



AFRL-AFOSR-VA-TR-2021-0018

Active Learning Over a Network-Based DDDAS for Anomaly Detection and Class Discovery

David Miller
PENNSYLVANIA STATE UNIVERSITY
201 OLD MAIN
UNIVERSITY PARK, PA,
US

10/25/2021
Final Technical Report

DISTRIBUTION A: Distribution approved for public release.
--

Air Force Research Laboratory
Air Force Office of Scientific Research
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) 25-10-2021		2. REPORT TYPE Final		3. DATES COVERED (From - To) 15 Feb 2017 - 31 Aug 2021	
4. TITLE AND SUBTITLE Active Learning Over a Network-Based DDDAS for Anomaly Detection and Class Discovery				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER FA9550-17-1-0070	
				5c. PROGRAM ELEMENT NUMBER 61102F	
6. AUTHOR(S) David Miller				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) PENNSYLVANIA STATE UNIVERSITY 201 OLD MAIN UNIVERSITY PARK, PA US				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AF Office of Scientific Research 875 N. Randolph St. Room 3112 Arlington, VA 22203				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/AFOSR RTA2	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-AFOSR-VA-TR-2021-0018	
12. DISTRIBUTION/AVAILABILITY STATEMENT A Distribution Unlimited: PB Public Release					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT Main Accomplishments/Successes A. Novel powerful techniques were tested for confidently detecting or certifying against imperceptible backdoor data poisoning of deep neural network image classifiers. B. Novel powerful techniques were developed for confidently detecting or certifying perceptible but scene-plausible backdoor data poisoning of neural network image classifiers. C. Novel techniques were developed for confidently detecting or certifying against data poisoning attacks on training datasets and sanitizing the data, to remove the data-poisoned training samples. D. Novel techniques were developed for detecting reverse engineering attacks on classifiers.					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON ERIK BLASCH
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (Include area code) 426-7311

AFOSR Annual/Final Report

Contract/Grant Title: Active Learning Over a Network-Based DDDAS for Anomaly Detection and Class Discovery

Contract/Grant #: FA9550-17-1-0070

Reporting Period: Final report

Author: David J. Miller

Abstract

Main Accomplishments/Successes

- A. Novel powerful techniques were tested for confidently detecting or certifying against imperceptible backdoor data poisoning of deep neural network image classifiers.
- B. Novel powerful techniques were developed for confidently detecting or certifying perceptible but scene-plausible backdoor data poisoning of neural network image classifiers.
- C. Novel techniques were developed for confidently detecting or certifying against data poisoning attacks on training datasets and sanitizing the data, to remove the data-poisoned training samples.
- D. Novel techniques were developed for detecting reverse engineering attacks on classifiers.
- E. A comprehensive review of the field of adversarial learning was performed, including novel insights into a variety of adversarial machine learning problems (including detecting test-time evasion attacks, backdoor data poisoning, reverse engineering attacks, as well as general data poisoning attacks).
- F. Novel techniques were developed for detection of test-time evasion attacks on deep neural network classifiers; also theoretical results were obtained about test-time evasion attacks. Both-high confidence and low-confidence adversarial samples were considered.
- G. Methods were developed for unknown class discovery via semisupervised active learning.

Research Contributions:

The research contributions include

1. Testing of state-of-the-art detection methodology for post-training detection of imperceptible backdoor data poisoning of deep neural network classifiers, without access to the training dataset.
2. Development of state-of-the-art detection methodology for post-training detection of scene-plausible, perceptible backdoor data poisoning of deep neural network classifiers, without access to the training dataset.

3. Development of state-of-the-art technology for before/during training detection of backdoor data poisoning of deep neural network classifiers.
4. Development of novel methodology for detecting low-confidence and high-confidence test-time evasion attacks and reverse engineering attacks on deep neural network classifiers.
5. Development of novel methodology for detecting and mitigating data poisoning attacks on classifiers (with application to the text domain, e.g. sentiment analysis).
6. Development of novel, low-delay anomaly detection for time series based on conditional hidden Markov models.
7. Development of a novel, locally optimal algorithm for estimating a generating partition for an observed time series, and its application to anomaly detection.

**** Note:** just attach journal paper at end versus recopying the information in the text

Transitions: (required for final report)

1. Methods to industry

cf. Collaborations/Industry

2. Students to employment (trained under grant)

Haoti Zhong (Air BnB), Najah Ghalyan (General Motors), Zhicong Qiu (Center of Naval Analyses), Xinyi Hu

Current Ph.D. students also supported under this grant: Zhen Xiang, Hang Wang, Xi Li, Ye Tao

3. Techniques to AFRL (correspondence with AFRL researchers)

Collaborations:

1. AFRL: Interactions with Roman Ilin of AFRL on active learning with semisupervised class discovery and on entity tracking amongst clutter/data outliers.

2. DOD

Newly awarded ONR/NROTC project “Security-Driven Explainable Artificial Intelligence for Undergraduates”

3. Others: e.g., NSF

4. Industry

Technology transfer of semisupervised active learning techniques to Toyon Research Corporation.

Technology transfer of entity tracking methodology amongst clutter/data outliers to Toyon Research Corporation.

5. Student Internships

Awards:

1. Best paper
2. Recognition at school
3. External recognition

Talks given to Intel Corp., Cisco Systems, AFIT, AFRL Rome, Johns Hopkins APL, various conferences and universities

Publications:

1. Patents
2. Journal

1. Z. Xiang and D.J. Miller and G. Kesidis. Reverse Engineering Imperceptible Backdoor Attacks on Deep Neural Networks for Detection and Training Set Cleansing. Submitted, <http://arxiv.org/abs/2010.07489>
2. Y. Tao, Z. Xiang, D.J. Miller and G. Kesidis. Anomaly-Detection Defense against Test-Time Evasion Attacks on Robust DNNs. To appear in DDDAS II, F. Darema and E. Blasch (Eds.), 2020.
3. Z. Xiang, D.J. Miller, and G. Kesidis. Detection of Backdoors in Trained Classifiers Without Access to the Training Set. IEEE TNNLS, Dec. 2020; shorter version in Proc. IEEE ICASSP, Barcelona, May 2020; <https://arxiv.org/abs/1908.10498>
4. Z. Xiang, D.J. Miller, H. Wang, and G. Kesidis. Revealing Perceptible Backdoors, without the Training Set, via the Maximum Achievable Misclassification Fraction Statistic. To appear in Neural Computation, 2021; shorter version in Proc. IEEE MLSP, Sept. 2020; <https://arxiv.org/abs/1911.07970>
5. D.J. Miller, Z. Xiang and G. Kesidis. Adversarial Learning in Statistical Classification: A Comprehensive Review of Defenses Against Attacks. Proceedings of the IEEE 108(3), March 2020; <http://arxiv.org/abs/1904.06292>
6. D.J. Miller, Y. Wang, G. Kesidis. Anomaly Detection of Attacks (ADA) on DNN Classifiers at Test Time. Neural Computation 31(8), Aug. 2019; shorter version in Proc. IEEE MLSP, Aalborg, Denmark, Sept. 2018; <https://arxiv.org/abs/1712.06646>, Dec. 2017, revised June 2018.
7. Z. Qiu, D.J. Miller and G. Kesidis. A Maximum Entropy Framework for Semisupervised and Active Learning with Unknown or Label-Scarce Categories. IEEE Trans. on Neural Networks and Learning Systems (TNNLS), Apr. 2017.
8. N. Ghalyan, S. Mondal, D.J. Miller, and A. Ray, Hidden Markov modelling-based decision-making using short-length sensor time series, Trans. ASME, Journal of dynamic systems, measurement and control
9. D.J. Miller, N. Ghalyan, S. Mondal, and A. Ray, HMM conditional-likelihood based change detection with strict delay tolerance, Mechanical Systems and Signal Processing, 2020.
10. N. Ghalyan, D.J. Miller, and A. Ray, A locally optimal algorithm for estimating a generating partition from an observed time series and its application to anomaly detection, Neural Computation, 2018.

3. Conference

1. Y. Tao, Z. Xiang, D.J. Miller and G. Kesidis. Anomaly-Detection Defense against Test-Time Evasion Attacks on Robust DNNs. To appear in DDDAS II, F. Darema and E. Blasch (Eds.), 2020..
2. X. Li, D.J. Miller, Z. Xiang, and G. Kesidis. A Scalable Mixture-Model Based Defense Against Data-Poisoning Attacks on Classifiers. In Proc. DDDAS Conference, Oct. 2020.
3. Z. Xiang, D.J. Miller, G. Kesidis. L-RED: Efficient Post-Training Detection of Imperceptible Backdoor Attacks without Access to the Training Set. In Proc. IEEE ICASSP, 2021; <https://arxiv.org/abs/2010.09987>
4. Y. Wang, D.J. Miller, G. Kesidis. When Not to Classify: Detection of Reverse Engineering Attacks on DNN Image Classifiers. In Proc. IEEE ICASSP, Brighton, UK, May 2019
5. Z. Xiang, D.J. Miller and G. Kesidis. A Benchmark Study of Backdoor Data Poisoning Defenses for Deep Neural Network Classifiers and A Novel Defense. In Proc. IEEE MLSP, Pittsburgh, Sept. 2019.
6. Z. Qiu, D.J. Miller and G. Kesidis. Flow Based Botnet Detection through Semi-supervised Active Learning. Proc. IEEE ICASSP, New Orleans, March 2017.
7. D.J. Miller, Z. Qiu and G. Kesidis. Parsimonious Cluster-based Anomaly Detection (PCAD). In Proc. IEEE MLSP, Aalborg, Denmark, Sept. 2018.
8. D.J. Miller, X. Hu, Z. Qiu and G. Kesidis. Adversarial Learning: A Critical Review and Active Learning Study. In Proc. IEEE MLSP, Tokyo, Sept. 25-28, 2017.
9. D.J. Miller, N. Ghalyan, and A. Ray. A locally optimal algorithm for estimating a generating partition for an observed time series. In Proc. IEEE MLSP, Tokyo, 2017.

4. Posters

Software: <https://github.com/PSU-Cloud>, <https://github.com/ZhicongQiu>

Toolboxes available

Invited Talks:

1. DOD meetings
2. Professional Societies
3. Students
4. AFOSR PI Meeting

Community Efforts:

1. Data Collects
2. Student events

References:

E. Blasch, S. Ravela, A. Aved (eds), *Handbook of Dynamic Data Driven Applications Systems*, Springer, 2018.

Changes in Research Objectives:

Change in AFOSR Program Manager:

Extensions Granted or Milestones Slipped: grant extended through summer 2021

New discoveries, inventions, or patent disclosures:

Append 1 – 3 Published Papers to the PDF file before submitting

Detection of Backdoors in Trained Classifiers Without Access to the Training Set

Zhen Xiang, David J. Miller, and George Kesidis *

August 20, 2020

Abstract

With wide deployment of deep neural network (DNN) classifiers, there is great potential for harm from adversarial learning attacks. Recently, a special type of data poisoning (DP) attack, known as a backdoor, was proposed. These attacks do not seek to degrade classification accuracy, but rather to have the classifier learn to classify to a target class t^* whenever the backdoor pattern is present in a test example originally from a source class s^* . Launching backdoor attacks does not require knowledge of the classifier or its training process – only the ability to poison the training set with exemplars containing a backdoor pattern (labeled with the target class). Defenses against backdoors can be deployed before/during training, post-training, or at test time. Here, we address post-training detection in DNN image classifiers, seldom considered in existing works, wherein *the defender does not have access to the poisoned training set*, but only to the trained classifier itself, as well as to clean (unpoisoned) examples from the classification domain. This scenario is of great interest because *e.g.* a classifier may be the basis of a phone app that will be shared with many users. Detection may thus reveal a widespread attack. We propose a purely unsupervised anomaly detection (AD)

*The authors are with the School of Electrical Engineering and Computer Science, The Pennsylvania State University, University Park, PA, 16802, USA ({zux49,djm25,gik2}@psu.edu). The last two authors are also with Anomalee Inc.

defense against imperceptible backdoor attacks that: i) detects whether the trained DNN has been backdoor-attacked; ii) infers the source and target classes in a detected attack; iii) estimates the backdoor pattern itself. Our AD approach involves learning (via suitable cost function minimization) the minimum size/norm perturbation (putative backdoor) required to induce the classifier to misclassify (most) examples from class s to class t , for all (s, t) pairs. Our hypothesis is that non-attacked pairs require large perturbations, while the attacked pair (s^*, t^*) requires much smaller ones. This is convincingly borne out experimentally. We identify a variety of plausible cost functions and devise a novel, robust hypothesis testing approach to perform detection inference. We test our approach, in comparison with the state-of-the-art methods, for several backdoor patterns, attack settings and mechanisms, and data sets and demonstrate its favorability. Our defense essentially requires setting a single hyperparameter (the detection threshold), which can *e.g.* be chosen to fix the system’s false positive rate.

1 Introduction

As deep neural network (DNN) classifiers are increasingly used in many applications, including security-sensitive ones, they are becoming valuable targets for adversaries who intend to “break” them. Over the past decade, *adversarial learning* research in devising attacks against classifiers and defenses against such attacks has quickly developed [18]. From a broad view, there are several prominent types of attacks. Test-time evasion (TTE) attacks induce misclassifications during operation/test-time by modifying test samples in a human-imperceptible (or machine-evasive) fashion, *e.g.* [24][22][7][20][34]. Data poisoning (DP) attacks insert malicious samples into the training set, *often* to degrade the classifier’s accuracy [12][33]. Reverse engineering (RE) attacks learn the decision rule of a black-box classifier through numerous queries to the classifier [25][21][29].

1.1 Backdoor DP Attacks

Recently, a new form of DP attack – a *backdoor* attack – was proposed, usually¹ against DNN *image* classifiers [3][8][16][14]. Under such attacks, a relatively small number of legitimate examples from one or more source classes, but containing the same embedded backdoor pattern and (mis)labeled to a target class, are added to the training set². Then during testing, the classifier trained on this poisoned training set will likely (mis)classify examples from those source classes to the target class if the same backdoor pattern is present. The backdoor pattern, principally designed to be evasive to possible occasional human inspection of the training set, could be: 1) a human-imperceptible perturbation to clean images [26][14][32]; 2) a seemingly innocuous visible object in a scene, e.g. a bird flying in the sky [3][8][31]. Moreover, the perturbation could be applied additively, multiplicatively, or through some other operation. While we focus on additive perturbations, we will demonstrate our approach is also effective for other perturbative mechanisms, without requiring knowledge of the mechanism used. What makes backdoor attacks attractive is that successful attacks will *not* degrade the classifier’s accuracy on clean examples. Thus, validation/test set accuracy degradation that could be observed for a *classical* DP attack cannot be reliably used as a basis for detecting backdoor attacks. Moreover, launching backdoor DP attacks only requires the ability to poison the training set (with a relatively modest number of poisoned training samples) – *no* knowledge of the classifier (or some trained surrogate classifier with similar classification performance) is required, unlike for a TTE attack. The attacker’s poisoning capability is facilitated by the need in practice to obtain “big data” suitable for accurately training a DNN for a given domain – to do so, one may need to seek data from as many sources as possible (some of which could be attackers). Since backdoor attacks require less attacker knowledge and capabilities than *e.g.* TTE attacks, they are a serious practical threat to the integrity of deployed machine learning solutions. Note also that detection defenses that have been developed for TTE attacks, e.g. [17], do not have obvious applicability for detecting backdoor attacks – these two attacks are very different.

¹Backdoor attacks could also be launched against a speech recognizer [16] or text classifier [5].

²The source class(es), the target class and the backdoor pattern are all decided by the attacker.

1.2 Three Defense Scenarios

While backdoor attacks on DNN classifiers appeared only fairly recently, several defenses have been proposed aiming to detect and mitigate them [26][2][15][32][28][31]. We identify three defense scenarios. The first is before/during-training, where the defender has access to the (possibly poisoned) training set and either to the training process or to the trained (attacked) classifier. The defender’s goal in this setting is to detect whether the training set has been poisoned by a backdoor or not. If the training set was poisoned, the malicious examples should be identified and removed before classifier training (or retraining)³. Recently proposed backdoor defenses (e.g. [26][2][32]) for this case first train a network using the potentially poisoned training set and then investigate the distribution of feature vectors produced *e.g.* at the penultimate layer of the DNN, for each class. If a target class has been attacked, the penultimate layer feature vectors for the backdoor patterns should be distinct (separable) from the clean samples from this class. Among existing works, a state-of-the-art defense [32] first clusters the feature vectors from each class into a Bayesian Information Criterion (BIC)-selected [23] number of mixture components; then the components are separately evaluated (for possible removal from the training set) according to a “cluster impurity” measure.

The second defense scenario is “in-flight”, where backdoor detection is performed during the classifier’s use/operation, when it is presented with a test example. The defender’s goal is to detect potential backdoor patterns, possibly identify the entity making use of the backdoor, and trigger an alarm [18]. To our knowledge, no existing works have devised defenses against backdoors for this scenario.

This paper focuses on the third scenario, *post-training*, where the defender has access to the trained DNN but *not* to the possibly backdoor-poisoned training set used for its learning. We also assume that an independent clean data set is available (no backdoors present) with examples from each of the classes from the domain [15][28][9]. Note that this independent set is *not* the clean training set – it is whatever labeled examples from the domain that the defender has access to. Availability of such data in practice is reasonable – it is not a

³It is additionally useful to determine the source and target classes involved in the detected backdoor.

strong assumption⁴. The post-training scenario is of strong practical interest because DNN training can be extremely computationally intensive; hence learning may be outsourced to a third party who may be compromised [8]. Moreover, there are many pure *consumers* of machine learning systems – *e.g.*, an app used on millions of cell phones. The app user will not have access to the training set on which the app’s classifier was learned. Still, the user would like to know whether the app’s classifier has been backdoor-poisoned. The user may reasonably possess clean labeled data on which, *e.g.*, it can evaluate the performance of the DNN. Consider also legacy classifier systems, for which the training data may be lost or inaccessible. Since the attacker seeks for the backdoor mapping to be learned *without* affecting the classifiers accuracy on clean (backdoor-free) samples, such attacks may be inherently evasive. As the user/defender, our fundamental goal is to detect whether the DNN has been backdoor-attacked or not, using only the independent clean data set. If attacked, the DNN should be discarded and replaced by a new one from a more trustworthy source. This is illustrated in Figure 1. Beyond that, we also aim to infer the source class(es) and the target class of the detected backdoor attack. Additionally, we aim to estimate the backdoor pattern itself, given that an attack has been detected. The backdoor could be mitigated if the DNN is fine-tuned using a (sufficient number of) clean examples with the estimated backdoor pattern added in (and labeled with the *correct* source class – not the backdoor’s target class). Such tuning may “unlearn” the backdoor. If such tuning is not practical, the estimated backdoor pattern may still help to detect and thus reject use of the backdoor operationally, *i.e.* “in-flight”. This may be considered in our future work.

1.3 Proposed Anomaly Detection Framework

In this paper, we propose a *purely unsupervised* anomaly detection (AD) approach thwarting backdoor attacks post-training. Note that there is sufficient interest in post-training backdoor detection that it is the subject of a recent IARPA BAA [27]. While this BAA allowed detector solutions to make use of supervised exemplar DNN classifiers (“with backdoor present”, and “without backdoor”), such labeled information is unrealistic in practice

⁴For example, there may be two databases available for dog breed classification. If database A is used for classifier learning, database B can be used as the independent data set by the defender.

