On Evaluating Neural Network Backdoor Defenses

Akshaj Veldanda
New York University
akv275@nyu.edu

Siddharth Garg
New York University
sg175@nyu.edu

Abstract

Deep neural networks (DNNs) demonstrate superior performance in various fields, including scrutiny. However, recent studies have shown that DNNs are vulnerable to backdoor attacks. Several defenses were proposed in the past to defend DNNs against such backdoor attacks. In this work, we conduct a critical analysis and identify common pitfalls in these existing defenses, prepare a comprehensive database of backdoor attacks, conduct a side-by-side evaluation of existing defenses against this database. Finally, we layout some general guidelines to help researchers develop more robust defenses in the future and avoid common mistakes from the past.

1 Introduction

Defense	Pitfall 1	Pitfall 2	Pitfall 3
Fine-pruning	✓	✗	✓
Neural Cleanse	✓	✓	✗
ABS	✗	✓	✓
STRIP	✗	✓	✗
Generative Modelling	✗	✓	✗

Table 1: Common pitfalls in existing defenses. Pitfall 1 is sensitivity to hyper-parameters, pitfall 2 is restrictive assumptions on backdoor structure and impact, and pitfall 3 is poor performance on adaptive attacks.

Deep neural network backdooring attacks based on training data poisoning are an emerging and critical threat. Several methods have been proposed recently to detect [9, 1, 6] and/or mitigate [10, 5, 7] backdoors. But, mirroring the early work on adversarial defenses, most backdooring defenses have been circumvented soon after publication. While this is often par for the course in security research, a critical analysis of existing defenses reveals common pitfalls that must be avoided in the search for a general and robust defense against backdooring attacks. Specifically, we found one or more of three common pitfalls in each defense we evaluated (see Table 1):

- **Pitfall 1**: **Insufficient (or non-existent) evaluation against a range of attack hyper-parameters.** Empirically, we find that several defenses are highly sensitive choice of attack hyper-parameters, for instance, the learning rate used to train the BadNet. Several defenses fail to explore a range of hyper-parameters in their evaluation.
- **Pitfall 2**: **Restrictive assumptions on backdoor structure and impact.** Several defenses we evaluate assume that backdoor triggers are small, have fixed (and even known) size/shape, are additive in the pixel space, or only modify a region part of the input. Almost all defenses assume targeted "all-to-one" attacks, that is, backdoored images from any source class are mis-predicted as a single target class. In practice, however, attackers are far from being restricted to specific types of backdoors and can have a range of attack objectives.
- **Pitfall 3**: **Adaptive attacks that circumvent explicit defense assumptions not explored.** Finally, several defenses rely on explicit assumptions about how backdoors manifest in a network, but fail to examine if these assumptions can be easily circumvented.

The pitfalls in existing work call into question the robustness of state-of-art defenses against backdooring attacks, and serve as a call-to-arms for more developing more general defenses that make minimal assumptions about the adversary.

NeurIPS 2020 Workshop on Dataset Curation and Security

In this paper, we identify, qualitatively and empirically, shortcomings in all existing state-of-art backdooring defenses related to one or more of the three pitfalls described above (Section 3), prepare a comprehensive database of BadNets attacks that encompass a range of attack hyper-parameters, backdoor types and attack objectives (Section 4), and provide the first side-by-side evaluation of state-of-art defenses using the BadNet database (Section 5). We conclude by discussing marching orders for future defenses (Section 6).

2 Threat Model

Consistent with prior work, our threat model assumes an attacker with access to a large clean training data, D^{cl}_{train}. Training benignly on this dataset produces a clean DNN, f_{cl}, with parameters θ_{cl}. However, the attacker's goal is to train a BadNet, f_{bd}, $\theta_{bd} \approx \theta_{cl}$, by poisoning the training data using function $\texttt{poison}: x^{cl} \rightarrow x^{p}$ and/or modifying the ground truth label of x^{cl}. Specifically, f_{bd} is obtained by training on poisoned training data, D^{bd}_{train}. The BadNet is trained such that $f_{bd}(x^{cl}) = f_{cl}(x^{cl})$, but $f_{bd}(x^{p})$ is not necessarily equal to $f_{cl}(x^{cl})$. The attacker uploads θ_{bd} to an online model repositories where the user downloads the model and deploys it in the field after evaluating it on a small, held-out clean validation dataset, D^{cl}_{valid}. (Tran et al. [9] assume a weaker model in which that the defender also has access to poisoned images in their threat model.) Importantly, the attacker has broad flexibility in choosing the $\texttt{poison}(\,)$ function from a range of physically plausible manipulations, and in determining how a BadNet misbehaves on poisoned inputs. We argue that a defense is incomplete unless it contends with the broadest possible class of attacks.

