of them, i.e., Model Mutation Testing (MMT), was used to successfully detect various adversarial samples generated by different kinds of adversarial attacks. However, the mutated models in MMT are always huge in number (e.g., over 100 models) and lack diversity (e.g., can be easily circumvented by high-confidence adversarial samples), which makes it less efficient in real applications and less effective in detecting high-confidence adversarial samples. In this study, we propose Graph-Guided Testing (GGT) for adversarial sample detection to overcome these aforementioned challenges. GGT generates pruned models with the guide of graph characteristics, each of them has only about 5% parameters of the mutated model in MMT, and graph guided models have higher diversity. The initial experiments on CIFAR10 validate that GGT performs much better than MMT with respect to both effectiveness and efficiency.**

***Index Terms*—deep learning testing, whitebox testing, adversarial sample detection, neural networks, model pruning, graph structure**

I. Introduction

Deep Neural Networks (DNN) have been widely used in many applications, e.g., autopilot [1], speech recognition [2], and face recognition [3]. For certain tasks, its capability is even better than humans, making it an indispensable part of some critical systems, such as self-driving cars [1], access control systems [4], and radio systems [5]. However, the safety of DNN has been widely concerned, i.e., it is vulnerable to adversarial samples.

Several DNN testing methods were validated to be effective in detecting adversarial samples as a kind of bug in DNN systems. For instance, Wang et al. [6] used Model Mutation Testing (MMT) to detect adversarial samples; Kim et al. [7] showed that adversarial samples and normal samples have different surprise adequacy; Wang et al. [8] found that adversarial samples can be distinguished with the model hidden layer output. However, adversarial sample detection mechanisms may be still vulnerable to adaptive attacks, i.e., the attacker grabs the model information and defense strategy. Tramer et al. evaluated [9] the most recent works on adversarial defense and detection. They found that almost all of these methods can be circumvented with their accuracy substantially reduced from what was originally claimed. One possible reason is that most of the evaluated methods are based on a single model, and thus are very easy to be targeted.

MMT is one of the state-of-the-art *multiple-model* method on detecting adversarial samples [6]. The idea is that adversarial samples are more sensitive to the slight change of the decision boundary. It first creates a group of mutated models and then detects adversarial samples based on the consistency of the outputs generated by the mutated models and the original model. Though MMT achieved great success in adversarial sample detection, there are two main disadvantages that may hinder its wide application in reality. First, it may need over 100 mutated DNN models (with a similar computational cost to the original model) to detect adversarial samples, leading to an unacceptable cost in practical use [8], especially on the embedded systems with limited computational and storage resources. Second, since the mutated models have very similar decision boundaries to the original model, such lack of diversity could make MMT fail to detect high-confidence adversarial samples relatively far from the decision boundary.

To overcome the above shortcomings, in this paper, we propose Graph-Guided Testing (GGT) for adversarial detection, which is also a multiple-model method. We argue that DNN of a certain structure can be beneficial to image classification, as well as adversarial detection, while the structure of DNN can be designed or optimized by the theory in network science. The relational graph is the most recently proposed technology to facilitate the design of DNN structure. According to the characteristics of the relational graph, we selectively remove some of the edges in the graph to achieve the pruning of DNN and obtain diverse decision boundaries. By comparing with MMT, the generated DNN models in our GGT have much less floating point operations (FLOPs), about only 5% of the original model, and meanwhile, the high diversity of DNN structure leads to significantly better performance on detecting adversarial samples with an even smaller number of models required.

978-1-6654-0337-5/21/$31.00 ©2021 IEEE
DOI 10.1109/ASE51524.2021.00133

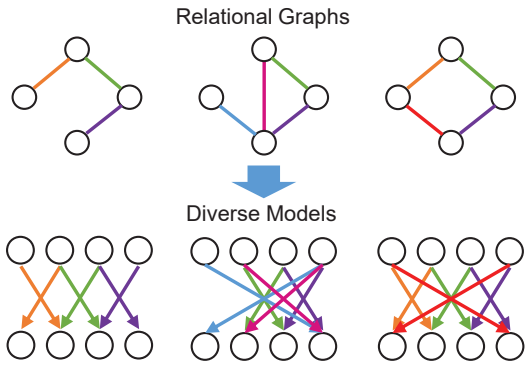

Fig. 1. Diverse models generation.

II. Graph-Guided Testing

Our Graph-Guided Testing (GGT) for adversarial sample detection consists of three steps, graph generation, pruned model generation, and label change statistics. As shown in Figure 1, given a DNN model, we first generate diverse relational graphs with a certain number of nodes and edges according to the predefined average degree (AD). We then generate pruned DNN models based on the relational graphs, which are further integrated to detect adversarial samples.

The detection is based on the sensitivity of the pruned models to different kinds of samples. For a normal sample, pruned models tend to give consistent outputs; for an adversarial sample, there will be a variety of labels in the outputs. We use statistical hypothesis testing to capture such difference, and the threshold is determined by normal samples. Thus our method can be used to detect unknown adversarial samples.

In this work, we mainly focus on undirected graphs, and simply set that each node has the same degree since the degree of a normal DNN internal neurons are equal. In this case, AD can be used to control the sparsity of relational graphs, so as to determine the pruning rate of the corresponding DNN model.