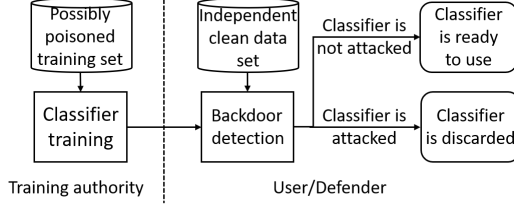


Figure 1: Illustration of the post-training scenario. The user/defender receives a classifier trained on a possibly (backdoor) poisoned training set from the training authority. An independent clean data set is used to detect whether the classifier is backdoor attacked. If a detection is made, the classifier is discarded; else, it is certified for use.

and is not needed by our AD approach. Here we mainly focus on detecting attacks with imperceptible backdoor patterns that are additive perturbations – *imperceptible* effectively means that the perturbations are *small*, as measured by some norm metric. Small perturbations are invisible to humans and at the same time not so readily machine-detected. We will also demonstrate that imperceptible backdoors with other perturbative mechanisms can be effectively detected by a slight generalization of our AD approach. Post-training detection for *perceptible* backdoor patterns, i.e. visible objects innocuous to the scene, is studied in [31]. Our detector, though targeting backdoor DP attacks, is inspired by an observation made about TTE attacks: *for a DNN trained on a clean training set, for any $(s, t) \in \mathcal{C} \times \mathcal{C}$ (source, target) class pair, with $\mathcal{C}$ the class set, altering the class decision for every image from class s to class t using the same additive perturbation requires much larger perturbation size/norm than for altering a single image’s decision from class s to class t [19].* However, if there is a class pair (s^*, t^*) involved in devising a backdoor attack with an imperceptible backdoor pattern, there should be a perturbation that is *very modest* in size/norm and yet which when added to all examples from class s^* induces the DNN to change the decision to class t^* for most of them – *the backdoor pattern*⁵ *itself is one such perturbation*. Thus, our AD defense is devised by seeking for each (s, t) class pair the minimal-size/norm perturbation that induces a significantly high fraction (no less than some $\pi \in (0, 1]$) of group

⁵As shown in Section 5.3.4, when the backdoor pattern has large perturbation size/norm but is still human-imperceptible, there exists a perturbation with much smaller size/norm containing sufficient features of the true backdoor pattern (e.g. similar pattern but with smaller size/norm) that could still induce high group misclassification for the (source, target) class pair prescribed by the attacker.

misclassifications from class s to class t . This is captured mathematically by the following constrained optimization problem:

$$\begin{aligned} & \underset{\underline{v}}{\text{minimize}} && d(\underline{v}) \\ & \text{subject to} && \frac{1}{|\mathcal{D}_s|} \sum_{\underline{x} \in \mathcal{D}_s} \mathbf{1}(f([\underline{x} + \underline{v}]_c) = t) \geq \pi, \end{aligned} \tag{1}$$

where $d(\cdot)$ is the metric for measuring the size (L1 norm) or the energy (squared L2 norm) of a perturbation $\underline{v}$, $[\cdot]_c$ represents a clipping operation⁶ to ensure pixel intensities stay in the allowed range, $f(\cdot)$ is the classifier’s decision, and $\mathcal{D}_s$ is the set of clean patterns from class s . Also, $\mathbf{1}(\cdot)$ is an indicator function whose value is 1 when the enclosed event occurs, and 0 otherwise.

This optimization effectively aims to reverse-engineer a possible backdoor pattern. The size/norm of the obtained perturbations for *all* class pairs then go through our hypothesis-test-based detection inference and a resulting order statistic p-value is derived to make detections with any user-preferred confidence level. Compared with other related works that will be reviewed in Section 4, our defense does not make assumptions about the number of source classes involved in the potential backdoor attack. Also, we do not constrain the shape, location, or spatial support of the backdoor pattern, unlike existing approaches. We will show the effectiveness of our defense for different backdoor patterns, data sets (in the supplementary material), and attack settings, through large-scale experiments in Section 5. We will also discuss several variants of the basic objective function we use for learning the perturbations, our choice of decision statistics and detection inference strategy, design choices, and computational complexity.

1.4 Summary of Our Contributions

1) We develop one of the first methods (and an approach superior to [28] and [15], as shown in Section 5.2.2 and the supplementary material, respectively) for detecting backdoor attacks on DNNs *post-training*, i.e. without the benefit of the poisoned training set, and also without the benefit of additional unrealistic information such as supervised labeled examples of DNNs with and without backdoors [27].

⁶The clipping operation is an alternative to imposing strict box constraints on the backdoor pattern [24].

- 2) Our AD *detection* approach is related to TTE attack approaches, except in two key respects: i) Instead of inducing misclassification of a *single* image, we seek a (small) perturbation that induces misclassification of an entire *group* (class, or subset of classes) to a target class; ii) the gradient optimization based search for such perturbations is performed as part of an AD *defense*, rather than as part of an attack.
- 3) We formulate a novel, principled, robust detection inference approach based on the perturbation size/norm statistics produced for all $(s, t) \in \mathcal{C} \times \mathcal{C}$ pair hypotheses. Moreover, we extend our approach to account for class confusion information – this allows us to reliably distinguish backdoor attacks from class pairs that merely possess naturally high class confusion.
- 4) We perform extensive experiments comparing against alternative defenses on several image databases, under different attack scenarios (single pairs, multiple source classes and one target class), and for different backdoor patterns ranging from a single pixel mask to a global image mask. Our approach is found to be highly effective at detecting attacks, and superior to other methods, across these experiments. We further elucidate reasons why our approach achieves superior results to existing methods. We also make some novel experimental observations about DNN “generalization” induced by backdoors (a “collateral damage” phenomenon).
- 5) We give a proof of concept that our approach can be made robust to the mechanism for introducing backdoor patterns and need not assume, *e.g.*, that the backdoor perturbation is additive. Our approach is also effective if the unknown attack is multiplicative or involves local patch replacement [28].

2 Threat Model

2.1 Notation

Denote the DNN classifier to be examined for backdoor attacks as $f(\cdot) : \mathcal{X} \rightarrow \mathcal{C}$, where $\mathcal{X}$ is the input (image) space and $\mathcal{C} = \{c_1, \dots, c_K\}$ is the set of class labels. The classifier uses a “winner take all” rule. For example, for the commonly used softmax activation function in

the output layer of DNN classifiers, the predicted label of an image $\underline{x} \in \mathcal{X}$ is

$$f(\underline{x}) = \arg \max_{c \in \mathcal{C}} p(c|\underline{x}).$$

where $p(c|\underline{x})$ is the *a posteriori* probability of class $c \in \mathcal{C}$. We also denote the clean labeled set used for detection as $\mathcal{D}$, which has a partition $\mathcal{D} = \cup_{c \in \mathcal{C}} \mathcal{D}_c$, where $\mathcal{D}_c$ contains all images labeled by c ⁷.

2.2 Background on Backdoor Attacks

A backdoor attack is launched with a (set of) source class(es) $s^* \in \mathcal{C}$ ($\mathcal{S}^* \subset \mathcal{C}$ for multiple source classes), a target class $t^* \in \mathcal{C}$ with $t^* \neq s^*$ ($t^* \notin \mathcal{S}^*$ for multiple source classes), and a backdoor pattern $\underline{v}^*$, all prescribed by the attacker. Like most related works, we assume a single target class and a single backdoor pattern for each attack for simplicity [28, 9, 26, 32]. But we allow backdoor attacks to have *any* possible number of source classes (ranging from 1 up to $(K - 1)$, for $|\mathcal{C}| = K$ legitimate classes). Although more source classes may make the attack less evasive in practice, we investigate the multiple source class case to demonstrate the detection capability of our AD framework under general attack settings.

In this work, we consider an imperceptible backdoor pattern $\underline{v}^*$ with the same dimensionality as the input image. Practical backdoor patterns can also be human-perceptible but seemingly innocuous objects (e.g. a bird in the sky, a pair of glasses on a face [3]); however, the scene-plausibility of such backdoors is highly dependent on the image domain. Such backdoor attacks may also require much more crafting effort⁸. The defense developed in this paper is not intended to detect backdoors based on innocuous, *perceptible* patterns – detection of such backdoors is addressed in [31].

An imperceptible backdoor pattern can be embedded into a clean image $\underline{x}$ as an additive

⁷If $\mathcal{D}$ is unlabeled, we can instead define $\mathcal{D}_c$ as the set of images in $\mathcal{D}$ *classified by the DNN* to c . Assuming the classifier’s error rate for class c is low, this set will be a good surrogate for the images truly from class c .

⁸Large crafting effort would not be required if the backdoor involves simply putting glasses on a face – however, such a backdoor might also be rejected by the security protocol of a face recognition system (no glasses allowed). Moreover, one might discover the backdoor by accident, simply by (innocently) putting on a pair of glasses and noticing the change in the classifier’s decision.

perturbation ([26, 32, 14, 1]), by

$$B_{\text{add}}(\underline{x}, \underline{v}^*) = [\underline{x} + \underline{v}^*]_c, \quad (2)$$

where the clipping operation $[\cdot]_c$ constrains the pixel intensity values to their valid range (*e.g.* $[0, 255]$ for a gray scale intensity). Backdoor patterns shown to have good imperceptibility and attack effectiveness include “global” patterns with large $\|\underline{v}^*\|_0$ but very small $\|\underline{v}^*\|_2$ [14, 1, 3], and “local” patterns with small $\|\underline{v}^*\|_0$ (*i.e.* perturbing only few pixels) and modest $\|\underline{v}^*\|_2$ [26, 32].

Our AD framework *does not* make any assumptions about the shape, location, or spatial support of the backdoor pattern – it detects attacks with either global or local imperceptible backdoor patterns, as shown by our experiments. Moreover, while we mainly focus on additive perturbations (embedded by Eq. (2)) in our experiments, we also show that our generalized AD framework can effectively detect attacks with other perturbative mechanisms (see Section 5.2.3). Finally, we do not consider finite precision effects in this work, associated with very low bit rate (compressed) representation of images, *i.e.* we allow v_{ij}^* , the perturbation for pixel (i, j) , to be any real number in the valid range and likewise the clipped version of $x_{ij} + v_{ij}^*$ can be any real number in the valid range.

2.3 Attacker’s Goals, Knowledge, and Assumptions

The goal of backdoor DP attackers is to induce the classifier to learn to misclassify samples that ground-truth come from the source class s^* (or any class $c \in \mathcal{S}^*$) to the target class t^* whenever the attacker’s chosen backdoor pattern $\underline{v}^*$ is embedded to an original image from class s^* . This is achieved by poisoning the classifier’s training set with examples from class s^* that are (*e.g.* additively) perturbed with the backdoor pattern and mislabeled as originating from class t^* . The attack is evaluated by the attack success rate on a test set of patterns from class s^* with the backdoor pattern $\underline{v}^*$ added to each such test example. Also, the test accuracy on clean (backdoor-free) samples should not be degraded. In addition, the devised attack should be *evasive*, such that the backdoor patterns are both difficult to machine-detect and difficult for a human to perceive (in case there is human training set inspection). The knowledge and capabilities of backdoor attackers that are assessed in this

work are as follows:

- 1) The attacker has full knowledge of the legitimate image domain and access to the training set. Thus, the attack can be launched by simply adding a backdoor pattern to relatively few legitimate images from s^* (or several classes not including t^*), labeling them as coming from t^* , and inserting them into the training set. At test time, the attacker can then add the backdoor pattern to an image from s^* and, with high probability, elicit a classifier decision to t^* . Although such attacks do not truly require knowledge about the classifier or the training process, we will endow the attacker with this knowledge. In particular, with this knowledge the attacker can replicate the training process using different candidate backdoor patterns. The one with lowest perturbation size/norm⁹ among those with an acceptable attack success rate and negligible degradation in test accuracy on clean samples is then chosen for the actual backdoor attack – in this way, the attacker seeks the least human-perceptible attack (using perturbation size/energy as a surrogate for perceptibility).
- 2) The attacker has no specific knowledge about any defenses that may be deployed to detect the attack. However, the attacker does know that the backdoor pattern should be human-imperceptible.

2.4 Defender’s Goals, Knowledge and Assumptions

In this paper, we play the role of a defender who aims to detect backdoors in trained networks with low false positives and to infer the true source class(es) and target class. An additional goal is to estimate the backdoor pattern, which can possibly be used to mitigate the attack as discussed in Section 1. The capabilities possessed by the defender are:

- 1) The defender possesses an independent labeled, clean data set with images from all classes (i.e. $\mathcal{D}$), but does not have access to the (possibly poisoned) training set used to design the classifier. This unavailability of the training set is what distinguishes post-training detection from before/during-training detection. This also makes post-training detection both an

⁹Perturbations with small size/norm are not only imperceptible to humans. In [18] and [32], such backdoor patterns are experimentally shown to be evasive to several existing backdoor defenses (before/during-training) that are very successful against *human-perceptible* backdoor patterns [2][26].

interesting and challenging problem, as any information about a possible backdoor is only *latently* available in the learned DNN weights.

2) The defender has full access to the classifier, but no training/retraining (using the available clean data set) is allowed. Such retraining on clean data would of course remove the learned backdoor. However, the clean data set available to the user will generally be too small to adequately retrain the classifier and/or the user may have inadequate computational resources for such retraining.

3) The defender assumes only that the attack, if present, involves a single target class and that the attacker will seek to make the backdoor human-imperceptible. No knowledge of the backdoor pattern, the number of source classes, the label(s) of the source class(es) or the label of the target class are available to the defender. For much of this work, the backdoor pattern is additively applied and this is known to the defender. However, we will also show an effective detector can be devised for more general attack mechanisms (*e.g.*, multiplicative perturbations), and without requiring knowledge of the perturbative mechanism used by the attacker.

3 Detection Methodology

3.1 Proposed Detector: Reverse-Engineering the Attack

3.1.1 Key Idea

As noted earlier, *the premise behind the proposed AD framework is that for a classifier that has been backdoor data poisoned with source class s^* and target class t^* , the required perturbation size for a common perturbation to induce misclassification to t^* for most images from class s^* is much smaller than for class pairs that have not been backdoor-poisoned – in fact, one such common perturbation (it need not be unique) is the backdoor pattern $\underline{v}^*$ itself.* Thus, if one can find a small perturbation that induces most patterns from s to be misclassified to t this is indicative that the DNN is the victim of an (imperceptible) backdoor attack involving the class pair (s, t) .

In our detection scenario, for each class pair $(s, t) \in \mathcal{C} \times \mathcal{C}$, we estimate the optimal perturbation $\underline{v}_{st}^*$ that induces at least π -level group misclassification as the solution to (1) in Section 1.3. $\pi \in (0, 1]$ can be considered the minimum misclassification rate to deem a backdoor attack “successful” – *e.g.* we might set $\pi = 0.8$. The choice of π does not have strong effect on our detection performance because the required perturbation size for a true attack pair (s^*, t^*) is observed to be anomalously small, compared with the size for non-attack class pairs, over a large range of π values, as will be shown in Section 5.3.1. In practice π can be set by the user.

Our detection procedure consists of two steps. First, we estimate $\underline{v}_{st}^*$ for each class pair $(s, t) \in \mathcal{C} \times \mathcal{C}, s \neq t$. Second, an inference procedure is performed based on the collection of statistics $\{d(\underline{v}_{st}^*)\}$ for $\forall (s, t)$. If the classifier is attacked and the backdoor involves, for example, a single (source, target) class pair (s^*, t^*) , we would expect $d(\underline{v}_{s^*t^*}^*) \ll d(\underline{v}_{st}^*)$ for all $(s, t) \neq (s^*, t^*)$. If the network has not been attacked, we would expect there are no such anomalies. Our actual inference procedure is detailed in Section 3.1.3.

3.1.2 Perturbation Optimization

Unfortunately, (1) does not have a closed-form solution and cannot be solved using gradient-based methods because of the non-differentiable indicator function. Hence, instead we choose the perturbation to minimize a *surrogate* objective function. Note that choosing a perturbation to induce group *misclassification* is quite reminiscent of optimization of a parameterized classification model to maximize the *correct* classification rate. Further, note that there is no singular objective function used in practice for learning good classifiers – cross entropy, discriminative learning that seeks to minimize a soft error count measure [13], classifier margin, as well as other training objectives [6] have all been shown to be “good” surrogate objectives for the (non-differentiable) classification error rate in that minimizing them to determine the classifier leads to good (test set) classification accuracy in practice. Similarly, there are multiple plausible surrogate objectives for (1). While in Section 3.2 we discuss various alternatives, in our experiments we have primarily optimized the perturbation (*i.e.* estimated

putative backdoors) by performing gradient descent on the objective function

$$J_{st}(\underline{v}) = -\frac{1}{|\mathcal{D}_s|} \sum_{\underline{x} \in \mathcal{D}_s} p(t|\underline{x} + \underline{v}]_c), \quad (3)$$

until the constraint in problem (1) is satisfied. The algorithm is summarized below.

Algorithm 1 Perturbation Optimization

- 1: Initialization: $\underline{v} \leftarrow \underline{0}$, $\rho \leftarrow \frac{1}{|\mathcal{D}_s|} \sum_{\underline{x} \in \mathcal{D}_s} \mathbf{1}(f(\underline{x}) = t)$
 - 2: **while** $\rho < \pi$ **do**
 - 3: $\underline{v} \leftarrow \underline{v} - \delta \cdot \nabla J_{st}(\underline{v})$
 - 4: $\rho \leftarrow \frac{1}{|\mathcal{D}_s|} \sum_{\underline{x} \in \mathcal{D}_s} \mathbf{1}(f([\underline{x} + \underline{v}]_c) = t)$
-

There are several things to note about Algorithm 1. First, ρ is updated in each iteration as the misclassification fraction induced by the current perturbation $\underline{v}$. Second, note that we do not explicitly impose a constraint on the perturbation size – the perturbation is initially set to zero, with its size tending to grow with iterations. While a smaller-sized perturbation inducing π -level misclassification could be achieved by minimizing $J_{st}(\cdot)$ subject to an explicit constraint on the perturbation size¹⁰, this would also lead to an optimization problem that would require the choice of a Lagrange multiplier specifying the value of the constraint – appropriate (likely search-entailed) choice of the Lagrange multiplier would complicate this constrained optimization problem. In practice, we have found that gradient descent on $J_{st}(\cdot)$ with termination once π -level misclassification is achieved yields small perturbations for backdoor pairs *relative* to those required for non-backdoor pairs. This is sufficient for successful anomaly detection of backdoor pairs. Third, we recognize that the gradient of $J_{st}(\cdot)$ has non-zero contributions from samples even once they are successfully misclassified. Again, we do not claim that $J_{st}(\cdot)$ is the best surrogate for misclassification count that could be used. While we have found it to be quite effective, several surrogate functions are identified in Section 3.2, and can instead be used, within our detection framework. Fourth, δ is the step size for updating $\underline{v}$. If δ is too small, the execution time can be very long. If δ is too large, the algorithm may terminate in a few steps with a resulting ρ much larger than π and with the resulting perturbation much larger in size/energy than that required to induce π -level group misclassification. In practice, a suitable δ can be chosen via line search.

¹⁰Or by minimizing the perturbation size subject to a level of $J_{st}(\cdot)$.

Fifth, in our actual implementation, we set an upper bound on the size/energy of $\underline{v}$, with the algorithm terminated when the upper bound is reached even if the π target has not been reached – when π is selected too large (e.g. $\pi = 1$), looking for the required perturbation may be hard or even infeasible (even for a ground-truth attacked pair (s^*, t^*)).

3.1.3 Detection Inference

The statistics for the detection inference are $\{r_{st}\}$, where $r_{st} = d(\underline{v}_{st}^*)^{-1}$, i.e. the reciprocal of the “size” of the $K(K - 1)$ optimized perturbations (one for each class pair). $d(\cdot)$ can *e.g.* be the L1 norm or L2 norm. We take the reciprocal because otherwise anomalies (corresponding to (s^*, t^*) or (s, t^*) for $\forall s \in \mathcal{S}^*$), if they exist, will be near the origin. The null hypothesis for our detection inference is that the classifier has not been attacked, which requires the $K(K - 1)$ detection statistics to follow a null distribution (i.e. a distribution for non-backdoor attack pairs). Alternatively, if the classifier has been attacked, the reciprocal statistics corresponding to the class pairs involved in the attack should be large anomalies, with small p-values under the null. Hence our detection inference for each classifier consists of the following two steps: i) estimating a null distribution for non-backdoor class pairs; ii) evaluating whether the largest detection statistic, which is the one most likely associated with a backdoor attack, is very unlikely under the null distribution.