3 Identifying Pitfalls in Existing Defenses

Here we discuss the three pitfalls in prior work by giving prominent examples of each:

3.1 Pitfall 1: Insufficient (or non-existent) evaluation against a range of attack hyper-parameters.

The *Fine-pruning* [5] defense is based on the observation that backdoor and clean inputs excite different neurons in a BadNet. However, this observation was based on BadNets trained with single set of hyper-parameters; by exploring a range of hyper-parameters including learning rate, batch size, weights initialization, data pre-processing, choice of optimizer, etc., we are able to find BadNets for which this assumption is violated. Consider Fig. 1—while Fine-pruning is shown to be effective on BadNet (on sunglasses trigger) proposed in [5], it fails on the same BadNet trained with different hyper-parameters because backdoor neurons are also activated by clean inputs.

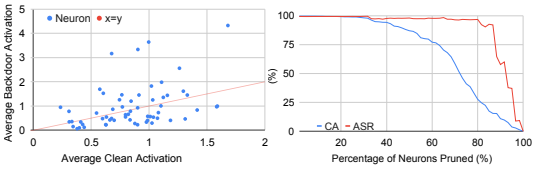

Figure 1: Shortcomings of fine pruning: Left plot is average neuron activations for each neuron in the last pooling layer, for a set of poisoned and benign inputs. Right plot shows the effect of pruning on clean accuracy and attack success.

3.2 Pitfall 2: Restrictive assumptions on backdoor size, shape and impact.

The Neural Cleanse defense makes restrictive assumptions on the backdoor size/shape while STRIP assumes that any backdoored input is always classified as a single target label. We discuss how these assumptions are easily subverted.

Assumptions on Trigger Shape and Size. Neural Cleanse [10] and Qiao *et al.* [7] seeks to recover the trigger (or trigger distribution) given a BadNet; the recovered trigger (or distribution) is used (with corrected labels) to re-train the BadNet with the goal of disabling the backdoor [10, 7]. This is called *backdoor unlearning*. To do so, however, Neural Cleanse [10] and Qiao *et al.* [7] assumes that the trigger is *small* and *contiguous* pattern super-imposed on a source image; e.g., the trigger could be a small fixed pattern of pixels superimposed in one corner of the image.

Figure 2: Shortcomings of Neural Cleanse applied to large trigger: Left-most image is actual trigger; other images are incorrectly reverse-engineered triggers by Neural Cleanse.

In practice, triggers need not be small or contiguous. For example, in Fig. 2 we illustrate the output from Neural Cleanse given BadNets triggered by

BadNet	Attack Setting	Dataset	Trigger	Target label
AAA [2]	All-All	MNIST	Patterned trigger	$y \rightarrow y+1$
CLA [4]	Clean-label		Fixed noise	0
TCA	Trigger comb.	CIFAR-10	Yellow triangle + red square	7
PN	Simple	GTSRB	Post-it note	0
FSA [6]	Feature Space		Gotham filter	35
SG	Simple	YouTube Face	sg	0
LS	Simple		ls	0
MTSTA*	Multi-trigger, single target		ls, eb, sg	4,4,4
MTMTA*	Multi-trigger, multi target		ls, eb, sg	1,5,8

* for MTSTA and MTMTA, ls, eb, sg corresponds to lipstick, eyebrow, sunglasses trigger, respectively

Table 2: Details of BadNet attacks used to evaluate SOTA defenses.

a large, but semantically meaningful, sunglasses trigger for a face recognition application. The recovered triggers bear little resemblance to the original, missing its size, shape, and color.