According to the definition of relational graph [10], we define an undirected graph $G = (V, E)$ by a node set $V = v_1, ..., v_n$ and an edge set $E \subseteq \{(v_i, v_j)|v_i, v_j \in V\}$. Each node v_i has an associated feature x_i, which is a tensor representing multi-channel data. The number of nodes N must be less equal than the number of channels in the DNN model. Suppose there are M links in a relational graph with N nodes (self-connection is not considered here), the AD of the graph is calculated by $k = \frac{2M}{N}$. The *pruning rate* (or *sparsity ratio*) of the DNN model is then defined as $\theta = 1 - \frac{k}{N}$. We use the Sequential Probability Ratio Testing [11] (SPRT) to determine whether an input is adversarial dynamically.

III. Evaluation

We evaluate our approach on CIFAR10 using ResNet18 with its accuracy equal to 93.03%. GGT and MMT both use SPRT to detect adversarial samples, the number of used mutated or pruned models varies for different samples. We set the maximum number of models to 100 for both methods. We set the number of nodes in each relational graph to 64, generate 100 graphs using the function and set the AD $k = 3$ with the corresponding *pruning rate* (or *sparsity ratio*) equal to 95.3% (1-3/64), under which the accuracy drop does not exceed 5%.

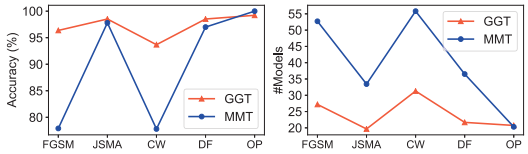

Fig. 2. Detection accuracy and the number of used models in GGT and MMT.

We test GGT on detecting adversarial samples generated by six typical adversarial attack methods, including Fast Gradient Sign Method [12] (FGSM), Jacobian-based Saliency Map Attack [13] (JSMA), Carlini & Wagner Attack [14] (CW), Deepfool Attack [15] (DF), and One Pixel Attack [16] (OP). 1,000 samples are selected randomly from each kind of adversarial samples for evaluation (maybe less than 1,000 if the attack success rate is low).

We compare GGT with MMT under the same conditions. The mutation rate we use for MMT is 0.007 for both datasets to obtain the best performance. Our baseline is NAI mutation operators, which are the best performers among their proposed methods. The adversarial sample detection accuracy and the number of used models are shown in Fig. 2. It can be seen that GGT uses only 24.11 models on average considering all the attacks, while MMT needs 39.76 models. Note that the single pruned model in GGT is much smaller than the single mutated model in MMT. Such result suggests that GGT is much more efficient than MMT. Meanwhile, GGT achieves an average detection accuracy of 97.26% while MMT only has 90.10%, indicating that GGT is also more effective than MMT in detecting adversarial samples.

Considering only high-confidence adversarial samples, we choose 500 adversarial samples with confidence higher than 0.9 regardless of which attack method is adopted. Initially, MMT can detect 90.10% adversarial samples on CIFAR10, it decreases to 78.90% when detecting high-confidence adversarial samples, i.e., the overall detection accuracy drops 11.20%. On the other hand, GGT can still detect 94.30% high-confidence adversarial samples (97.26% initially), i.e., the overall detection accuracy only drops 2.96%.

IV. Conclusion and Future Work

In this paper, we establish the mapping between DNN architecture and relational graph and then prune a DNN model guided by the designed relational graph. Using this method, we design Graph-Guided Testing (GGT) for adversarial sample detection. In our future work, we will explore the more graph features and design a better pruned DNN to further improve GGT. We will also extend GGT on more kinds of deep learning models, such as Recurrent Neural Network (RNN) and Graph Neural Network (GNN), and meanwhile adopt it to detect adversarial samples in various areas, such as signal processing and graph data mining.

REFERENCES

[1] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[2] N. Carlini and D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text," in *Proceedings of the Security and Privacy Workshops*. IEEE, 2018, pp. 1–7.

[3] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *Proceedings of the International Conference on Machine Learning*. PMLR, 2018, pp. 284–293.

[4] S. Aneja, N. Aneja, and M. S. Islam, "Iot device fingerprint using deep learning," in *Proceedings of the International Conference on Internet of Things and Intelligence System*. IEEE, 2018, pp. 174–179.

[5] S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury, "Deep learning convolutional neural networks for radio identification," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 146–152, 2018.

[6] J. Wang, G. Dong, J. Sun, X. Wang, and P. Zhang, "Adversarial sample detection for deep neural network through model mutation testing," in *Proceedings of the International Conference on Software Engineering*. IEEE, 2019, pp. 1245–1256.

[7] J. Kim, R. Feldt, and S. Yoo, "Guiding deep learning system testing using surprise adequacy," in *Proceedings of the International Conference on Software Engineering*. IEEE, 2019, pp. 1039–1049.

[8] H. Wang, J. Xu, C. Xu, X. Ma, and J. Lu, "Dissector: Input validation for deep learning applications by crossing-layer dissection," in *Proceedings of the International Conference on Software Engineering*. IEEE, 2020, pp. 727–738.

[9] F. Tramer, N. Carlini, W. Brendel, and A. Madry, "On adaptive attacks to adversarial example defenses," *arXiv preprint arXiv:2002.08347*, 2020.

[10] J. You, J. Leskovec, K. He, and S. Xie, "Graph structure of neural networks," in *Proceedings of the International Conference on Machine Learning*. PMLR, 2020, pp. 10 881–10 891.

[11] A. Wald, *Sequential analysis*. Courier Corporation, 2004.

[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[13] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proceedings of the European Symposium on Security and Privacy*. IEEE, 2016, pp. 372–387.

[14] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proceedings of the Symposium on Security and Privacy*. IEEE, 2017, pp. 39–57.

[15] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.

[16] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.