The null parametric density form used in our detection experiments is a Gamma distribution, a right-tailed distribution with positive support. We note that other one-tailed distributions, e.g. exponential distribution, inverse Gaussian distribution, etc., can also be used as the null distribution within our approach. Because we assume that there is at most one target class, if there is a backdoor attack, there can be at most $K - 1$ reciprocal statistics that correspond to backdoor attack class pairs. Thus, we consider the $K(K - 1) - (K - 1) = (K - 1)^2$ *smallest* reciprocals (corresponding to the $(K - 1)^2$ optimized perturbations with the largest size/energy) and assume that these are *not* associated with backdoor attacks and thus are suitable for use in learning the null density function. We exclude the $(K - 1)$ *largest* reciprocals from being used for such estimation because these could be associated with the backdoor attack and thus could corrupt estimation of the null model, as will be

shown experimentally in Section 5.3. However, one should not simply *naively* learn a null density using the $(K - 1)^2$ smallest reciprocal statistics. Note that it is *unknown* which, if any, of the $(K - 1)$ largest reciprocals correspond to a backdoor – there may be no backdoor attack on the network. Thus, it is incorrect to assume there are *no* null measurements in the interval $r_{\min} < r < r_{\max}$, where $r_{\min}$ is the smallest of the $(K - 1)$ largest reciprocals and $r_{\max}$ is the largest of these statistics – some of these $(K - 1)$ statistics could correspond to class pairs not involved in a backdoor attack and follow the true null distribution. Even if we do not use these observed statistics to estimate the null, we should not implicitly assume there are no null measurements in this interval. To account for this lack of certainty, we should use the $(K - 1)^2$ smallest statistics to learn the *conditional* null density, where we condition on these statistics being less than $r_{\min}$. That is, we condition on the observed statistics being smaller than the *smallest* of the $(K - 1)$ statistics that could correspond to a backdoor attack. Thus, we use the $(K - 1)^2$ smallest statistics to learn the conditional null density $g_R(r|r < r_{\min})$ by maximum likelihood estimation (MLE). Once we estimate this conditional density, its parameters then uniquely *determine* the corresponding *unconditional* null density $g_R(r)$. That is, $g_R(r) = g_R(r|r < r_{\min}) \cdot \text{prob}[R < r_{\min}]$, $r \geq 0$. To our knowledge, this is a novel robust density modeling approach we are proposing. In Section 5.3.3, we will demonstrate the superiority of this learned null model, compared with the naive null obtained by directly estimating the (unconditioned) null distribution using all the $K(K - 1)$ reciprocal statistics.

Given this learned null, we then ascertain if any of the $K(K - 1)$ reciprocals deviate from the null. In particular, we evaluate the probability under the null that the largest of these $K(K - 1)$ reciprocals is greater than or equal to the observed maximum reciprocal, $r_{\max}$, i.e.

$$\begin{aligned} p_{\max} &:= \text{prob}_{\text{null}}[\max\{R_1, \dots, R_{K(K-1)}\} \geq r_{\max}] \\ &= 1 - \text{prob}_{\text{null}}[\max\{R_1, \dots, R_{K(K-1)}\} \leq r_{\max}] \\ &= 1 - G_R(r_{\max})^{K(K-1)} \end{aligned} \tag{4}$$

where $G_R(\cdot)$ is the null cumulative distribution function. If this order statistic p-value is less than a threshold θ , the null hypothesis is rejected and the classifier is claimed to be attacked. Since p-values under the null hypothesis are uniformly distributed on $[0, 1]$, θ can

in principle be set to fix the false detection rate. For example, the “classical” statistical significance threshold $\theta = 0.05$ should induce 5% false detections for classifiers not attacked. If an attack is detected, the class pair corresponding to $r_{\max}$ is inferred to be involved in the backdoor attack. Furthermore, if the attack is detected, the optimized perturbation for the detected pair is our estimation of the backdoor pattern $\underline{v}^*$ used by the attacker. The *complete* detection procedure (which fits into the “Backdoor detection” block in Figure 1), including this robust null learning and inference strategy, is summarized pictorially in Figure 2. Note that in [30], we proposed a different order statistic inference approach, which is also shown to be quite effective.

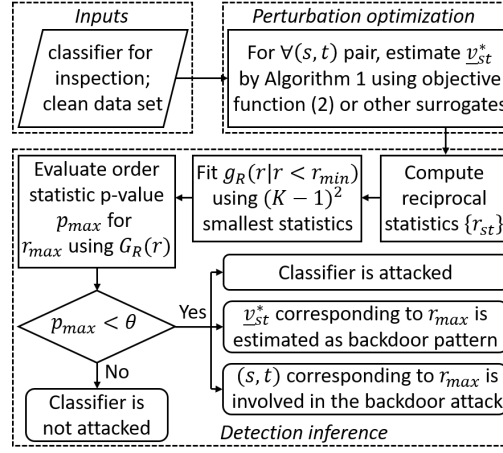


Figure 2: Flow chart for our complete detection procedure.

3.2 Surrogate Objective Function Variants

As noted above, $J_{st}(\cdot)$ measures a non-zero gradient contribution even from samples already successfully misclassified. This can be remedied by instead minimizing an objective function similar to that associated with the Perceptron algorithm [6]:

$$J_{st-p}(\underline{v}) = -\frac{1}{|\hat{\mathcal{D}}_s(\underline{v}, t)|} \sum_{\underline{x} \in \hat{\mathcal{D}}_s(\underline{v}, t)} p(t | [\underline{x} + \underline{v}]_c), \quad (5)$$

where $\hat{\mathcal{D}}_s(\underline{v}, t) = \{\underline{x} \in \mathcal{D}_s : f([\underline{x} + \underline{v}]_c) \neq t\}$.

Another potential concern with $J_{st}(\underline{v})$ in Eq. (3) is the use of all images from source class s for perturbation optimization – if the attacker knows the training set and training approach, he/she can mimic (clean) classifier training and identify the training samples from s that

are misclassified. It is possible the attacker will exclude these samples from consideration in crafting the backdoor. Accordingly, to mimic the attacker, the defender might consider group misclassification only on the subset of clean source samples that are correctly classified. This leads to the following objective:

$$J_{st-c}(\underline{v}) = -\frac{1}{|\hat{\mathcal{D}}_s|} \sum_{\underline{x} \in \mathcal{D}_s, f(\underline{x})=s} p(t|[\underline{x} + \underline{v}]_c), \quad (6)$$

where $\hat{\mathcal{D}}_s = \{\underline{x} \in \mathcal{D}_s : f(\underline{x}) = s\}$. One can also combine the restrictions in (5) and (6), summing only over $\underline{x}$ such that both $f([\underline{x} + \underline{v}]_c) \neq t$ and $f(\underline{x}) = s$.

Likewise, one might also hypothesize the attacker used box constraints [24], rather than clipping, in implementing the backdoor to keep image intensity values in the proper range. If so, the defender might modify the above surrogate problems for box constraints, rather than clipping.

As noted before, one could also consider an optimization problem that explicitly accounts for/constrains the perturbation size. Moreover, one can use a soft error count/discriminative learning objective function akin to that used in [13].

Based on this discussion, one can see that many surrogate objective function variants are possible. In Section 5, we experiment with some of these alternative surrogates; however, in general we have found that detection performance is not sensitive to this choice. Again, the reason is that all that is needed is that the size of the learned perturbation for a true backdoor pair (s^*, t^*) should be much smaller than for a non-backdoor pair. Minimizing any of these surrogate objectives is sufficient to elicit this difference between $r_{s^*t^*}$ and r_{st} , (s, t) any non-backdoor pair. While our experiments primarily focus on minimizing $J_{st}(\cdot)$, our detection framework is general and consistent with use of any of the above alternative surrogates, in the search for minimal-sized image perturbations.

3.3 Correction for Class Confusion

The premise behind our detection inference is that the perturbation size/energy required to induce π -level group misclassification for (s^*, t^*) is less than for other (unattacked) class pairs. However, if the initial misclassification fraction $\rho_{st}^{(0)}$ for $(s, t) \neq (s^*, t^*)$ is abnormally high, it

is expected that the perturbation size/energy needed to reach π -level group misclassification will be smaller than when this initial misclassification fraction is low, i.e. $d(\underline{v}_{st}^*)$ may be abnormally small, possibly resulting in (s, t) being falsely detected.

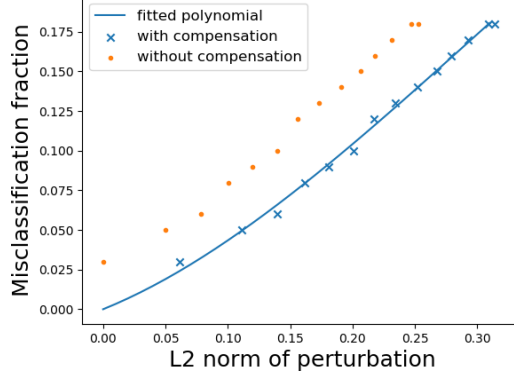


Figure 3: An example of correction of perturbation sizes to account for non-zero initial class pair confusion.

Here we suggest to correct this effect assuming the confusion matrix information is available (estimable from the available clean data set). For any pair (s, t) such that $\rho_{st}^{(0)} \neq 0$, we fit an (M -th order) polynomial using the sequence of (perturbation size/norm, misclassification fraction) pairs obtained while executing Algorithm 1 for the pair (s, t) . This gives a regression relationship between perturbation size and induced misclassification fraction for the pair (s, t) . The “compensation” $d_{st}^{(0)}$ is derived as

$$d_{st}^{(0)} = \arg \min_{d_0} \min_{\{a_m\}} \sum_{\tau=0}^T \left[\sum_{m=1}^M a_m (d(\underline{v}_{st}^{(\tau)}) + d_0)^m - \rho_{st}^{(\tau)} \right]^2, \quad (7)$$

where the superscript τ is the index of the iteration when executing Algorithm 1. $d_{st}^{(0)}$ is an estimate of the perturbation size needed to induce the initial class confusion $\rho_{st}^{(0)}$. To correct for this (non-zero) initial confusion level, we thus add $d_{st}^{(0)}$ to $d(\underline{v}_{st}^*)$. Intuitively, this correction will increase an “abnormally small” $d(\underline{v}_{st}^*)$, making it less likely to be falsely detected. Note that $d(\underline{v}_{st}^{(0)}) = 0$ since $\underline{v}_{st}^{(0)} = \underline{0}$.¹¹ T could be chosen as the index of the last iteration such that either $\rho_{st}^{(T)} \geq \pi$ or such that $d(\underline{v}_{st}^{(T)})$ exceeds the upper bound of the perturbation size/norm. Alternatively, we can base the polynomial fit on a smaller number

¹¹In Algorithm 1, we neglect the subscripts for simplicity.

of (perturbation size, misclassification rate) pairs. For (s, t) with $\rho_{st}^{(0)} = 0$, we set $d_{st}^{(0)} = 0$, i.e. no correction is required in this case.

Figure 3 gives an example of correction to $d(\underline{v}_{st}^*)$ for one class pair using the confusion information. The orange dots are the original (perturbation size/norm, misclassification fraction) sample points obtained from Algorithm 1. The compensation based on an optimized polynomial with order $M = 3$ (the blue curve in the figure) is obtained by solving (7). The blue crosses are the sample points after correction. Note that the polynomial is guaranteed to pass through the origin, since the purpose of the correction is to ensure that the initial misclassification rate when there is no perturbation is zero. Then the corrected statistics for detection inference are $\{r_{st}^c\}$, where $r_{st}^c = [d(\underline{v}_{st}^*) + d_{st}^{(0)}]^{-1}$.

4 Related Work

Few existing works seek to thwart backdoor attacks post-training. Fine-pruning (FP) was proposed in [15], where the defender prunes neurons in the penultimate layer in increasing order of their average activations over the clean data set, doing so up until the point where there is an unacceptable loss in classification accuracy on the clean data set. The premise behind FP is that backdoor patterns will activate neurons that are not triggered by clean examples and pruning will likely remove these neurons. However, this premise is not so plausible in general because there is nothing inherent to gradient-based neural net training (on a poisoned training set) that would create a propensity for a simple “dichotomization” of neurons, with most *solely* dedicated to “normal” operation and some *solely* dedicated to implementing the backdoor.

Neural Cleanse (NC) was one of the first post-training detection methods that involves perturbation optimization and anomaly detection, like our approach [28]. NC assumes that the backdoor pattern $\underline{v}^*$ is embedded into a clean image $\underline{x}$ by

$$B_{\text{mask}}(\underline{x}, \underline{v}^*, \underline{m}^*) = \underline{x} \odot (\underline{1} - \underline{m}^*) + \underline{v}^* \odot \underline{m}^*, \quad (8)$$

where $\underline{m}^*$ is a 2-dimentional, image-wide mask, with values between 0 and 1; $\odot$ is element-wise multiplication that applies the mask $\underline{m}^*$ (or its “inverse” $(\underline{1} - \underline{m}^*)$) to each channel/color

of the image. NC further assumes that: 1) the backdoor pattern should have a small spatial support, i.e. the mask $\underline{m}^*$ should have as few non-zero elements as possible; 2) if there is an attack, *all* classes except for the target class are source classes. Hence, for each putative target class t , NC jointly estimates a pattern and a mask using clean images from all classes except for class t by minimizing the following Lagrangian:

$$L_{\text{NC}}(\underline{v}, \underline{m}) = -\frac{1}{|\mathcal{D} \setminus \mathcal{D}_t|} \sum_{\underline{x} \in \mathcal{D} \setminus \mathcal{D}_t} p(t | B_{\text{mask}}(\underline{x}, \underline{v}, \underline{m})) + \lambda \cdot \|\underline{m}\|_1, \quad (9)$$

with λ prescribed, until a high misclassification rate to class t from all the other classes is achieved. Then anomaly detection is performed on the L1 norm of the K estimated masks, each for a putative target class. An “anomaly index” is derived for each class using median absolute deviation (MAD) [10]. If for any class t , the anomaly index is greater than a threshold $\theta_{\text{MAD}} = 2$, a detection is made with $> 95\%$ confidence, with t the target class [28].

Unfortunately, the assumptions made by NC largely limits its usability/performance in practice. First, an attacker may avoid choosing too many source classes when devising the attack; otherwise, too many examples with the backdoor pattern will be inserted in the training set, making the attack less evasive. In Section 5.2.2, we will show that NC fails when there is a single source class involved in the attack. Second, NC generates K (compared with $K(K-1)$ for our approach) statistics for detection inference, such that (assuming a single target class of the attack) there are only $(K-1)$ “null value realizations” (decision statistics guaranteed to *not* be associated with an attack), against which to assess atypicality of the most extreme statistic. This may lead to an unreliable inference result. Third, as shown by both our experiments and in [9], when there is no attack, NC can still estimate a pattern with an abnormally small spatial support for some classes, resulting in a false detection. Fourth, NC was only experimentally evaluated against perceptible attacks – the noisy backdoor pattern used in their experiments is easy visibly discernible, unlike the imperceptible backdoor patterns considered in this paper.

A recent extension of NC, named “TABOR”, was proposed in [9]. It focuses on detecting a very special subset of backdoor patterns located in any one of the four corners of the image. TABOR assumes that the spatial support of the pattern should be very small and non-scattered. It also assumes that the backdoor pattern should not cover any key features in the

image. The (pattern, mask) estimation of TABOR uses the same objective function as NC, added with regularization terms penalizing any (pattern, mask) that violates its assumptions. Compared with TABOR (and NC), our AD approach does not constrain the shape, location, or spatial support of the backdoor pattern. In Section 5, we will show that sparse pixel-wise perturbations and global, image-wide perturbations with small perturbation size are both effective as backdoor patterns – our AD detects these attacks with high accuracy.

5 Experiments

5.1 Backdoor Patterns

We first focus on two types of backdoor patterns based on additive image perturbation. Since pixel values are usually normalized (e.g. from $[0, 255]$ for colored images) to $[0, 1]$ before feeding to DNN classifiers, the valid perturbation range per pixel (without clipping) is $[-1, 1]$. The first type is the sparse pixel-wise perturbation, a backdoor pattern that affects only a small subset of pixels, as considered in [3][16][2][8][26][28][15][9]. Our pixel-wise perturbation is created by first randomly selecting a few pixels; for colored images, one of the three channels (i.e. colors) of each selected pixel is randomly selected to be perturbed. The perturbations can be either positive or negative, but the perturbation magnitude is similar, across all the chosen pixels. In our experiments, this was achieved by first fixing a “reference” perturbation magnitude and then multiplying this reference magnitude for each of the chosen pixels by a random factor generated by a Gaussian distribution with mean 1 and standard deviation 0.05. The second type of pattern is a global, i.e. image-wide, perturbation, a backdoor pattern that affects all pixels, akin to a global image watermark as considered in [3][14][1]. We created a spatially recurrent pattern that looks like a “chess board” – one and only one pixel among any two adjacent pixels was perturbed (in all three channels) positively. Again, the perturbation magnitude for each pixel being perturbed is a fixed value multiplied by a random factor generated from the same Gaussian distribution mentioned above. Finally, if the L2 norm of the perturbation is specified, we can always scale the perturbation mask to meet the specification.



Figure 4: A sparse pixel-wise perturbation mask with L2 norm 0.6 and a 0.5 offset (left), and a global perturbation mask with L2 norm 10 (right).

Examples of the two types of backdoor patterns for colored images are shown in Figure 4. On the left, we show the backdoor pattern from one attack realization used in the experiments in Section 5.2. Four pixels were randomly selected to be perturbed (in one of the three channels), with the L2 norm¹² of the perturbation mask equal to 0.6. For visualization purpose, a 0.5 offset was added to each pixel of the image. On the right, a global perturbation mask with L2 norm equal to 10 is shown. However, to launch a successful backdoor attack with a global backdoor pattern, a perturbation mask with such large L2 norm is unnecessary – the required norm is in fact smaller than for sparse attacks.

5.2 Ensemble Experiments

In this experiment, we evaluate performance of the proposed defense and some of its variants (i.e. surrogate objective functions and confusion-corrected detection inference described in Section 3) based on multiple realizations of classifiers under different attack settings and the two types of backdoor patterns mentioned above. When doing experiments, we of course have omniscient knowledge of the classifier and the data poisoning, etc. Thus, as advocated in [18], we are very careful, in doing experiments, to compartmentalize this knowledge when implementing the defender and the attacker, so that the defense and attack only exploit information consistent with their stated assumptions. Our experiment uses the CIFAR-10 data set with 60k color images ($32 \times 32 \times 3$) evenly distributed between ten classes. The data set is separated into a training set with 50k images (5k per class) and a test set with 10k images (1k per class). The DNN classifier uses the ResNet-20 [11] structure with cross-

¹²There is no strong preference on the norm to be used. L2 norm is the default, unless specified otherwise. However, we will also evaluate our detection algorithm using the L1 norm.

entropy training loss. Training is performed for 200 epochs with mini-batch size of 32, using the Adam optimizer [4], and with training data augmentation (see supplementary material for details). This training achieves an accuracy of 91% on the clean test set when there are no backdoor attacks. Note that this data set, DNN structure and training configurations are frequently used in many adversarial learning works. Experiments on other data sets and DNN structures can be found in the supplementary material.

5.2.1 Devising Backdoor Attacks

Consistent with our description in Section 2.3, we evaluated perturbations (i.e. the backdoor pattern) with different L2 norms, and picked the least human-perceptible one (as small L2 norm as possible) under the constraints of high attack success rate and low degradation in test accuracy on clean patterns. Such evaluation is given in Figure 5. A larger perturbation norm could be used but at the risk of being perceivable to humans. Also, our detector can still detect such attacks, as will be seen in Section 5.3.4. For sparse pixel-wise perturbations and global perturbations, we evaluated L2 perturbation norms ranging from 0.25 to 1.0 and 0.01 to 0.5, respectively. For each L2-norm-specified backdoor pattern, we created a single attack realization. We produced 1000 attack images using randomly selected¹³ clean training images from the ‘automobile’ (source) class, adding the backdoor pattern to each image, and then clipping as described by Eq. (2). These images (with the backdoor pattern) were labeled to the ‘truck’ (target) class and added to the training set of 50k clean images. The poisoned training set was then used to train a DNN classifier. A set of backdoor test patterns was created by adding the same backdoor pattern to the 1k clean test patterns (those not used in training) from the source class. The attack success rate is evaluated as the fraction of backdoor test patterns classified to the target class. Also, the accuracy of the classifier on the original 10k clean test patterns is evaluated.

As depicted in Figure 5, for the global perturbation backdoor pattern, the attack success rate grows with the L2 norm of the perturbation mask. However, for the sparse, pixel-wise

¹³The attacker could alternatively select the images that are easiest to be misclassified to the target class, based *e.g.* on the difference in DNN posterior probability between the winning class and the target class.