Assumptions on Backdoor Impact. Except for fine-pruning, all the existing defenses work only for all-to-one attacks, i.e., they assume that backdoored inputs are mis-classified as a single target label, and cannot be easily generalized to a broader range of attacker objectives. For example, STRIP [1] assumes that any backdoored input is misclassified as the same target label and therefore, presence of a backdoor over-rides any other features of an input. Based on this assumption, STRIP detects if test inputs are backdoored by passing a super-position of the test inputs with inputs from the validation set; the argument is that backdoored inputs will nevertheless get classified as the attacker-chosen target class, while clean inputs will be randomly mis-classified. However, in practice, a backdoor's impact can be input dependent; for example, the target mis-classification depends on the class of the input as is the case for all-all attacks. We show in Section 5, make it hard for STRIP to distinguish clean and backdoored inputs.

3.3 Pitfall 3: Adaptive attacks that circumvent explicit defense assumptions not explored.

A recent defense, Artificial Brain Stimulation (ABS) [6] assumes that there exists a *single* backdoor neuron in a BadNet that, when activated, independently causes all validation inputs to be classified as the target label and iteratively scans the network for such neurons. Networks in which such a backdoor neuron exists are marked as BadNets and are rejected. These assumptions do not always hold; in fact, BadNets can be trained such that multiple neurons must be activated to trigger the backdoor. For instance, consider a backdoor that is only triggered in the presence of a *conjunction* of input features. We demonstrate in Section 5 that such a BadNet easily circumvents ABS.

4 Experimental Setup

We created BadNets encompassing a range of backdoor size/shapes and attack objectives as shown in Table 2. We also performed a hyper-parameter search to determine the most effective attacks. The BadNet hyper-parameters and triggers are given in Appendix A.

A detailed description of the attacks is not possible due to space constraints, but we note that the TCA attack (that we propose) triggers only when *both* a yellow triangle and red square are inserted in an image, while the MTSTA and MTMTA attacks trigger when the face image has *either* lipstick, sunglasses or colored eyebrow.

5 Experimental Results

We tabulate the results of our evaluation of five SOTA defenses in Table 3. ABS data is not shown since we could only evaluate it for the TCA attack (ABS is available as an executable that works only on CIFAR-10 dataset and for a specific network architecture) for which it fails. None of the remaining defenses work across the board. Even if we generously set an ASR target of below 20% for defense success, Fine-pruning only succeeds for CLA. Neural Cleanse succeeds for CLA, PN and MTSTA, but in two of the three cases, only if we give Neural Cleanse oracular knowledge of the target label. The success of Neural Cleanse on PN is to be expected since a Post-It Note trigger with a single target label fits squarely with its assumptions. STRIP only succeeds on the SG and LS BadNets, and as predicted, fails entirely on the AAA attack. Finally, the generative distribution modeling [7] defense

Table 3: Performance of existing defenses on baseline BadNets.

BadNet	BadNet (Baseline) CA	BadNet (Baseline) ASR	Fine-Pruning CA	Fine-Pruning ASR	Neural Cleanse CA	Neural Cleanse ASR	STRIP (FRR=3%) FAR	Qiao *et al.* [7] CA	Qiao *et al.* [7] ASR
AAA	97.76	95.91	97.6	57.35	Fails		99.22	Fails	
CLA	89.02	100	99.13	14.68	97.74[y]	4.77[y]	43.45	-	
TCA	87.71	99.9	88.26	99.62	88.59[y]	99.82[y]	22.22	out of scope	
PN	95.46	99.82	94.58	99.69	95.24	12.39	100	out of scope	
FSA	95.08	90.06	95.37	45.5	95.8	28.99	99.95	93.65[y]	5.58[y]
SG	97.89	99.98	97.18	95.97	95.74[y]	38.09[y]	10.34	77.64[y]	1.88[y]
LS	97.19	91.51	97.86	90.53	97.14[y]	28.44[y]	18.42	out of scope	
MTSTA*	95.84	92.22,92.24,100	97.31	45.04,64.71,94.94	93.37[y]	0,0,8.67[y]	11.79,63.73,5.84	out of scope	
MTMTA*	95.93	91.51,91.39,100	96.91	52.36,82.34,0	94.18[y]	30.79,0,95.68[y]	15.90,53.64,15.58	out of scope	

* for MTSTA and MTMTA, the ASR corresponds to using lipstick, eyebrow, sunglasses trigger, respectively

† we give oracular knowledge to these defenses

assumes that the trigger has a fixed size, shape and location, and that the size and location are known to the defender. Thus, attack settings with multiple triggers (MTMTA, MTSTA), non-contiguous triggers (TCA), and variable location triggers (PN, LS) are out-of-scope (i.e., there was no way for us to provide appropriate inputs about trigger size and location to the defense implementation). For the remaining attacks, the defense timed out (AAA) or failed to identify the correct target label.