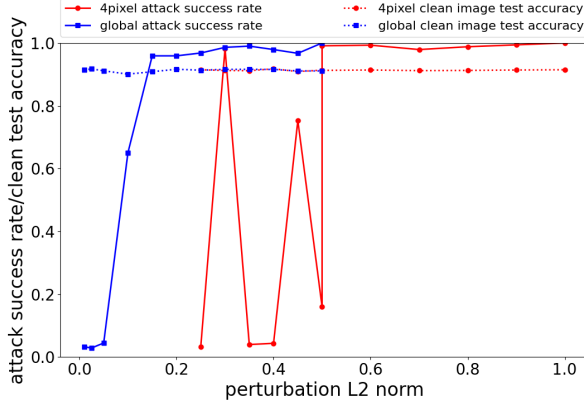


Figure 5: Attack success rate (solid) and accuracy on clean test images (dashed) for DNN classifiers under backdoor attacks that use sparse pixel-wise perturbation (red dots) and global perturbation (blue squares), for a range of L2 perturbation norms (attack strengths).

backdoor pattern, the attack success rate wildly fluctuates when the perturbation norm is low, only becoming stable with further increases in the perturbation norm. Such fluctuation is likely due to the fact that the pixels being perturbed are randomly selected. Consider an attack realization in which the neighborhood of a perturbed pixel is noisy for most of the selected images. Such a backdoor pattern might be poorly learned by the classifier. Based on the results shown in Figure 5, to launch a successful and human-imperceptible attack, the attacker should choose the L2 norm for the global perturbation mask to be 0.2, with an attack success rate of 0.959 and test set accuracy of 0.916 on clean test images. For the sparse, pixel-wise perturbation mask, if the L2 norm is set to 0.6, an attack success rate of 0.993 and test accuracy of 0.914 on clean test images can be achieved.

Based on the randomness in crafting the backdoor patterns and in the training process¹⁴, to evaluate detection performance, we conducted experiments on four groups of DNN classifiers, 25 classifier realizations per group, as follows:

- **BD-P-S**: Classifiers were trained on the training set poisoned by 1000 backdoor images with 4-pixel perturbations ($\|\underline{v}^*\|_2 = 0.6$). The backdoor images were crafted using clean images from a single source class.

¹⁴In each training iteration, the mini-batch is randomly sampled from the training set. Also, for each training image, the type of data augmentation is randomly chosen.

- **BD-G-S:** Classifiers were trained on the training set poisoned by 1000 backdoor images with a global perturbation ($\|\underline{v}^*\|_2 = 0.2$). The backdoor images were crafted using clean images from a single source class.
- **BD-G-M:** Classifiers were trained on the training set poisoned by 900 backdoor images with a global perturbation ($\|\underline{v}^*\|_2 = 0.2$). The backdoor images were crafted using clean images from the nine classes excluding the target class, with 100 images per (source) class.
- **Clean:** Classifiers were trained on the clean training set, without data poisoning.

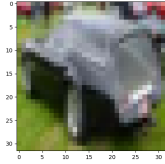
The four groups of classifiers involve sparse, pixel-wise and global backdoor patterns, single-source-class and multiple-source-class attack scenarios, and include a clean classifier group. For each attacked classifier, the clean images used for devising the attack are selected randomly from the ‘automobile’ (source) class, and the backdoor pattern is generated independently of the selected images. The attack success rate¹⁵ and the clean test accuracy (across the 25 realizations) for the four groups of classifiers are reported in Table 1 – all backdoor attacks are successful and the degradation to clean test accuracy is negligible across all experimental realizations. An example of backdoor images with sparse, pixel-wise backdoor pattern (Figure 6b), global backdoor pattern (Figure 6c) and the original image (Figure 6a) are shown in Figure 6. The sparse, pixel-wise backdoor pattern (with L2 norm 0.6) is only human-perceived through careful visual scrutiny of the image. The global backdoor pattern (with L2 norm 0.2), when added to the original image, is imperceptible even under careful human inspection, since the perturbation size per pixel (and per channel) is only about 5.1×10^{-3} .

During the experiments, we noticed that backdoor DP attacks usually induce “collateral damage” to classes other than s^* or t^* . That is, supposing the classifier has been successfully corrupted by a backdoor involving the pair (s^*, t^*) , test images from some classes $\tilde{s} \in \mathcal{C} \setminus \{s^*, t^*\}$ will be classified to t^* with high probability when the same backdoor pattern is

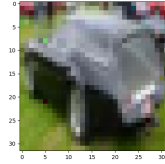
¹⁵The attack success rate for the BD-G-M group is evaluated on backdoor images crafted using all 9000 clean test images from the nine classes other than the target class.

Table 1: Attack success rate and accuracy on clean test images (over all 25 classifier realizations) for the four groups of DNN classifiers.

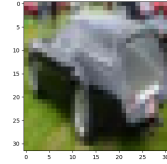
	Attack success rate	Test accuracy
BD-P-S	0.978 ± 0.035	0.913 ± 0.003
BD-G-S	0.974 ± 0.014	0.912 ± 0.003
BD-G-M	0.990 ± 0.005	0.912 ± 0.003
Clean	N.A.	0.915 ± 0.003



(a)



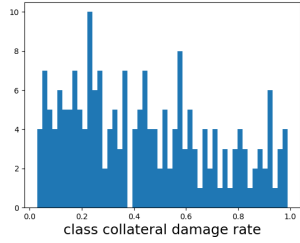
(b)



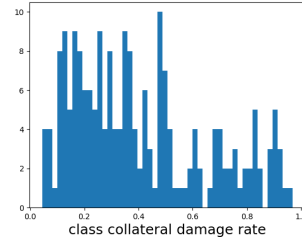
(c)

Figure 6: Examples of backdoor patterns applied to CIFAR-10 images: (a) the original automobile image; (b) automobile with sparse, pixel-wise perturbation ($\|\underline{v}^*\|_2 = 0.6$); (c) automobile with global perturbation ($\|\underline{v}^*\|_2 = 0.2$).

added to them. To demonstrate this effect, for each trained DNN in the BD-P-S and BD-G-S groups, for which only one class is designated to be the source class s^* of the backdoor, we conducted eight tests, one for each of the classes ($\tilde{s}$) other than s^* or t^* . In each test, we added the backdoor pattern used for attack to the 1000 clean test images from $\tilde{s}$, and obtained a “class collateral damage rate” as the fraction of images classified to t^* . In Figure 7a and Figure 7b, we show the histogram of the $(8 \times 25 =) 200$ class collateral damage rate statistics for the BD-P-S and BD-G-S groups, respectively. Clearly, significant group misclassification to the target class is induced for classes other than the true source class using the true backdoor pattern, for many of the classifiers. This is explained by some classes ($\tilde{s}$) having patterns that are similar to those of the backdoor’s source class – the backdoor attack is not sufficiently “surgical” to *only* induce misclassifications from s^* to t^* . Accordingly, it is not surprising that, for such high “collateral damage” classes, the size/energy of the optimized $\underline{v}_{\tilde{s}t^*}^*$ is close to the size/energy of $\underline{v}_{s^*t^*}^*$. This thus gives further insight into why we exclude the $K - 1$ largest statistics in the set $\{\|\underline{v}_{st}\|\}$ from use in estimating the null distribution. This also helps explain why we only infer a single (source, target) class pair based on the



(a) BD-P-S group



(b) BD-G-S group

Figure 7: Histograms of the collateral damage rate statistics for (a) the BD-P-S group and (b) the BD-G-S group.

most extreme statistic (as illustrated in Figure 2) – “collateral damage” classes, paired with the inferred target class, may be prone to false detection, and it is not so easy to distinguish the “collateral damage” phenomenon for a single source attack from a truly multiple-source backdoor attack. We do not view this as a true limitation of our detection method – our method should be able to accurately identify multiple true source classes (using the second to the $(K - 1)$ -th largest order statistics) *if* the backdoor attack is more surgical, *i.e.* if it does not induce collateral damage to classes not explicitly attacked.

5.2.2 Detection Performance Evaluation

We first evaluate the performance of the proposed AD framework and compare with that of the NC approach, using the four groups of classifiers. The clean data set used for detection (for both AD and NC) for each classifier contains 1000 images (100 per class) randomly sampled from the clean test set held out from training. The combinations of the objective functions for perturbation optimization and the detection inference strategies to be evaluated are as follows:

- **AD-J-P**: Eq. (3) objective combined with the principal detection inference approach.
- **AD-Jp-P**: Eq. (5) objective combined with the principal detection inference approach.
- **AD-Jc-P**: Eq. (6) objective combined with the principal detection inference approach.
- **AD-J-C**: Eq. (3) objective combined with the confusion-corrected detection inference approach described in Section 3.3.

- **AD-J-P-L1:** As the other variants take $d(\cdot)$ to be the L2 norm, this variant is the same as AD-J-P except using the L1 norm for detection purposes.

For all the variants above, we set π , the target misclassification fraction, to be 0.8. However, we have observed that detection performance is not sensitive to the choice of π (shown in Section 5.3.1). For the corrected detection inference used with the AD-J-C variant, we selected the order of the polynomial as $M = 3$. Again, other choices of M have little impact on detection accuracy – $M = 2, 4, 5$ gave similar results. The only remaining hyperparameter to be selected for all the variants is the detection threshold θ on the p-values. θ can be set to fix the theoretical false detection rate. Here we use the “classical” statistical significance threshold $\theta = 0.05$ as the default, which also matches with the choice of NC in terms of the significance level. We will also evaluate performance for more conservative and liberal choices of θ later in the current section.

For NC, minimizing (9) requires specifying the Lagrange multiplier λ . We tested both $\lambda = 1.5$ and $\lambda = 1.0$.¹⁶ We conjecture (without testing) that if λ is selected too large, perturbation optimization for NC may never reach the π -level group misclassification target. For NC, we used the same $\pi = 0.8$ target as for the proposed AD framework; we also evaluated $\pi = 0.5$. However, we found for some classifiers that inducing π -level misclassification to a subset of putative target classes from all source classes (i.e. NC’s objective) is not feasible. For these classes, the optimization process is terminated when the size/norm of the perturbation mask reaches a pre-set upper bound value. When this occurs, the derived detection statistics for these problematic classes skew estimation of the median during NC inference. In the experiments, we discarded these abnormal perturbation statistics in order to make NC inference more robust. For our proposed AD framework, this upper bound on the perturbation size/energy was never reached in our experiments – π -level group misclassification was always achieved by our method. Even if the upper bound were to be reached for several class pairs, taking the reciprocal will send these large statistics close to zero and have little effect on the estimation of the tail of the null density. The variants of NC detection that we evaluated are:

¹⁶The default is $\lambda = 1.5$, see “<https://github.com/bolunwang/backdoor>”.

- **NC-L1-1.5:** L1-regularized objective function with $\lambda = 1.5$. The anomaly indices are derived from the L1 norm of each optimized perturbation.
- **NC-L2-1.5:** L2-regularized objective function with $\lambda = 1.5$. The anomaly indices are derived from the L2 norm of each optimized perturbation.
- **NC-L1-1.0:** L1-regularized objective function with $\lambda = 1.0$.
- **NC-L1-1.5-0.5:** Same as NC-L1-1.5, except that $\pi = 0.5$ instead of $\pi = 0.8$.

Table 2: Detection accuracy of all the variants of the proposed AD framework (with detection threshold $\theta = 0.05$) and of the NC approach for the four groups of DNN classifiers for CIFAR-10.

	BD-P-S	BD-G-S	BD-G-M	Clean
AD-J-P	0.92	0.92	1.00	1.00
AD-Jp-P	0.92	0.92	1.00	1.00
AD-Jc-P	0.96	0.92	1.00	1.00
AD-J-C	0.96	0.92	1.00	1.00
AD-J-P-L1	1.00	0.92	1.00	1.00
NC-L1-1.5	0.36	0.16	1.00	0.84
NC-L2-1.5	0.56	0.64	1.00	0.72
NC-L1-1.0	0.40	0.28	1.00	0.76
NC-L1-1.5-0.5	0.36	0.16	0.88	0.96

In Table 2, detection accuracy of each variant of the proposed AD and NC detection methods for each group of classifiers is shown. Accuracy for the groups of classifiers under attack, i.e. BD-P-S, BD-G-S and BD-G-M, is defined as the fraction of classifiers successfully detected as attacked. For the proposed AD variants, a successful detection *also* requires the detected source and target class pair (corresponding to the most extreme detection statistic) to be ground-truth involved in the attack. For NC, since it assumes all classes except for the target class are involved in the backdoor attack, there is no inference on the source class(es). In our experiments, we thus consider NC detection to be successful if the class label corresponding to the most extreme anomaly index, if detected (i.e. above θ_{MAD}), is the true backdoor target class label t^* . Thus, for BD-P-S and BD-G-S, our AD approach meets a stronger true

detection requirement than NC. For the group of clean networks, the detection accuracy is defined as the fraction of networks inferred to *not* be attacked.

All proposed AD variants achieved perfect detection for the BD-G-M and Clean group experiments. Detection accuracy for the BD-P-S and BD-G-S groups is also very high (all above 0.90). We note that we can always make more conservative AD inferences and claim a successful detection when *only* the target class is correctly inferred. This is reasonable due to the collateral damage effect shown previously. Allowing such conservative inference, all five AD variants achieve perfect detection for the BD-G-S group. In fact, all incorrect source classes by AD variants for the results shown in Table 2 have high class collateral damage rate (above 88.4%).

NC detection, as expected, achieves strong performance for the BD-G-M group (except for the NC-L1-1.5-0.5 variant) since it is designed for backdoor attacks that involve all $(K - 1)$ source classes. However, all the variants of NC are not effective at detecting backdoor attacks involving a single source class, as shown in Table 2 by the accuracy for BD-P-S and BD-G-S. Also, the detection power of NC cannot be improved by choosing a smaller θ_{MAD} , since the false detection rate on clean classifiers is then made quite non-negligible. In Figure 8, we show the histogram of the maximum anomaly indices for BD-G-S and Clean, obtained using NC-L1-1.5 – these two groups of anomaly indices are not clearly separable by *any* choice of θ_{MAD} .

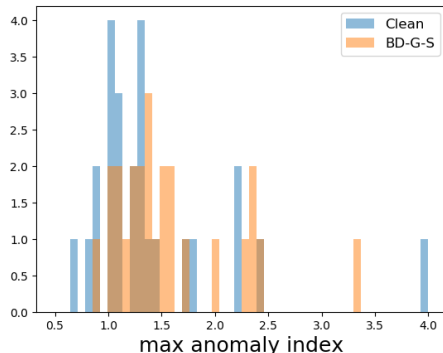


Figure 8: Histogram of the maximum anomaly indices for the BD-G-S and Clean groups, obtained using NC-L1-1.5.

Unlike NC, for the proposed AD variants, the order statistic p-value for clean classifiers and

for attacked classifiers are easily separable in practice – there is a large range of thresholds θ which yield the same (or very similar) performance. In Table 3, we show the detection performance for a very liberal detection threshold $\theta = 0.2$. We can observe a slight increment in false detection rate for the Clean group. Simultaneously, the true detection rate for the BD-P-S group is slightly increased. If we use a very conservative detection threshold $\theta = 0.01$, as shown in Table 4, only the true detection rate for the BD-P-S group is reduced slightly. Moreover, we find from Table 2 and Table 3 that the false detection rate for the Clean group is much lower than its expectation, which equals the detection threshold θ . This is because, for the classifiers in the Clean group, the distribution of the reciprocal statistics used for detection is not left-skewed like a typical Gamma distribution – few reciprocals sit on the tail of the unconditional null distribution. Hence the order statistic p-value follows a slightly right-skewed distribution rather than a truly uniform distribution on $[0, 1]$. For example, the mean of the order statistic p-value across the 25 classifier realizations from the Clean group is 0.590 when the AD-J-P variant is applied for detection.

Table 3: Detection accuracy of all the variants of the proposed AD framework with detection threshold $\theta = 0.2$ for the four groups of DNN classifiers.

	BD-P-S	BD-G-S	BD-G-M	Clean
AD-J-P	0.96	0.92	1.00	1.00
AD-Jp-P	0.96	0.92	1.00	0.96
AD-Jc-P	0.96	0.92	1.00	0.96
AD-J-C	0.96	0.92	1.00	1.00
AD-J-P-L1	1.00	0.92	1.00	0.92

Table 4: Detection accuracy of all the variants of the proposed AD framework with detection threshold $\theta = 0.01$ for the four groups of DNN classifiers.

	BD-P-S	BD-G-S	BD-G-M	Clean
AD-J-P	0.84	0.92	1.00	1.00
AD-Jp-P	0.88	0.92	1.00	1.00
AD-Jc-P	0.88	0.92	1.00	1.00
AD-J-C	0.96	0.92	1.00	1.00
AD-J-P-L1	1.00	0.92	1.00	1.00

In addition to accurate attack detection inference, our detection approach also gives an estimate of the ground truth backdoor pattern. The left figure of Figure 9 is an estimate of

one of the 25 4-pixel perturbations used to create the BD-P-S group. A 0.5 offset is added for visualization. The ground truth backdoor pattern is the one on the left of Figure 4. Comparing with the ground truth backdoor pattern, we can see that instead of perturbing all four pixels, perturbing the region near the ground truth perturbed pixel on the top left is most effective to induce group misclassification to the target class. The right figure of Figure 9 is an estimate of one of the 25 global perturbations used to create the BD-G-S group. Since nearly half of the pixels are perturbed negatively, and the perturbation size is too small to be visualized, we first add an offset such that the resulting perturbations are all positive; then the shifted perturbation mask is scaled by 30 times. *Clearly, we recover a “chess board” pattern similar to the ground truth backdoor pattern.*



Figure 9: Examples of estimated backdoor patterns with ground truth a 4-pixel perturbation (left) and a global perturbation (right), respectively. The estimated global perturbation mask has been scaled by 30 times to be human visualizable.

Finally, we evaluate FP using the classifiers from BD-P-S, BD-G-S and BD-G-M. We demonstrate the main weakness of FP that there is no simple “dichotomization” of neurons, with most *solely* dedicated to “normal” operation and some *solely* dedicated to implementing the backdoor. The technical details and the results are deferred to the supplementary material due to space limitations.

5.2.3 Generalized AD approach

Our AD approach can be easily generalized, without knowledge of the attacker’s mechanism, to detect imperceptible backdoors with various perturbative mechanisms. Note that most DNNs $f(\cdot)$ can be decomposed as two consecutive sub-models $f_1 \circ f_2(\cdot)$, such that $f_1(\cdot) : \mathcal{X} \rightarrow \mathcal{Z}$ maps an image $\underline{x}$ to its internal layer features $\underline{z} = f_1(\underline{x}) \in \mathcal{Z}$, and $f_2(\cdot) : \mathcal{Z} \rightarrow \mathcal{C}$ maps $\underline{z}$ to the predicted class label. Our AD framework can be generalized, taking AD-J-P

as an example, by replacing Eq. (3) with the generalized objective function

$$J_{st}^{\mathcal{G}}(\underline{v}) = -\frac{1}{|\mathcal{D}_s|} \sum_{\underline{x} \in \mathcal{D}_s} p_{\mathcal{G}}(t|f_1(\underline{x}) + \underline{v}), \quad (10)$$

where $p_{\mathcal{G}}(\cdot|\underline{z})$ is the DNN’s posterior given the internal layer feature vector $\underline{z}$. That is, for each class pair (s, t) , we search the minimal-sized perturbation that induces high misclassification rate from s to t in the internal feature space $\mathcal{Z}$ instead of the image space $\mathcal{X}$.

By applying AD-J-P with the generalized objective function Eq. (10), with $f_1(\cdot)$ the first convolutional layer, on the classifiers from the BD-P-S, BD-G-S, and BD-G-M group, we obtain detection rates 0.88, **1.00**, and **1.00**, respectively. For the clean group, we detects **no attacks** for all the 25 classifiers.

We further consider a multiplicative perturbation $\underline{u}^*$ that can be applied to a clean image $\underline{x}$ by

$$B_{\text{multi}}(\underline{x}, \underline{u}^*) = [\underline{x} \odot \underline{u}^*]_c,$$

where $\underline{u}^*$ has the same dimension as the image $\underline{x}$, with u_{ijk} the scaling factor of x_{ijk} . Here, we create a backdoor attack with a multiplicative chessboard pattern – we set $u_{ijk} = 1$ if pixel x_{ijk} is not perturbed and set $u_{ijk} = 1.02$ if pixel x_{ijk} is perturbed. Other configurations of the attack are the same as the attacks on the classifiers in the BD-G-S group.

In Figure 10, we show the histogram of the reciprocal statistics obtained by applying AD-J-P with the generalized objective function Eq. (10) on the attack above. We again choose $f_1(\cdot)$ as the first convolutional layer. The reciprocals corresponding to the true target classes are clearly, abnormally large. Our generalized AD successfully detects the attack (with order statistic p-value $2.0 \times 10^{-13} \ll \theta = 0.05$) with correct inference of the (source, target) class pair. In the supplementary material, we show that our generalized AD is also effective for attacks using the local patch replacement backdoor embedding mechanism assumed by NC (i.e. Eq. (8)). While both generalized and the original additive perturbation AD approach work well for attacks with multiplicative perturbations, for attacks with the local patch replacement backdoor embedding mechanism, only the generalized AD approach is successful.

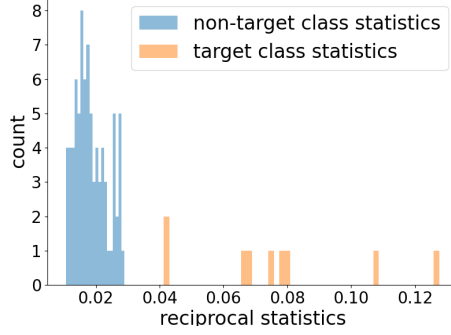


Figure 10: Histogram of the reciprocal statistics for AD-J-P with the generalized objective function applied on an attack with multiplicative chessboard pattern.

5.3 Other Design Choices

5.3.1 Choice of π

Unlike NC, the proposed detection approach is not very sensitive to the choice of π . As an example, we evaluate the detection accuracy of the AD-J-P variant on the 25 classifiers in the BD-P-S group, with a range of choices of π from 0.4 to 0.9. We use the default detection threshold $\theta = 0.05$. As shown in Figure 11, the detection accuracy does increase, but not dramatically with π . The minimum detection accuracy across the range of choices of π is 0.8, which can be further improved by use of a more liberal detection threshold. This phenomenon can be understood from Figure 12, in which we plot the sequences of $(\|\underline{v}_{st}^{(\tau)}\|_2, \rho_{st}^{(\tau)})$ for all (s, t) pairs during the perturbation optimization using Algorithm 1 with $\pi = 0.9$, while applying AD-J-P on an example classifier realization in the BD-P-S group. The sequence corresponding to the ground truth backdoor class pair (s^*, t^*) is represented using red crosses, and is clearly separated from the sequences for the non-backdoor pairs. In other words, there is a huge range for the choice of π (not exceeding 0.9) to achieve correct detection for this example. For any choice of π in such range, the $(\|\underline{v}_{st}^{(\tau)}\|_2, \rho_{st}^{(\tau)})$ sequence for each (s, t) pair is truncated at the minimum τ such that $\rho_{st}^{(\tau)}$ first exceeds π . The resulting $\|\underline{v}_{st}^{(\tau)}\|_2$ for the backdoor class pair (s^*, t^*) , as can be seen from the figure, will be much smaller than those for the non-backdoor pairs, which will lead to a successful detection.

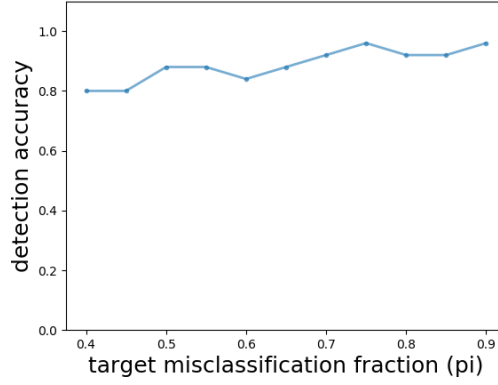


Figure 11: Detection accuracy of AD-J-P on the BD-P-S group, with a range of choices of π .

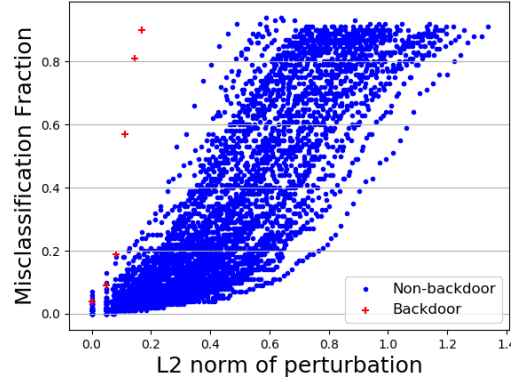


Figure 12: Sequences of $(\|\underline{v}_{st}^{(\tau)}\|_2, \rho_{st}^{(\tau)})$ for all (s, t) pairs, including the ground truth backdoor pair (s^*, t^*) represented using red crosses, during the execution of Algorithm 1 with $\pi = 0.9$, while applying AD-J-P on an example classifier realization in the BD-P-S group.

5.3.2 Size of Clean Data Set

Here we test the impact on detection accuracy of the size of the clean data set available to the defender. Again, we apply AD-J-P detection (with $\theta = 0.05$) to the classifiers in the BD-P-S group, but with the number of clean images per class used for detection equaling 5, 10, 25, 50, 100 for each trial, respectively. As shown in Figure 13, the detection accuracy increases as the number of clean images per class grows, approaching a stably high value when the number of clean images per class is greater than 25. Note that detection accuracy greater than 80% is achieved with just 10 clean samples per class. Comparing with the size of the training set (5000 images per class) and the number of backdoor images required to launch

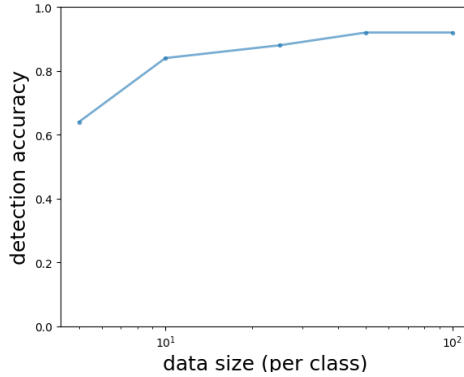


Figure 13: Detection accuracy of AD-J-P on BD-P-S for a range of sizes of the clean data set.

the attack (1000 images in total¹⁷), the detection cost is “cheap” in terms of the required number of clean images. Even if there are abundant clean images available to the defender, he/she can always use only a subset of images to reduce the defender’s computational effort.

5.3.3 Fitting Null Distribution Using All Statistics

As described in Section 3.1.3, during the detection inference process, we fit a conditional null density using the $(K - 1)^2$ smallest statistics. One may also consider a naive approach that fits an unconditional null using all detection statistics. In Table 5, we show the detection accuracy of the AD-J-P variant with detection threshold $\theta = 0.05$ for the four groups of DNN classifiers based on this naive null learning. Clearly, there is severe degradation in accuracy of detecting backdoors.

5.3.4 Size/Energy of the Ground Truth Backdoor Pattern

Here we create 25 classifier realizations using the same setting as for the BD-G-S group, except that the L2 norm of the backdoor pattern is set to 1.2 – much larger than required to

¹⁷For backdoor patterns with innocuous objects, poisoning the training set with a few to a few tens of backdoor images yields a successful attack [3]. For backdoor patterns with human-imperceptible perturbations, hundreds of backdoor images should be used for poisoning the training set to make the attack successful [26].

Table 5: Detection accuracy of the AD-J-P variant with detection threshold $\theta = 0.05$ for the four groups of DNN classifiers, where the null density is estimated using all $K(K - 1)$ reciprocal statistics.

	BD-P-S	BD-G-S	BD-G-M	Clean
detection acc.	0.48	0.44	0.52	1.00

launch a successful backdoor attack (and visualizable by carefully scrutinizing the image). When applying AD-J-P for detection, with the detection threshold $\theta = 0.05$, the detection accuracy reaches 1.00. More importantly, the estimated backdoor patterns for the 25 classifier realizations are all scaled “chess board” patterns with L2 norm around 0.101 ± 0.017 , while the L2 norm of the optimized perturbation for all non-backdoor class pairs across the 25 realizations is 1.243 ± 0.321 . That is, the “estimated” backdoor pattern will have the *minimum* norm necessary to achieve the π target level, even if the actual backdoor pattern has much larger norm. Regardless, the estimated backdoor pattern *is* accurate (it is a “chess board” pattern). Also, from this experiment, we note that the proposed detector does not require the size/energy of the ground truth backdoor pattern to be significantly smaller than the size/energy of perturbations required to induce group misclassification for non-backdoor class pairs.

5.4 Computational Complexity

The computation in this work is performed on an Amazon EC2 g3s.xlarge virtual machines powered by a NVIDIA Tesla M60 GPU. When applying AD-J-P to the BD-P-S, the average running time across the 25 classifier realizations is 698s. In comparison, when applying NC-L1-1.5 to BD-P-S, the average running time across the 25 classifier realizations is 912s. For both approaches, the perturbation optimization step incurs most of the computational cost. For the NC defense, each iteration of perturbation optimization requires $2(K - 1)N$ back-propagations on the DNN, where N is the number of clean images per class used for detection. Since NC solves K perturbation optimization problems, the total computational cost is $2(K - 1)NTK$ back-propagations, where T is the number of iterations required to reach

π -level misclassification. For our AD framework, each iteration of perturbation optimization requires $2N$ back-propagations on the DNN since we only use clean images from a single class for detection. The total computational cost is then $2NTK(K - 1)$ since we solve $K(K - 1)$ perturbation optimization problems. Theoretically, the computational cost of the two defenses are the same. But in practice, the difficulty of inducing group misclassification to a target class from all the other classes can be much higher than inducing misclassification from a single source class to the target class, especially when the backdoor attack is crafted with a single source class. Therefore, the number of iterations in perturbation optimization required for the NC defense is generally larger than that for our proposed defense.

6 Conclusions and Future Work

In this work, we developed a purely unsupervised AD defense that detects imperceptible backdoor attacks in DNN classifiers post-training. We tested multiple variants of our AD framework and compared them with other existing post-training defenses for several backdoor patterns, data sets (shown in the supplementary material), and attack settings. Our defense was experimentally shown to be more effective and robust over a range of detection scenarios.

There are several directly related issues that have been considered and can be further explored. First, although we proposed a correction scheme accounting for class pairs with high confusion, it is possible for two classes to be very similar but for their confusion to be low. In such case, the minimum perturbation size required to induce group misclassification for such class pairs may be as small as for a true backdoor pair. We believe it is possible to devise an extension of our scheme that can distinguish such false pairs from true ones. Second, our proposed defense focuses on detecting backdoor attacks with human-imperceptible backdoor patterns. If the backdoor pattern is an innocuous object in the image (*e.g.* a tennis ball in a dog image or glasses on a face), the norm of the backdoor pattern might be much larger than the average norm of the minimum perturbation required to induce group misclassification for non-backdoor class pairs. Such backdoors violate our fundamental assumption about the attacker and require an alternative detection strategy. In fact, we have recently devised a quite effective approach for this case [31]. Third, applying perturbation optimization at

an internal layer, in addition to addressing more general backdoor attack mechanisms, also address other domains – particularly where the inputs to the DNN are *discrete-valued* (*e.g.*, text) and, thus, for which gradient-based optimization of a (continuous-valued) perturbation cannot be applied. Fourth, backdoor attacks and defenses can be considered for classifiers other than DNNs, *e.g.* support vector machines.

There are several other related topics that can be further explored. First, while our cost function minimization was proposed as part of a backdoor attack detection scheme, it can also in principle be applied to provide *interpretability* to existing (unattacked) DNN classifier solutions. In particular, our approach may identify a class pair (s, t) that is highly confusable in the presence of an unusually small perturbation. This is revealing of a possible hidden fragility of the learned classifier. Moreover, it is possible that retraining with an augmented training set that includes perturbed examples from class s labeled as class s may help to remedy such fragilities. Second, a defense “in-flight” can be developed to identify backdoor patterns during the operation of the classifier. For example, the estimated backdoor pattern revealed by our approach may be used to correlate against test patterns. Third, we can develop defenses against backdoor attacks while assuming a much stealthier attacker. For example, the attacker may be able to optimize the backdoor pattern, to make the backdoor less detectable. Or we may study defenses against “white box” attacks where the attack is assumed to be aware of the deployed defense. Moreover, a more surgical attack on (s^*, t^*) may be devised by inserting images crafted using clean images from all classes other than s^* or t^* , added with the backdoor pattern and *correctly* labeled, into the training set together with backdoor images to minimize “collateral damage”.

References

- [1] H. Pirsiavash A. Saha, A. Subramanya. Hidden trigger backdoor attacks. In *AAAI*, 2020.
- [2] 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. <http://arxiv.org/abs/1811.03728>, Nov. 8, 2018.
- [3] X. Chen, C. Liu, B. Li, K. Lu, and D. Song. Targeted backdoor attacks on deep learning systems using data poisoning. <https://arxiv.org/abs/1712.05526v1>, 2017.
 - [4] J. Ba D. P. Kingma. Adam: A method for stochastic optimization. In *ICLR*, 2015.
 - [5] J. Dai and C. Chen. A backdoor attack against lstm-based text classification systems. <https://arxiv.org/abs/1905.12457>, 2019.
 - [6] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern classification*. Wiley, 2001.
 - [7] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *Proc. ICLR*, 2015.
 - [8] T. Gu, B. Dolan-Gavitt, and S. Garg. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. <https://arxiv.org/abs/1708.06733>, Mar. 2019.
 - [9] 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. <https://arxiv.org/abs/1908.01763>, 2019.
 - [10] F. R. Hampel. The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, 69, 1974.
 - [11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016.
 - [12] L. Huang, A.D. Joseph, B. Nelson, B.I.P. Rubinstein, and J.D. Tygar. Adversarial machine learning. In *Proc. 4th ACM Workshop on Artificial Intelligence and Security (AISec)*, 2011.
 - [13] B.-H. Juang and S. Katigiri. Discriminative learning for minimum error classification. *IEEE Trans. on Signal Processing*, 1992.
 - [14] C. Liao, H. Zhong, A. Squicciarini, S. Zhu, and D.J. Miller. Backdoor embedding in convolutional neural network models via invisible perturbation. CoRR, 2018.

- [15] K. Liu, B. Doan-Gavitt, and S. Garg. Fine-Pruning: Defending Against Backdoor Attacks on Deep Neural Networks. In *Proc. RAID*, 2018.
- [16] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, and Zhai J. Trojaning attack on neural networks. In *Proc. NDSS*, San Diego, CA, Feb. 2018.
- [17] D.J. Miller, Y. Wang, and G. Kesidis. When Not to Classify: Anomaly Detection of Attacks (ADA) on DNN classifiers at test time. *Neural Computation*, 31, 2019.
- [18] D.J. Miller, Z. Xiang, and G. Kesidis. Adversarial learning in statistical classification: A comprehensive review of defenses against attacks. <https://arxiv.org/abs/1904.06292>, 2019.
- [19] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2017.
- [20] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. DeepFool: a simple and accurate method to fool deep neural networks, 2016.
- [21] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. Celik, and A. Swami. Practical black box attacks against machine learning. In *Proc. Asia CCS*, 2017.
- [22] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *Proc. 1st IEEE European Symp. on Security and Privacy*, 2016.
- [23] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.
- [24] C. Szegedy, W. Zaremba, I Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *Proc. ICLR*, 2014.
- [25] F. Tamer, F. Zhang, A. Juels, M. Reiter, and T. Ristenpart. Stealing machine learning models via prediction apis. In *Proc. USENIX Security Symposium*, 2016.
- [26] B. Tran, J. Li, and A. Madry. Spectral signatures in backdoor attacks. In *Proc. NIPS*, 2018.

- [27] Trojans in artificial intelligence (TrojAI). <https://www.iarpa.gov/index.php/research-programs/trojai>.
- [28] 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 *Proc. IEEE Symposium on Security and Privacy*, 2019.
- [29] Y. Wang, D.J. Miller, and G. Kesidis. When Not to Classify: Detection of Reverse Engineering Attacks on DNN Image Classifiers. In *Proc. IEEE ICASSP*, Brighton, UK, May 2019.
- [30] Z. Xiang, D. J. Miller, and G. Kesidis. Revealing backdoors, post-training, in dnn classifiers via novel inference on optimized perturbations inducing group misclassification. In *ICASSP*, pages 3827–3831, 2020.
- [31] Z. Xiang, D. J. Miller, and G. Kesidis. Revealing Perceptible Backdoors, without the Training Set, via the Maximum Achievable Misclassification Fraction Statistic. In *Proc. IEEE MLSP*, Oct. 2020.
- [32] Z. Xiang, D.J. Miller, and G. Kesidis. A Benchmark Study of Backdoor Data Poisoning Defenses for Deep Neural Network Classifiers and A Novel Defense Only Legitimate Samples. In *Proc. IEEE MLSP*, Pittsburgh, Oct. 2019.
- [33] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, and F. Roli. Support vector machines under adversarial label contamination. *Neurocomputing*, 160(C):53–62, July 2015.
- [34] X. Yuan, P. He, Q. Zhu, and X. Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9):2805–2824, Sep. 2019.

Revealing Perceptible Backdoors in DNNs, Without the Training Set, via the Maximum Achievable Misclassification Fraction Statistic*

Zhen Xiang, David J. Miller, Hang Wang, and George Kesidis

School of EECS, The Pennsylvania State University

University Park, PA 16802 USA

{zux49,djm25,hzw81,gik2}@psu.edu

April 7, 2020

Abstract

Recently, a backdoor data poisoning attack was proposed, which adds mislabeled examples to the training set, with an embedded backdoor pattern, aiming to have the classifier learn to classify to a target class whenever the backdoor pattern is present in a test sample. Here, we address *post-training* detection of innocuous *perceptible* backdoors in DNN image classifiers, wherein the defender does *not* have access to the poisoned training set, but only to the trained classifier, as well as unpoisoned examples. This problem is challenging because without the poisoned training set, we have no hint about the actual backdoor pattern used during training. This post-training scenario is also of great import because in many practical contexts the DNN user did not train the DNN and does not have access to the training data. We identify two important properties of perceptible backdoor patterns – spatial invariance and robustness – based upon which we propose a novel

*This research supported by an AFOSR DDDAS grant and a Cisco Systems URP gift.

detector using the *maximum achievable misclassification fraction* (MAMF) statistic. We detect whether the trained DNN has been backdoor-attacked and infer the source and target classes. Our detector outperforms other existing detectors and, coupled with an *imperceptible* backdoor detector, helps achieve post-training detection of all evasive backdoors.

1 Introduction

Deep neural network (DNN) classifiers have achieved state-of-the-art pattern recognition performance in many research areas such as speech recognition [1], bioinformatics [2], and computer vision [3][4]. However, they have also been shown to be vulnerable to adversarial attacks [5]. This has inspired adversarial learning research, including work in devising both formidable attacks as well as defenses against same [6]. Test-time evasion (TTE) is a prominent type of adversarial attack aiming to induce misclassifications during operation/test-time by modifying test samples in a human-imperceptible (or machine-evasive) fashion, e.g. [7][8][9][10][11]. Another type of attack, data poisoning (DP), inserts malicious samples into the training set, *often* to degrade the classifier’s accuracy [12][13].

Recently, a new form of *backdoor* DP attack, usually against DNN image classifiers, was proposed, aiming to “sneak” a backdoor mapping into the learning of the DNN, while *not* degrading the performance in accurately classifying clean test samples [14][15][16]. A learned backdoor can be easily achieved by inserting a relatively small number of poisoned images into the training set¹. Backdoor training images are crafted by embedding the same backdoor pattern into legitimate (clean) images from one or more source classes, and (mis)labeling to a target class (selected by the attacker). The backdoor pattern, principally designed to be evasive to possible occasional human inspection of the training set, could be: 1) a human-imperceptible, additive perturbation applied to clean images (dubbed here an *imperceptible* backdoor pattern)

¹The attackers poisoning capability is facilitated by the need in practice to obtain “big data” suitable for accurately training a DNN for a given domain – to do so, one may need to seek data from as many sources as possible (some of which could be attackers).