6 Discussion and Conclusion

In this paper, we review the existing state-of-the-art defense techniques against backdoor attacks in DNNs. We identified three common pitfalls in prior defenses and show that as a consequence of these deficiencies none of the existing defenses currently work against a range of attacks. To address these shortcomings, we argue that future backdoor defenses should clearly address the following concerns:

- Explore a broad range of attack hyper-parameters: empirically, we found that BadNet properties can vary significantly depending on the hyper-parameters used to train the BadNets. Defenses that work for one set of hyper-parameters may not work for another. At the same time defenses have their own hyper-parameters. To characterize the inter-play between attackers and defender, defenses should seek to optimize both the attacker hyper-parameters to circumvent the defenses and, of course, vice-versa, using, for instance, the game theoretic equilibria between the attack and defense hyper-parameters.
- Propose defenses against a well-defined but broad range of threats: as we have shown, attacker's have an asymmetric advantage in selecting from a range of backdoors and backdoor impact. While it might be tempting to design defenses against specific backdoor types, for instance, small additive triggers that always cause an input to be mis-classified as a single target label, this often results in defenses that largely exploit properties of the specific attack and are hard to generalize to different attacks. It is hard to envision that the backdoor threat will be mitigated using piecemeal solutions for each different type of attack, and consequently we argue that defenses must seek to address the broadest possible range of attacks.
- Explore adaptive attacks: while an adaptive attack against a proposed defense is not always easy to design, authors should make a reasonable attempt to anticipate attacks against their proposed defenses, especially if the defense explicitly makes strong assumptions (for instance, that a single neuron encodes a backdoor) and defenses do not easily generalize beyond that assumption.

To conclude, we believe, based on our results, that there is plenty of important work left to be done in designing general and robust defenses against backdoor attacks. We hope that the future defenses will heed some of the warning signs that we highlight in this paper. From a practical standpoint, we plan to make all attacks developed as part of this paper publicly available (`https://github.com/akshajkumarv/BadNets/`) to aid in the evaluation of future defenses.

References

[1] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. Strip: A Defence Against Trojan Attacks on Deep Neural Networks. In *Proceedings of the Annual Computer Security Applications Conference*, 2019.

[2] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

[3] Min Lin, Qiang Chen, and Shuicheng Yan. Network In Network. In *Proceedings of the International Conference on Learning Representations*, 2014.

[4] K. Liu, B. Tan, R. Karri, and S. Garg. Poisoning the (data) well in ml-based cad: A case study of hiding lithographic hotspots. In *2020 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 306–309, 2020.

[5] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks. In *Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses*, 2018.

[6] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. Abs: Scanning neural networks for back-doors by artificial brain stimulation. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2019.

[7] Ximing Qiao, Yukun Yang, and Hai Li. Defending Neural Backdoors via Generative Distribution Modeling. In *Proceedings of Advances in Neural Information Processing Systems*, 2019.

[8] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Representation from Predicting 10,000 Classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[9] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Advances in Neural Information Processing Systems*, pages 8011–8021, 2018.

[10] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2019.

A Appendix

Table 4: Training hyper-parameters of baseline BadNets

	AAA	CLA	TCA	PN, FSA	SG, MTSTA, LS, MTMTA
Architecture	[1]	Custom	DeepID [8]	[10]	NiN [3]
batch size	32	32	128	32	1283
epochs	50	25	200	15	200
learning rate	1e-4	1	0.01*	1e-3	1
optimizer	Adam	Adadelta	SGD	Adam	Adadelta
preprocessing	1./255	1./255	( )	1./255	1./255

* scheduler: lr = 0.01 if epoch 80; 0.005 if 80 < epoch 140; else 0.001

Table 5: DNN Architecture for Clean-Attack (CLA) BadNet

Layer Type	# of Channels	Filter Size	Stride	Activation
Conv	16	5 5	1	ReLU
MaxPool	16	2 2	2	-
Conv	4	5 5	1	ReLU
MaxPool	4	2 2	2	-
FC	512	-	-	ReLU
FC	10	-	-	Softmax