[17][18][19][20]; 2) a seemingly innocuous object in a scene (dubbed here a *perceptible* backdoor pattern), e.g. a bird flying in the sky or glasses on a face [14][16][21]. The DNN trained on the poisoned training set will still correctly classify clean test images with high accuracy (because the number of poisoned samples is relatively small); hence, validation set accuracy degradation, e.g. [22], *cannot* be reliably used as a basis for detecting backdoors. However, when faced with a test image containing the same backdoor pattern used in training, the DNN is likely to misclassify to the target class prescribed by the attacker.

Defenses against backdoor attacks can be deployed before/during training, post-training, or potentially in-flight (i.e. at operational/test-time).² In the before/during-training scenario, the defender has access to the (possibly poisoned) training set and the trained classifier [18][23][19]. The defender's goals are to detect whether the training set has been poisoned and, if so, to identify and remove the embedded backdoor training images before (re)training. In the post-training scenario considered here, however, the defender has access to the trained DNN but *not* to the training set used for its learning [24][25][21][20]. This post-training scenario is of strong interest because there are many pure *consumers* of machine learning systems. For example, critical infrastructure is often based on *legacy* system classifiers. In such a scenario, it is very possible the original training data used to build the classifier is unavailable. Likewise, an app may be used on millions of cell phones. The app user will not have access to the training set on which the app's classifier was learned. Still, the user would like to know whether the app's classifier has been backdoor-poisoned. The user is assumed to possess a clean labeled dataset with examples from each of the classes in the domain on which, e.g., it can evaluate the performance of the DNN. The clean dataset is also assumed to be relatively small (this is reasonable, as the user may not have access to either substantial training data nor to computational resources necessary for DNN training); hence, while useful for building a backdoor detector, it is not sufficient for training a surrogate DNN (i.e. one without a backdoor present). The defender's fundamental goal in the post-training scenario is to detect whether the DNN has been backdoor-poisoned or not. If a detection is made, the source class(es) and the target class should also be inferred, so

²There are currently no published approaches for in-flight detection, to our knowledge.

that decisions made to classes *not* involved in the attack could still possibly be trusted when there are no replacement DNNs available.

In this paper, we focus on *post-training* detection of backdoor attacks with *perceptible* backdoor patterns, which is a very challenging problem with no well-founded method proposed yet. Elsewhere, post-training detection of *imperceptible* backdoors is addressed [20]. Together, our work and the work for the imperceptible case [20] “cover” all cases of interests, providing a complete solution to post-training detection of evasive backdoors. Our main contributions are as follows:

1) We propose a novel approach for detecting backdoor attacks with perceptible backdoor patterns, post-training. Our detection is based on the *maximum achievable misclassification fraction* (MAMF) statistic, obtained by essentially reverse-engineering the putative backdoor pattern for each (source, target) class pair. Our detection inference uses an easily chosen threshold. No unrealistic assumptions on the shape, spatial location or object type of the backdoor pattern are made, unlike [21].

2) We identify and experimentally analyze two important properties – spatial invariance and robustness – of perceptible backdoor patterns that are applicable to general attack instances. These properties, which have neither been identified nor exploited in prior works, are the basis of our detector and may also potentially inspire other defenses.

3) We perform substantial experimental evaluation and show the strong capability of our detector, both in general and compared against the two existing methods [25][24] for the post-training detection problem we address. Our experiments involve five datasets of practical interest, five commonly used DNN structures, nine backdoor patterns of different types and two compared detection methods.

The rest of the paper is organized as follows. In Section 2, we provide a thorough and unified review of all major aspects of backdoor attacks and defenses. In section 3, we review existing post-training defenses and point out their limitations. In Section 4, we discuss two properties of perceptible backdoor patterns, based upon which we develop our detection procedure. In Section 5, we report our experiments.

2 Backdoor Attack

A backdoor attack is a recently proposed type of adversarial learning attack, usually against DNN image classifiers. A typical backdoor attacker aims to have the classifier learn to classify to a target class whenever the backdoor pattern is present in a test image, while still correctly classifying test images without the backdoor pattern. Such aims are usually achieved by poisoning the training set of the classifier using a relatively small number of images, from the attacker’s source class (or classes), with the same backdoor pattern, but labeled to the attacker’s target class.

There are many works on devising backdoor attacks and defenses against backdoor attacks have also appeared recently. Unfortunately, most of these works, especially the defenses, only focus on a narrow aspect of the problem (e.g. considering one particular type of backdoor pattern or defense scenario). In this section, we provide a broad overview of backdoor attacks and defenses. We first compare backdoor attacks with other prominent types of adversarial attacks in terms of the goals of the attacker and the assumptions required to launch the attack. Next, we discuss the design choices of backdoor attacks, e.g. the different types of backdoor patterns being considered in existing works on backdoor attacks and defenses. Lastly, since the current paper mainly focuses on *defenses* against backdoor attacks, we discuss several defense scenarios with their assumptions and goals, and then categorize major existing defenses, and their suitability under different scenarios. To our knowledge, we are the first to provide such a unified view of the field of backdoor attacks and defenses. We hope this section helps to situate our work and its novel capability, relative to existing adversarial learning defense works.

2.1 Comparison with Other Adversarial Attacks

Backdoor attacks are easily distinguishable from other prominent types of adversarial attacks, e.g. test-time evasion (TTE) attacks and traditional data poisoning (DP) attacks, because the attacker of each type has very specific goals and assumptions:

TTE attacks: The attacker aims to induce misclassification at operation/test-time by modifying clean test images in a human-imperceptible (or machine-evasive) fashion

[7][8][9][10][11]. For each test image, an *image-specific* additive perturbation with as small L_p norm as possible is produced, using the knowledge (including the structure and the learned weights) of the target classifier or a surrogate classifier that behaves similarly on the same data domain [26].

DP attacks: The attacker aims to degrade the accuracy of the classifier by poisoning its training data with malicious samples (e.g. incorrectly labeled or noisy samples) [12][13][27]. A typical DP attacker only needs knowledge of the classification domain and the ability to poison the training data, while recent works show that more powerful attacks could be devised if the knowledge of the classifier (considering a re-training scenario) is available [28]. Note that DP attacks change the trained classifier, while TTE attacks *do not* affect the classifier.

Backdoor attacks: The goals of a backdoor attacker are twofold. First, the learned classifier is supposed to classify to an *attacker-desired* target class, whenever a test image contains the *attacker-specified* backdoor pattern. Second, the trained classifier should correctly predict clean test images (without the backdoor pattern) [14][16]. The attack is launched by poisoning the training data with images containing the *same* attacker-specified backdoor pattern, with these images labeled to the attacker’s target class. Clearly, a backdoor attacker has the same knowledge and ability as a typical DP attacker; hence backdoor attacks can be viewed as a special type of DP attack [14][16][6].

Readers can refer to [6] for other types of attacks, including reverse engineering attacks and attacks on data privacy. Compared with TTE attacks, backdoor attacks require much lower cost to launch, especially for large-scale deep learning frameworks (where data is often acquired from the public or from multiple sources) – “stealing” a model (required for TTEs) should be more difficult than becoming a malicious data contributor in practice. Compared with general DP attacks, backdoor attacks are stealthier because a successful attack will not affect the prediction of clean images (without the backdoor pattern) – validation set accuracy degradation cannot be used as a basis for detecting backdoors.

2.2 Variants of Backdoor Attacks

Launching a backdoor attack requires the attacker to select a backdoor pattern, a (set of) source class(es), and a target class.

2.2.1 Choice of Backdoor Pattern

A backdoor pattern, a.k.a. a backdoor “trigger” [16] or a backdoor “key” [14], is an attacker-specified pattern used to poison the training data, and then to induce misclassification to the attacker-desired target class when embedded in clean test images. In principle, the backdoor pattern should be designed to be *evasive* to possible occasional human inspection of the training set. Also, similar to the stealthiness required by TTE attacks, a backdoor pattern, when embedded into clean test images during operation, should not be easily noticeable to humans. Although backdoor patterns considered by existing works (on both attacks and defenses) vary largely, they mainly fall into two categories: **perceptible backdoor patterns** and **imperceptible backdoor patterns**.

A perceptible backdoor pattern is usually a seemingly innocuous object in the scene, e.g. a pair of glasses on a face [14]. Perceptible patterns could be photo-shopped into digital images or physical objects in a scene that are digitally captured. A backdoor image $\tilde{\mathbf{x}}$ with a perceptible backdoor pattern $\mathbf{v}$ is crafted (digitally) from a clean image $\mathbf{x}$ as expressed mathematically as follows:

$$\tilde{\mathbf{x}} = g_P(\mathbf{x}, \mathbf{v}, \mathbf{m}) = \mathbf{x} \odot (\mathbf{1} - \mathbf{m}) + \mathbf{v} \odot \mathbf{m}, \quad (1)$$

where $\mathbf{m}$ is a 2-dimensional mask having the same spatial support as $\mathbf{x}$, with $m_{ij} \in \{0, 1\}$ for any pixel (i, j) ³. Often, the spatial region of 1’s for a mask is contiguous (e.g. Figure 1), or forms several contiguous pieces. For grayscale images, $\odot$ denotes element-wise multiplication. For colored images, the 2-dimensional mask $\mathbf{m}$ (or the “inverse mask” $(\mathbf{1} - \mathbf{m})$) is applied to *all* channels of the images. Note that the location of the 1’s for the mask is usually dependent on $\mathbf{x}$, i.e. $\mathbf{m}(\mathbf{x})$ is an image-specific mask carefully designed by the attacker to “carve” the *same* backdoor pattern into the best (suitably innocuous) location for each originally clean image (e.g., a pair of glasses

³In this paper, pixel intensity values are in $[0, 1]$.

must be carefully matched to the face). An example is shown in Figure 1, where a backdoor image used for attacking a pet breed classifier is created by embedding a tennis ball into an image from class “chihuahua”, and labeling to class “Abyssinian”.

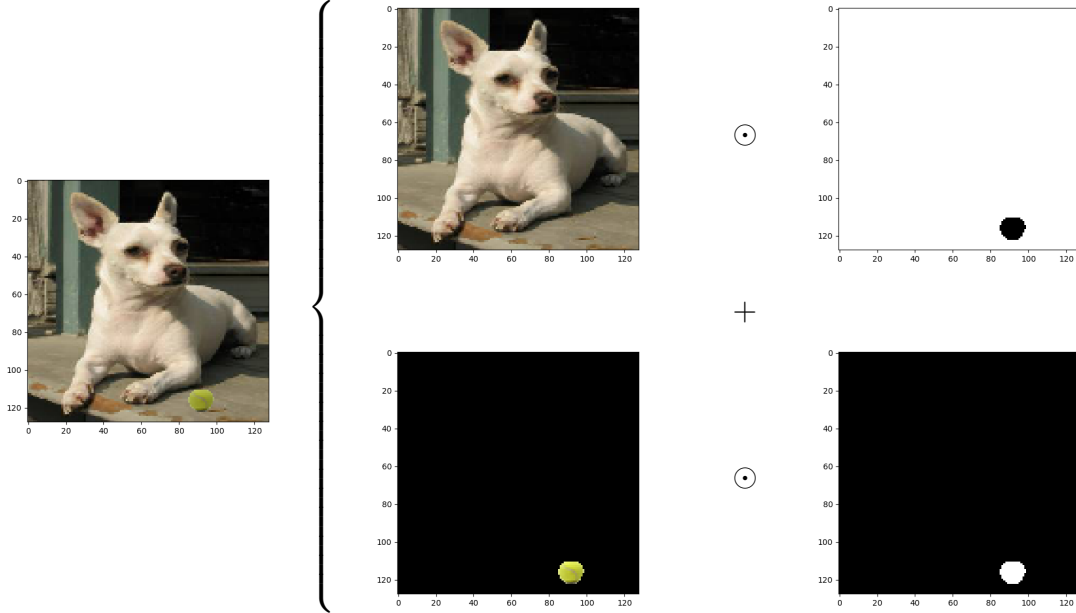


Figure 1: An example backdoor image used for attacking a pet breed classifier. A clean training image from class “chihuahua” is added with a perceptible backdoor pattern – a tennis ball – and labeled to class “Abyssinian”.

Perceptible backdoor patterns are discussed in the earliest works on backdoor attacks [14][16], where the visual-stealthiness of the embedded pattern is addressed⁴. Defenses targeting perceptible backdoor patterns have been proposed by [23][24][25][21][29] and *in the current paper*, among which the backdoor pattern used by [25] (a noise-like patch) and [21] (a firefox icon) could be easily visually identified as anomalous. In our experiments, we spent laborious human effort to choose the object used as the perceptible backdoor pattern and to embed it into the training images to make them seemingly

⁴Digitally embedding a perceptible backdoor pattern into a clean image, while keeping the backdoor-embedded image seemingly innocuous could be much more difficult than directly launching a physical attack in reality. So, in their experiments, the perceptible backdoor patterns were not necessarily carefully positioned to be most innocuous. But the objects used as the backdoor pattern were patterns with the potential to be realistically stealthy (if placed so as to be innocuous).

innocuous to humans. While ignoring the innocuousness of the backdoor pattern (e.g. [25][21]) should not affect the quality of the performance evaluation of a *defense*, which involves no human inspection, it is problematic to evaluate the defense against a backdoor attack with the perceptible backdoor pattern *spatially fixed* (e.g. to the right bottom corner of the image [25]). First, as we have mentioned, whether a perceptible backdoor pattern is truly innocuous depends on the context of the originally clean image, and hence may require careful positioning of the pattern in the image. Second, if the same backdoor pattern is embedded at the same location in all backdoor images used to poison the training set, a naive data sanitization applied “before/during training” (a defense scenario that will be described in Section 2.3) could detect the attack by (e.g.) checking the pixel value distribution. Third, and most importantly, fixing the spatial location of the backdoor pattern while creating backdoor training images will harm the robustness of the attack significantly during testing. As will be seen in Section 5.5, for a perceptible backdoor pattern fixed to the bottom left of the backdoor training images, even a single column/row shift of the backdoor pattern at test time will severely degrade the attack effectiveness.

An imperceptible backdoor pattern is usually an additive perturbation⁵, where the evasiveness (under possible human inspection) is achieved by imposing a constraint on its L_p norm. A backdoor image $\tilde{\mathbf{x}}$ is created by

$$\tilde{\mathbf{x}} = g_I(\mathbf{x}, \mathbf{u}) = [\mathbf{x} + \mathbf{u}]_c, \quad (2)$$

where

$$\|\mathbf{u}\|_p < \epsilon. \quad (3)$$

Here, $\mathbf{u}$ is the imperceptible backdoor pattern, i.e. the perturbation; and $[\cdot]_c$ is the clipping operation that guarantees the perturbed pixel intensity values are still in the valid range. Again, imperceptible backdoor patterns are specified by the attacker; and the *same* pattern (with its position fixed in the image support) is added to all the originally clean images used for poisoning the training set. Similar to the image-specific pertur-

⁵There are other formulations that are less-frequently studied to embed an imperceptible backdoor pattern into a clean image. For example, Eq. (1) with $m_{ij} \in \{0, \delta\}$ for any pixel (i, j) , where δ is very small, is proposed by [14] to embed a “watermark” into clean training images.

bation used by a TTE attacker, imperceptible backdoor patterns are more suitable for devising a digital attack than a physical attack – there is freedom to choose the pattern to be used as long as the norm constraint is not violated.

Imperceptible backdoor patterns are also widely involved in many works on both backdoor attacks and defenses. For example, a “static” imperceptible backdoor pattern, with constraint on its L_∞ norm that controls the maximum perturbation size, is experimentally evaluated by [17]. Imperceptible backdoor patterns with a constraint on the L_0 norm (i.e. with only a single or few pixels being perturbed) have been targeted by the defenses proposed in [18][19][20]. Also in [20], the proposed defense is evaluated on an image-wide imperceptible backdoor pattern with an L_2 norm constraint. An example of embedding an imperceptible backdoor pattern into a clean image used by [19] is shown in Figure 2, where a single pixel in an “airplane” image has been perturbed; the image is labeled to class “bird”.

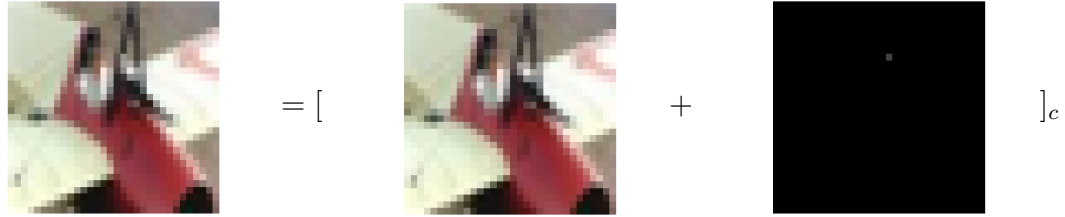


Figure 2: An example used by [19] of an imperceptible backdoor pattern, in this case a single-pixel perturbation, added to an “airplane”; and the perturbed image is labeled to class “bird”.

2.2.2 Choice of Source Class(es) and Target Class

The source class(es) and the target class involved in a backdoor attack are chosen by the attacker. In most works on backdoor attacks or defenses, an attack involves a single target class [14][15][18][23][25][20]. Backdoor attacks involving more than one target class, dubbed an “all-to-all” attack, have been discussed in [16], where the same backdoor pattern, when embedded in a clean test image from class i , is supposed to induce

a misclassification to class $(i + 1)$. However, such a backdoor attack can be decomposed into K (assuming K classes in the classification domain) attacks using the same backdoor pattern, but each involving a unique (source, target) class pair.

The number of source classes involved in a backdoor attack could range from 1 to $(K - 1)$, i.e. from a single source class (e.g. [18][23][19][20]) to all classes except for the target class (e.g. [14][16][25][20]). For attacks using an imperceptible backdoor pattern, the source classes could be arbitrarily chosen by the attacker. However, if a perceptible backdoor pattern is used, the source class(es) and the backdoor pattern must be well-matched to achieve stealthiness. For example, a bird flying in the sky could be used as a perceptible backdoor pattern when most of the source class images contain the sky; hence a class for which most of the images capture an underwater scene is not well-matched with such a backdoor pattern. Clearly, finding a perceptible backdoor pattern that is seemingly innocuous when embedded in images from *all* classes (except for the target class) is very hard in many classification domains. So, the number of source class(es) involved in a backdoor attack using a perceptible backdoor pattern depends on both the preference of the attacker and the classification domain. Moreover, for both perceptible and imperceptible backdoors, even if the attacker specifies a *single* (source, target) class pair, after training on the backdoor poisoned data, the classifier (during operation) could possibly classify images originally from a class *other than* both the source and target classes to the target class, if the image contains the same backdoor pattern used by the attacker – this phenomenon was experimentally observed in [20] and called “collateral damage”. Above all, for researchers developing defenses against backdoor attacks, it is very important to note that a powerful defense should *not* require any assumptions on the number of source classes involved in the attack.

2.2.3 Other Backdoor Attacks

While devising a backdoor attack requires only the knowledge of the classification domain and the capability to poison the training set, more powerful backdoor attacks could be devised if the structure and training settings of the classifier are known to the attacker. For example, [17] shows that knowledge of the classifier could be used to search for an

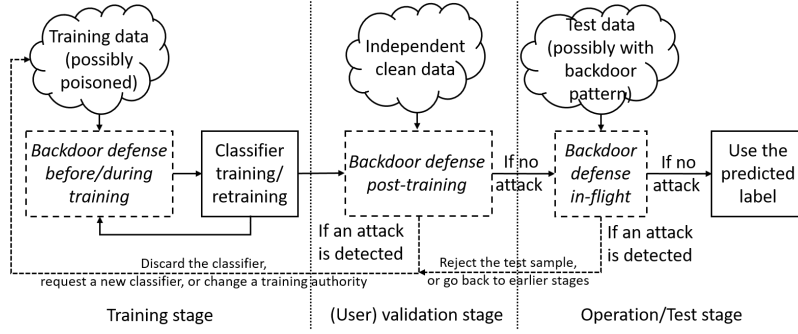


Figure 3: Illustration of backdoor defense scenarios – before/during training, post-training, and in-flight.

effective backdoor pattern with very small perturbation norm (hence, imperceptible). Such knowledge is also used by other works, e.g. [30][31][32], to design backdoor patterns that achieve stealthiness to some extent.

Recently, backdoor attacks are also studied for other learning frameworks and classification domains. For example, [33] and [34] developed backdoor attacks under the federated learning framework [35]. Backdoor attacks could also be a threat to text classification as shown by [36].

2.3 Defense Scenarios, Assumptions, and Goals

Defenses against backdoor attacks could be deployed at any stage during the entire pipeline, i.e. from the training stage to the operation stage, for a DNN classifier (Figure 3). Existing works on backdoor *defenses* mainly focus on defeating backdoor attacks **before/during training** or **post-training** (but before the operation of the classifier). Here, we describe both defense scenarios in detail, with the assumptions and goals of the defender. Although defenses designed for different scenarios *cannot* be compared directly, they can be deployed simultaneously to provide better security. Moreover, backdoor defenses designed for the same scenario can be deployed in parallel. For example, the proposed *post-training* defense in the current paper, which outperforms other post-training defenses against *perceptible* backdoors (as will be shown in Section 5.2), could be jointly deployed with the state-of-the-art post-training defense against *imperceptible* backdoors [20] to provide a complete solution to post-training defense

against *all* backdoor attacks (both imperceptible and perceptible but innocuous). We will also discuss an **in-flight** scenario, which has not been addressed by any existing works yet, where the defense could be potentially deployed during operation/use of the classifier. An advantage of such a defense is that it may identify entities attempting to exploit the backdoor.

2.3.1 Before/During Training Scenario

In the before/during training scenario, the defender, who could also be responsible for training the classifier, is assumed to have access to the (possibly poisoned) training set, and either to the training process or to the trained (attacked) classifier [18][23][19][29]. The goals of the defender are: 1) to detect if the training set has been poisoned; 2) to *correctly* identify and remove training images with the backdoor pattern before training/retraining.

As backdoor attacks are designed to be evasive against sanitization⁶ and possible human inspection of the training data, existing defenses often first train a classifier using the training data, even though the training data could possibly be poisoned. Then the trained classifier is analyzed to identify and remove suspicious training images before retraining (see Figure 3). For example, the Spectral Signature (SS) approach proposed by [18], the Activation Clustering (AC) approach proposed by [23], and the Cluster Impurity (CI) proposed by [19], all use the internal layer features (obtained from the classifier trained on the possibly poisoned training set) to separate out backdoor patterns (that are labeled to the target class) from clean images (considering all images labeled to the putative target class). The SentiNet approach proposed by [29], which targets perceptible backdoors, locates the backdoor pattern if it appears in a training image, using a combinatoric algorithm based on Grad-CAM [37].

2.3.2 Post-Training Scenario

The defender in the post-training scenario could also be the user/consumer of the trained, possibly backdoored classifier, who does not know whether the training authority is

⁶Poorly devised backdoor attacks may be defeated by sanitization of the training data.

trustworthy or not and cannot force the training authority to deploy a before/during training defense. As shown in Figure 3, the defender *does not* have access to the (possibly poisoned) training set used to train the classifier. Hence there is no clue about what backdoor patterns look like, if there is an attack, which makes post-training detection of backdoors very difficult. The defender does have access to the trained classifier (including its structure and weights), but does not have the resources (data and/or computational) to train a new classifier. The defender is also assumed to possess an independent, clean (free of backdoors), labeled dataset on which, e.g., it can evaluate the performance of the classifier. Again, the user/defender cannot train a surrogate classifier using this relatively small dataset. The goals of the defender are: 1) to detect whether the classifier has been backdoor attacked or not; 2) if an attack is detected, to infer the source class(es) and the target class involved in the backdoor attack. The second goal is very important when there are no replacement classifiers available. In such a case, if the target class involved in the backdoor attack could be correctly inferred, prediction results to classes other than the detected target class can still be trusted.

Clearly, approaches for before/during training defense (e.g. separating backdoor patterns from clean ones) cannot not be applied in the post-training scenario. Also, since backdoor attacks aim to *not* degrade accuracy on clean images, the defender cannot infer whether the classifier has been attacked solely based on the accuracy on the independent clean dataset. Existing post-training defenses include the Fine-Pruning (FP) approach proposed by [24], the Neural Cleanse approach proposed by [25], the “TABOR” approach proposed by [21], and the Anomaly Detection (AD) approach proposed by [20]. Since the current paper also proposes a post-training defense against (perceptible) backdoors, we will review these “competitors” in detail and point out their limitations in Section 3.

2.3.3 In-Flight

Defenses against backdoor attacks can potentially be deployed during the operation/use of the classifier. In this “in-flight” scenario, the defender only has access to the classifier and aims to determine whether there is a backdoor pattern embedded in a test image.

To our knowledge, no in-flight defense has been published to date. Moreover, some existing defenses against TTE attacks, e.g. [38][39], are unsuitable for in-flight detection of backdoor attacks. The reason is that these methods assume the training set that was used to design the DNN is clean (free of attack images) and that there are a large number of clean images for estimating a null model for each class (In our backdoor scenario, the user will not have a *large* set of available clean images). The “blurring” method proposed by [40], a defense against TTE attacks, is applicable for in-flight backdoor detection in principle. But we will show in Section 5.8 that it largely fails as a backdoor detector.

3 Review of Backdoor Defenses Post-Training

Before introducing our post-training defense against backdoor attacks with perceptible backdoor patterns, we review existing post-training defenses and point out their limitations. To the best of our knowledge, the FP approach proposed in [24] is the first attempt towards post-training defense against backdoors. FP, which removes neurons from the DNN, assumes that there is a simple “dichotomization” of neurons, with most *solely* dedicated to “normal” operation but with some *solely* dedicated to implementing the backdoor. Unfortunately, this assumption is not valid in many cases, as will be shown in our experiments in Section 5.2. Moreover, FP does not actually detect the presence of backdoor attacks and it removes neurons even from non-attacked DNNs.

The AD approach proposed in [20] considers post-training detection of imperceptible backdoors. For each (source, target) class pair, an optimized perturbation that induces high group misclassification fraction when added to clean source class images is obtained. If the DNN has been attacked, the optimized perturbation associated with the class pair involved in the attack will have abnormally low L_2 norm, and a detection can be made on this basis. However, this approach cannot be borrowed for detecting perceptible backdoors. First, reverse engineering a perceptible backdoor pattern (which could be any object) requires also reverse engineering the associated mask. This is more challenging compared with searching for a small additive perturbation starting from a zero (image-wide) initialized additive perturbation. Poor initialization while optimizing

the backdoor pattern *and* the mask could lead to an estimated pattern irrelevant to what was used for devising the attack and one requiring improbably large spatial support to achieve high group misclassification. Second, even if the backdoor pattern used for devising the attack could be luckily reversely engineered for the true backdoor class pair (s, t) , it could be hard to find a metric to distinguish this from estimated patterns for non-backdoor class pairs $(s', t') \neq (s, t)$, because in the perceptible case the assumption in [20] that the backdoor pattern is small in size/norm, i.e. the constraint in (3), does not hold. However, while [20] is not applicable to detecting perceptible backdoors, our proposed detection approach is inspired by the detection framework in [20]. Moreover, this detector targeting imperceptible backdoors could be deployed in parallel with the detector proposed in the current paper to detect backdoor attacks with *any* type of evasive backdoor pattern.

Perhaps closest to the current work, the NC approach proposed in [25] aims to detect perceptible backdoors post-training. NC obtains, for each putative target class, a pattern and an associated mask by optimizing an objective function that induces a high misclassification fraction when added to images *from all classes other than the target class*. If there is an attack, the L_1 norm of the obtained *mask* for the true backdoor target class will be abnormally small and a detection is made based on the median absolute deviation (MAD) [41]. However, NC relies on the assumption that *all* classes except for the target class are involved in the attack, which is usually not guaranteed to hold for attacks using perceptible backdoor patterns for the many reasons discussed in Section 2.2.2. Moreover, NC penalizes the L_1 norm of the mask while maximizing the group misclassification fraction to estimate the pattern and the mask; but the detection is sensitive to the choice of the penalizing multiplier. For CIFAR-10, unless the multiplier is carefully chosen, the results could be either a mask with low L_1 norm but not achieving the target misclassification fraction, or a wild-looking backdoor pattern with improbably large and distributed spatial support, which has no relation to the ground truth backdoor pattern (for the true target class when there is in fact an attack). We will show in our experiments (in Section 5.2) that NC is clearly outperformed by our proposed detection approach.

The “TABOR” approach proposed by [21] focuses on detecting localized backdoor

patterns post-training, which can be viewed as an extension of NC. Similar to NC, TABOR first jointly searches for a pattern and an associated mask using the same objective function as NC, but with regularization terms penalizing for perceptible backdoor patterns that are overly large, sparsely distributed, or not located at the image periphery. Then MAD, the same anomaly detection approach used by NC, is used for detection inference performed on a heuristically derived metric. Note that in general, perceptible backdoor patterns could be dispersed as will be shown in Section 5.1, or not at the image periphery, e.g. glasses on a face [14] and a sticker on a stop sign [16]. Also, TABOR makes the same unrealistic assumption about perceptible backdoor patterns as NC that all classes except for the target class are involved in the attack, based on which their metric for anomaly detection is derived. Hence in Section 5.2, we compare our proposed detection approach against the more general NC approach instead of TABOR, which makes more assumptions/limitations on the backdoor pattern used by the attacker.

4 Post-Training Detection of Perceptible Backdoors

We propose a *post-training* detector against backdoor attacks with a *perceptible* backdoor pattern. Our detector is designed for general perceptible backdoor patterns (ref. Section 2.2.1) that could be embedded into clean training images using Eq. (1). No additional assumptions about the shape, location, spatial support of the backdoor, or object to be used are made. Moreover, we do not make any assumptions about the number of source classes chosen by the attacker (ref. Section 2.2.2). The premise behind our detection approach allows it to detect backdoor attacks with the number of source classes ranging from 1 to $(K - 1)$.

Like other works on post-training defenses, our detector follows the assumptions for the post-training scenario described in Section 2.3.2: the defender has access to a relatively small clean dataset containing images from all classes; *the defender does not have access to the training set used for the DNN’s learning*; the defender does not have adequate resources to train a new classifier using a (sufficiently large) clean data set. Our detector is designed both to detect whether the classifier has been attacked or not

and to infer the source class(es) and the target class if an attack is detected.

4.1 Basis of Our Detection

Our detection is based on two properties of perceptible backdoor patterns that hold for general attack instances.

Property 1 Spatial invariance of backdoor mapping: *If the perceptible backdoor pattern is spatially distributed (not at a fixed location) in the backdoor poisoned training images (placed so as to be most innocuous in each image), the learned backdoor mapping will be spatially invariant in inducing targeted misclassifications on test images.*

As we have addressed in Section 2.2.1, the perceptible backdoor pattern could be spatially located anywhere in a backdoor training image so as to be scene-plausible. The trained DNN will then learn the backdoor pattern, but not its spatial location. This property is actually favored by the attacker, since there will be more freedom to embed the backdoor pattern in clean test images. As will be seen in Section 5.1, at test time, backdoor images with the backdoor pattern randomly located are still likely to yield misclassifications to the target class prescribed by the attacker, which well-justifies the attacker performing data poisoning consistent with satisfying this property.

Property 2 Robustness of perceptible backdoor patterns: *It is unnecessary to use exactly the same perceptible backdoor pattern at test time to induce targeted misclassifications.*

As will be seen in Section 5.3, even when strong noise is added to the backdoor pattern while creating backdoor test images, they will still likely be (mis)classified to the designated target class. More importantly, if only *part* (in terms of spatial support) of the backdoor pattern is added to the clean source class images during testing, this also induces (mis)classification to the target class. This is not surprising because perceptible backdoor patterns, like all other patterns, are learned by the DNN by extracting key features, which usually occupy a much smaller spatial support than the full backdoor pattern itself.

4.2 Detection Approach

Key ideas: 1) For (source, target) class pairs used for devising backdoor attacks, finding a pattern that induces a high misclassification fraction (from the source class to the target class) on the clean dataset can be achieved using a **relatively small** spatial support that is **arbitrarily located** in the image. 2) For non-backdoor class pairs, **much larger** spatial support is required to find a pattern that induces a high misclassification fraction from the source class to the target class.

The key ideas of our detection are well supported by the properties in Section 4.1. Based on Property 1, since the learned backdoor mapping is spatially invariant, it should be possible to reverse engineer (estimate) a backdoor pattern (inducing high group misclassification) for the backdoor class pair by choosing the center of the estimated pattern’s spatial support to be *anywhere* in the image (except at the image border, where the pattern’s full spatial support is not contained). Thus, it does not matter where we place the estimated pattern’s spatial support – in our experiments we fixed a location for all clean images used in our detection system, but it is even possible that one could vary the center position for each clean image – accurate backdoor detection should still be possible.

According to Property 2, a pattern that effectively induces a high group misclassification fraction for the (source, target) backdoor class pair could be spatially smaller than the true backdoor pattern, as long as sufficiently “representative” features of the backdoor can be captured by this restricted spatial support – for the same (relatively small) spatial support, finding a pattern that induces a high misclassification fraction for the backdoor class pair is much more achievable than for non-backdoor pairs. This crucial property, which allow us to accurately identify the source and target classes involved in an attack, will be borne out experimentally in the sequel.

Our two-step detection procedure is as follows:

1) *Pattern Estimation:* Given a K -class DNN classifier to be inspected, for *each* of the $K(K - 1)$ (source, target) class pairs, we perform pattern estimation separately on each of L square-shaped⁷ spatial supports (increasing in size). Each square spa-

⁷Other shapes, e.g. rectangular or circle, could also be used.

tial support, with a location arbitrarily chosen⁸ and *fixed* for all clean images used for detection, has a specific (absolute) width denoted by $w \in \mathbb{Z}^+$, such that $w \in \lceil r_{\min} \times W \rceil, \lceil r_{\max} \times W \rceil$. Here, W is the width of the input image; $r_{\min}$ and $r_{\max}$ (both in $[0, 1]$) are the minimum and maximum *relative* support widths for pattern estimation, respectively. For example, for $W = 32$, with $r_{\min} = 0.15$ and $r_{\max} = 0.2$, $L = 2$ pattern estimations with $w = 5$ and $w = 6$ respectively will be performed for each class pair.

For each class pair (s, t) and support width w , we estimate a pattern $\mathbf{v}_{stw}^*$ so as to *maximize* the group misclassification fraction from class s to class t by solving the following problem:

$$\underset{\mathbf{v}_{stw}}{\text{maximize}} \quad \sum_{\mathbf{x} \in \mathcal{D}_s} p_t(g_P(\mathbf{x}, \mathbf{v}_{stw}, \mathbf{M}_w)). \quad (4)$$

Here, $\mathcal{D}_s$ is the set of clean images from source class s possessed by the defender that are correctly classified, $p_t(\cdot)$ is the DNN’s posterior probability for class t , and $\mathbf{M}_w \in \{0, 1\}^{W \times W}$ is the mask with the fixed $w \times w$ square spatial support specified by locations with 1’s. Also, $g_P(\cdot)$ is the image resulting from backdoor embedding (Eqn. (1)). Our pattern estimation problem, which could be solved using e.g. gradient descent, is more tractable than NC, which *jointly* estimates a pattern and its support mask. Finally, our detection statistic, the maximum achievable misclassification fraction (MAMF) associated with each class pair (s, t) and spatial support width w , denoted by $\rho_{stw} \in [0, 1]$, is computed as the fraction of all $\tilde{\mathbf{x}} = g_P(\mathbf{x}, \mathbf{v}_{stw}^*, \mathbf{M}_w)$, where $\mathbf{x} \in \mathcal{D}_s$, that are (mis)classified to class t , i.e.

$$\rho_{stw} = \frac{1}{|\mathcal{D}_s|} \sum_{\mathbf{x} \in \mathcal{D}_s} \mathbb{1}(f(g_P(\mathbf{x}, \mathbf{v}_{stw}^*, \mathbf{M}_w)) = t), \quad (5)$$

where $\mathbb{1}(\cdot)$ is the indicator function and $f(\cdot)$ is the classifier mapping from the input to the (maximum *a posteriori*) predicted label. Note that the objective function in (4) is a surrogate for (5) that, unlike (5), is amenable to gradient descent. Other surrogate objectives consistent with maximizing misclassifications could also be used, e.g. an objective function similar to that associated with the Perceptron algorithm [42].

⁸Because of Property 1, it does not matter where we locate a square support. It will not impact pattern estimation and detection performance.

2) *Detection Inference*: For each class pair (s, t) , we compute the *average* MAMF $\bar{\rho}_{st}$ over the L MAMF statistics, each corresponding to one of the L spatial supports used for pattern estimation. For a relatively small $r_{\max}$ (and the L associated small spatial supports), if there was an attack, we would expect at least one true backdoor pair to have a large average MAMF; otherwise, the maximum $\bar{\rho}_{st}$ for all (s, t) pairs is expected to be small. Hence there should be a range of thresholds $\pi \in (0, 1]$ (with very high true detection rate and low false positive rate as confirmed in Section 5.2), such that if

$$\rho^* = \max_{(s,t)} \bar{\rho}_{st} > \pi, \quad (6)$$

the DNN is inferred to *be attacked*; otherwise, no backdoor is detected. We will further discuss the choices of $r_{\min}$ and $r_{\max}$ in Section 5.2. Moreover, if an attack is detected, $(s^*, t^*) = \arg \max_{(s,t)} \bar{\rho}_{st}$ is inferred as one (source, target) class pair involved in the backdoor attack.

5 Experiments

5.1 Devising Backdoor Attacks

We demonstrate the validity of the properties in Section 4.1 and the performance of our detector using nine attacks, involving five datasets, five DNN structures, and nine backdoor patterns. The datasets and the DNN structures considered here are commonly used in computer vision research. Other popular datasets, e.g. Caltech101, ImageNet, etc., are not used in our experiments since the overall test accuracy or the accuracy of particular classes are fairly low for them.

For each attack, we first trained a benchmark DNN using an unpoisoned training set and report the accuracy on clean test images as the “benchmark accuracy”. The dataset being used, image size, number of classes, training size and test size are shown in Table 1. In particular, for Attack G on Oxford-IIIT, the test accuracy using the entire dataset, in absence of backdoor attacks, is very low [43]. We achieve a reasonable benchmark test accuracy on G by using a subset of data involving only 6 classes. For Attack I on PubFig, many images are not able to be downloaded; hence we only consider 33 (out of

Table 1: Details of the attacks.

	Attack A	Attack B	Attack C	Attack D	Attack E	Attack F	Attack G	Attack H	Attack I
Dataset	CIFAR-10	CIFAR-10	CIFAR-10	CIFAR-10	CIFAR-10	CIFAR-100	Oxford-IIIT	SVHN	PubFig
Image size	32×32	32×32	32×32	32×32	32×32	32×32	128×128	32×32	256×256
No. classes	10	10	10	10	10	100	6	10	33
Training size	50000	50000	50000	50000	50000	50000	900	73257	2782
Test size	10000	10000	10000	10000	10000	10000	300	26032	495
DNN structure	ResNet-18	ResNet-18	ResNet-18	ResNet-18	VGG-16	ResNet-34	AlexNet	ConvNet	VGG-16
Learning rate	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-4}	10^{-5}	10^{-3}	10^{-4}
Batch size	32	32	32	32	32	32	16	32	32
No. training epochs	200	200	200	200	200	200	120	80	120
Benchmark acc. (%)	86.7	88.1	86.7	87.6	87.9	71.9	88.7	89.2	76.0
Source class	“cat”	“deer”	“airplane”	“frog”	“truck”	“road”	“chihuahua”	“3”	“B. Obama”
Target class	“dog”	“horse”	“bird”	“bird”	“automobile”	“bed”	“Abyssinian”	“8”	“C. Ronaldo”
Backdoor Pattern	“bug”	“butterfly”	“rainbow”	“bug&butterfly”	“gas tank”	“marmot”	“tennis ball”	“bullet holes”	“sunglasses”
No. backdoor training images	150	150	150	150	150	100	50	500	40
Attack test acc. (%)	87.0	86.9	86.8	87.0	89.1	71.7	90.0	90.1	77.0
Attack succ. rate (%)	99.3	98.0	96.4	98.0	97.9	92.0	84.0	91.4	93.3

60) classes with more than 60 images. The DNN structures involved in our experiments include ResNet-18, ResNet-34 [44], VGG-16 [45], AlexNet [3] and ConvNet [46]. The choices of DNN structure, learning rate, batch size and number of training epochs for each attack instance are also shown in Table 1. The Adam optimizer is used for all DNN training in this paper. For the benchmark training for the datasets involving Attacks F, G and I, we also use training data augmentation including random cropping, random horizontal flipping and random rotation. For the benchmark training for the dataset involving Attack G, we fine-tune a pretrained AlexNet provided by Pytorch. For the benchmark training for the dataset involving Attack I, we adopt transfer learning by retraining the last four layers of a pretrained VGG-face model [47].

Under each attack, we train the DNN using the same training settings as for the benchmark, except that the training set is poisoned by a number of backdoor images. The backdoor images are created using clean images from *one* source class, with a backdoor pattern added following Eq. (1), and then labeling to a target class. In the experiments in this section, we chose to evaluate our detector against backdoor attacks

involving one source class for the convenience of crafting scene-plausible backdoor patterns that could possibly be used in practice. As we have discussed in Section 4.2, the design of our detector allows it to detect backdoor attacks with any number of source classes. In Section 5.7, we will show the effectiveness of our detector against backdoor attacks involving $(K - 1)$ source classes, even though the backdoor patterns may not be scene-plausible (e.g. a rainbow will be placed in an image without the sky). For the experiments in the current section, the choices of the backdoor pattern, the number of backdoor training images, the source class and the target class for each attack instance are shown in Table 1.

To create scene-plausible backdoor attacks, we first created a large number of candidate backdoor images, each with the backdoor pattern randomly located in the image. Then we manually picked the images in which the backdoor pattern looks scene-plausible⁹. For example, for Attack C where the source class is “airplane” and the backdoor pattern is a “rainbow”, a valid backdoor training image being selected should have the rainbow in the sky (see Figure 4c). Since the focus of this paper is on the defense against attacks, we did not polish the backdoor patterns to make them completely innocuous to humans. In Figure 4, we show an example backdoor training image and its original clean image for each attack. The backdoor patterns being considered here are representative of multiple types of practical backdoor patterns. The “bug” for Attack A and the “butterfly” for attack B represent backdoor patterns in the periphery (not covering the foreground object of interest) of an image and with a modest size. The “rainbow” for Attack C represents large backdoor patterns with an irregular shape. The backdoor patterns for Attack D and H represent dispersed patterns. The “sunglasses” for Attack I represent backdoor patterns overlapping with features of interest.

In Table 1, we also report the accuracy on clean test images and the attack success rate for all attacks. The attack success rate is defined as the fraction of backdoor *test* images being classified to the target class prescribed by the attacker. A backdoor test image is created using a clean test image from the source class(es), adding to it the same backdoor pattern used for creating the backdoor training images. We took an automated approach (due to the huge number of test images) to create backdoor test

⁹Only for Attack I we carefully place the sunglasses on the face.

images, randomly placing the backdoor pattern in the image (except for Attack I where we manually place the sunglasses on the faces). By doing so for a sufficiently large test set, we should be covering all possible spatial locations where the attacker could place the backdoor pattern when creating backdoor test images in practice. As shown in the table, for all attacks, the attack success rate is high, and there is no significant degradation in clean test accuracy compared with the benchmark accuracy; hence all attacks are considered successful. *Moreover*, we emphasize that such success is achieved with randomly placed backdoor patterns in test images, which experimentally verifies Property 1, i.e. the spatial invariance of the learned perceptible backdoor mapping.

5.2 Detection Performance Evaluation

In this section, we evaluate the detection performance of the proposed detector, in comparison with the state-of-the-art detector NC [25], and an even earlier defense FP [24], using the above-mentioned attacks.

For each attack, detection is applied to both the DNN being attacked and the clean benchmark DNN. For the best discrimination between the two categories of DNNs, for our proposed method, relatively small spatial supports should be used for pattern estimation, so that a DNN being attacked can have a much larger ρ^* than a clean DNN. On the one hand, the spatial support should be smaller than that of the foreground objects associated with the actual classes; otherwise, the actual target class object(s) could possibly be reverse engineered, causing high group misclassification even for a clean DNN. Moreover, if $r_{\max}$ needs to be chosen large to achieve high ρ^* for a DNN that was backdoor-poisoned, then the backdoor is likely not innocuous, i.e. it is likely an *overly large* perceptible pattern and likely not scene-plausible. Here, $r_{\max} = 0.22$ for all attack instances for our experiments. In practice, an $r_{\max}$ could be adaptively chosen for each dataset as the *critical* spatial support allowing a proportion of class pairs (e.g. 50%) to achieve a modest MAMF (e.g. 0.4) during the pattern estimation step. On the other hand, the spatial support should be large enough for estimating any pattern related to the backdoor. Here, $r_{\min} = 0.08$, such that for a 32×32 image, there is at least 3×3 spatial support for pattern estimation.

The details of our detection settings are as follows. For Attack A–E and Attack H, 200 clean images that are correctly classified *per class* are used for detection. For Attack F, G and I, 50, 30 and 9 correctly classified clean images per class are used for detection, respectively. For all DNNs, class pairs and spatial supports, we solve (4) using Adam optimizer and learning rate 0.5 for 100 epochs¹⁰. We use mini-batch size 32 for Attack A–E and H, 10 for Attack F and G and 3 for Attack I. The spatial support for pattern estimation, used in our detection, is a square covering the top left corner of each image. Due to Property 1 that has been verified in Section 5.1, other locations yield similar results, as will be illustrated in Section 5.6.

In Figure 5, for both the DNN being attacked and the clean benchmark DNN, for each attack, we show the (L) MAMF statistics for the class pair with the largest average MAMF (i.e. the class pair corresponding to ρ^*). For example, for Attack A where the data image size is 32×32 , the (absolute) support widths being considered are $w = 3, 4, 5, 6, 7$. For large image size, instead of performing pattern estimation for each integer support width in the interval $[\lceil r_{\min} \times W \rceil, \lfloor r_{\max} \times W \rfloor]$, we could efficiently downsample and perform pattern estimation for fewer support widths. Here, $L = 9$ and $L = 7$ support widths are considered for Attack G (see Figure 5g) and Attack I (see Figure 5i) respectively. In each sub-figure of Figure 5, a large gap is observed between the clean and attacked curves, clearly distinguishing the DNN being attacked from the clean DNN.

In Figure 6, we show the largest average MAMF (i.e. ρ^*) for both the clean and attacked DNNs, for all attacks. Note the clear large difference between the two bars for each attack. Thus any detection threshold π in $(0.6, 0.8)$, if used, could successfully detect whether the DNN is attacked. Moreover, for the DNN being attacked, for all attack instances, the class pair corresponding to the maximum average MAMF is precisely the *true backdoor pair*. Hence when a detection is made, the (source, target) class pair (used by the attacker) is also correctly inferred, in all cases.

For the attacks in the current experiment (that use a single source class), NC is not expected to make a correct detection, since NC is based on the assumption that *all classes* other than the target class are involved in the attack. Also, NC jointly estimates

¹⁰Fewer epochs are actually needed for decent convergence in all our experiments.

Table 2: Detailed settings of NC, including the number of clean images per class used for detection, the choice of λ , the learning rate and the batch size, for each attack instance.

	Attack A	Attack B	Attack C	Attack D	Attack E	Attack F	Attack G	Attack H	Attack I
No. images per class	100	100	100	100	100	10	30	100	9
λ	0.1	0.1	0.1	0.1	0.6	0.1	0.5	0.2	0.1
Learning rate	0.05	0.05	0.05	0.05	0.05	0.05	0.001	0.01	0.005
Batch size	90	90	90	90	90	90	60	100	30

a pattern and an associated mask for each putative target class to induce at least ϕ -level misclassification. Such optimization relies on the choice of the penalty multiplier λ (for the L_1 -regularization of the mask) and the training settings. Here we evaluate NC using the same attacks. We used the Adam optimizer with the parameters suggested by the authors of NC and performed mini-batch optimization for a sufficient number of epochs (until convergence). If λ is too large, ϕ -level group misclassification cannot be achieved since the mask “size” is over-penalized. If λ is made small, high group misclassification fraction will be maintained, but the L_1 norm of the mask is unreasonably large. Hence we carefully adjust λ and the training parameters for each attack to achieve a mask with small L_1 norm and a pattern inducing a high group misclassification fraction to the true backdoor class. In this way, we in fact are optimistically tuning NC’s hyperparameter λ to maximize NC’s performance. For all the attack cases, we set $\phi = 0.9$, and the optimization is performed for 200 epochs. The number of clean images per class used for pattern estimation in the NC detection procedure, the choice of λ , the learning rate and the batch size for each attack instance are shown in Table 2.

The detection inference of NC uses an “anomaly index” derived based on median absolute deviation (MAD) [41] of the L_1 norm of the mask for all putative target classes [25]. If the anomaly index is larger than 2.0, a detection is made with 95% confidence. In Figure 7, we show the anomaly indices for both the DNN being attacked and the clean DNN for each attack instance. Only for Attack B and Attack G, NC successfully detects the attack – for these two attacks, there is a pattern and an associated *relatively small* mask which, when applied to clean test images from all classes other than the

Table 3: Attack success rate (%) of backdoor test images with (Gaussian) noisy backdoor patterns with $\sigma^2 = 0.01, 0.25, 1$ and cropped backdoor patterns to 64% and 36% of the original size.

	σ^2			Crop	
	0.01	0.25	1	64%	36%
Attack A	85.2	53.6	53.7	84.6	73.2
Attack B	97.8	86.7	87.6	67.8	26.3
Attack C	62.4	46.9	23.6	96.0	29.1
Attack D	98.1	99.9	99.0	81.7	60.2
Attack E	78.4	45.6	32.0	91.4	46.0
Attack F	97.0	91.0	83.0	86.0	69.0
Attack G	78.0	38.0	28.0	78.0	62.0
Attack H	91.3	66.7	67.2	41.7	20.7
Attack I	86.7	86.7	80.0	40.0	26.7

target class, induces a high group misclassification (even though the backdoor targeted only a single source class). This phenomenon (where non-source class images are with high probability misclassified to the target class when the backdoor pattern is added to them) was discovered and identified in [20] as “collateral damage” (ref. Section 2.2.2).

We also show that the FP defense proposed by [24] is ineffective against most of our devised attacks. We prune the penultimate layer neurons of each classifier being attacked (until only few are left), in increasing order of their average activations over all clean test images. In Figure 8, for each attack, we show the accuracy on clean test images and the attack success rate versus the number of neurons being pruned. For most of the attacks (except Attack E and H), the attack success rate does not drop before the accuracy on clean test images is degraded, as the number of neurons being pruned grows. Thus, the FP defense is generally unsuccessful against these backdoor attacks.

5.3 Verification of Property 2

Here we verify Property 2, the robustness property of perceptible backdoor patterns, from two aspects. First, for each attack, we modify the backdoor pattern embedded into *each* clean test image (from the source class) by adding Gaussian noise $N(0, \sigma^2)$ to each pixel (and then clipping each pixel value to $[0, 1]$). Second, instead of adding

noise, we crop part of the backdoor pattern outside its center before embedding to the clean test images. We evaluate the attack success rate for noisy backdoor patterns with $\sigma^2 = 0.1, 0.5, 1$ and cropped backdoor patterns to 64% and 36% of the original size in Table 3. For most of the attack instances, using noisy or cropped backdoor patterns at test time still achieves high attack success rate (even though cropping around the center of a backdoor pattern may remove critical features on the periphery of the backdoor pattern).

5.4 Number of Images for Detection

We note that if too few clean images are available to the defender, the performance of our detector could be affected. When there are no attacks, a false detection may be made since it is much easier to find a pattern that induces a high misclassification rate for a small group, compared with a larger one. This has been essentially verified in [48], as a “universal” TTE perturbation (a common TTE perturbation applied to all images) inducing high group misclassification needs much larger norm (and full spatial support) than a perturbation inducing single-image misclassification at test time. In Figure 9, we show the MAMF statistics for the class pair with the largest average MAMF for the clean and attacked DNNs for Attack A, with 5, 10, 25, 50, 100 and 200 clean images per class used for pattern estimation. When the number of clean images used for detection is greater than 10, there is a clear gap between the two curves for the DNN being attacked and its clean counterpart. But if fewer images are used for detection, the curve corresponding to the clean DNN approaches 1.0 quite quickly, i.e. achieving a high group misclassification fraction becomes easier and requires a smaller spatial support. This is the same phenomenon as in Figure 5i for Attack I, where the curve for the clean DNN quickly achieves 1.0 as the support width for pattern estimation increases, since only 9 clean images are used for detection. Note, though, that our detector is successful even in this case – there is still a large gap between the curves.

In practice, for the purpose of evaluating the clean test accuracy of the DNN (clean test images can be used both in a backdoor detection procedure and for evaluating clean test accuracy), the user should possess sufficient clean images from all classes. If not,

we suggest to use a smaller detection threshold to compensate.

5.5 Backdoor Patterns with Fixed Spatial Location

Here we perform a simple experiment to show that the power of the *attack* will be degraded if the perceptible backdoor pattern is spatially fixed when poisoning the clean training images. We use the same dataset, training settings, (source, target) class pair and backdoor pattern under Attack B. The only difference is that, the backdoor training images are created by embedding the backdoor pattern, the “butterfly”, into the bottom left corner of *every* clean training image to be poisoned.

After training, the accuracy of the DNN on clean test images is 87.2%, which is similar to the accuracy of the clean benchmark DNN. Now we create four groups of images, 1000 each, using the clean test images from the source class prescribed by the attacker, such that the backdoor pattern is embedded:

- 1) into the bottom left corner (i.e. the same location as for the backdoor training images) of all images.
- 2) spatially randomly in each image.
- 3) one row up from the bottom left corner of all images.
- 4) one column right of the bottom left corner of all images.

Here we only focus on the spatial location of the backdoor pattern without considering whether the backdoor patterns are scene-plausible or not. The fraction (%) of images in each group that are (mis)classified to the target class prescribed by the attack are 99.7, 4.1, 47.8, 14.4, respectively. Clearly, only if the backdoor pattern during testing is located at the *same* place as during training, the backdoor image will be reliably (mis)classified to the target class. Hence fixing the spatial location of the backdoor pattern when creating backdoor training images largely degrades the robustness of the attack.

5.6 Location of the Spatial Support for Pattern Estimation

According to Property 1, the spatial invariance of the perceptible backdoor mapping, the location of the spatial support in the pattern estimation step can be arbitrarily chosen.

Previously, we fixed the spatial support location to cover the top left corner for all clean images used for detection. Here, we experimentally verify that such choice is not critical to our detection.

Again, we consider the same DNN being attacked and the same clean benchmark DNN under Attack B. We apply the same detection to both the clean and attacked DNNs except that the spatial support for detection is fixed to cover 1) the top right corner, 2) the bottom left corner, 3) the bottom right corner of all clean images used for detection. In Figure 10, for each location of the spatial support, we show the L MAMF statistics (by varying the support width) for the class pair with the largest average MAMF, for both DNNs. In each figure, we observe a large gap between the two curves, indicating that the DNN being attacked could be easily detected for a large range of thresholds. Thus, our detection approach is indeed robust to the chosen location of the spatial support of the pattern mask.

5.7 Detecting Backdoor Attacks with Multiple Source Classes

Here, we show the performance of our detector against backdoor attacks involving multiple source classes. In particular, we consider the case where all classes other than the target class are the source classes. We devise Attack A', B', and C', using the same settings as Attack A, B, and C, respectively, but with all classes except the target class involved as the source classes and 20 backdoor training images per source class (i.e. 180 backdoor training images in total). Note that the purpose of this experiment is to show that our detector does not rely on knowledge of the number of source classes involved in an attack. We do not require the backdoor patterns used in this attack experiment to be scene-plausible. For example, for Attack C', we simply embed the backdoor pattern, the rainbow, into random locations of images from classes like "frog", "dog", and "cat", while most of images from these classes do not contain the sky. The attack success rate (%) of Attack A', B', and C' are 98.0, 95.3, and 98.7, respectively; the accuracy (%) on clean test images are 87.4, 86.8, 87.0, respectively. We apply our detector with the same setting as in Section 5.2 to these three attacks respectively and show the MAMF statistics for the class pair with the largest average MAMF, for each attack, in Figure

Table 4: Percentage (%) of predicted label changes for clean test images and backdoor test images when applying an average filter and a median filter to the images respectively, for each attack.

	avg. filter		med. filter	
	clean test image	backdoor test image	clean test image	backdoor test image
Attack A	19.1	36.7	17.1	41.3
Attack B	20.3	87.4	17.9	90.3
Attack C	21.0	71.4	17.8	63.2
Attack D	21.4	89.0	17.4	80.1
Attack E	21.3	73.9	19.5	60.5
Attack F	28.6	23.0	23.0	8.0
Attack G	12.0	4.0	11.7	6.0
Attack H	6.4	10.3	10.0	18.5
Attack I	27.7	40.0	27.9	13.4

11. Comparing with the benchmark MAMF statistics shown in Figure 5a, 5b, and 5c, the three attacks (with multiple source classes) could be easily detected by our detector.

5.8 Detecting Perceptible Backdoors In-Flight

Here we show that the TTE detection approach proposed by [40] cannot effectively detect perceptible backdoors in-flight. This approach blurs each test image (using, e.g., an average filter), hoping that if a perceptible backdoor pattern exists, it will be “destroyed” by the blurring filter, while most of the features corresponding to the source class still remain recognizable. Hence a detection could be made if the predicted label of a test image changes after blurring. While this defense is shown to be pretty effective against TTE attacks, where the image-specific perturbation for each test image is very subtle and could be easily destroyed, it cannot effectively defeat perceptible backdoor attacks in-flight, likely because blurring does not fully destroy the backdoor pattern and because the perceptible backdoor pattern has been robustly learned by the classifier during training.

In this experiment, we blur the clean test images (without considering whether they are correctly classified or not) and the test images with backdoors for each attack de-

vised in Section 5.1, using an average filter and a median filter respectively. For Attack A–H, the filter size is 2×2 , while for Attack I where the images have much higher resolution, the filter size is 10×10 . In Table 4, we show the percentage of the clean test images and the percentage of the backdoor test images whose label prediction (by the classifier being attacked) is changed after blurring, when applying the average filter and the median filter respectively, for each attack. These two percentages are the false positive rate (FPR) and the true positive rate (TPR) of the detection approach. Only for Attack B and D, this blurring-based detector achieves a relative high TPR ($> 80\%$), but at the cost of a non-negligible FPR ($> 15\%$). For other attacks, further increasing the filter size may result in a higher TPR, but the FPR will increase correspondingly. Moreover, the filter size is a hyperparameter that cannot be easily determined in the in-flight defense scenario. Clearly, detecting perceptible backdoors in-flight is still an open problem – in the absence of availability of the poisoned training set, post-training defense is currently the best defense solution to backdoor attacks for users of DNN classifiers.

6 Conclusions and Future Work

In this paper, we proposed a detector for perceptible backdoors, post-training and *with-out access* to the (possible poisoned) training set. Our detector, inspired by two properties of perceptible backdoor patterns, is based on the maximum achievable misclassification fraction statistic. With an easily chosen threshold, our detector shows strong detection capability for a range of datasets and backdoor patterns.

Yet there are potential improvements to our detector that could be studied in the future. First, even for a DNN with relatively high test accuracy, there might be classes very similar to each other and with high class confusion; hence there might be a pattern found on a small spatial support that induces high group misclassification fraction for such a class pair. To avoid false detections, one may build knowledge of confusion matrix information into the detector, akin to what was done for imperceptible backdoor detection [20]. Second, the pattern estimation step in our detector is performed for all class pairs. When there are thousands of classes, a more efficient version of our detector

should be of great interest.

References

- [1] A. Graves, A. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *ICASSP*, May 2013.
- [2] M. Spencer, J. Eickholt, and J. Cheng, “A deep learning network approach to ab initio protein secondary structure prediction,” *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 12, no. 1, pp. 103–112, Jan 2015.
- [3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *NIPS*, 2012, pp. 1097–1105.
- [4] Y. LeCun, K. Kavukcuoglu, and C. Farabet, “Convolutional networks and applications in vision,” in *ISCAS*, 2010, pp. 253–256.
- [5] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in *Proc. ICLR*, 2014.
- [6] D. Miller, Z. Xiang, and G. Kesidis, “Adversarial learning in statistical classification: A comprehensive review of defenses against attacks,” *Proceedings of the IEEE*, vol. 108, pp. 402–433, March 2020.
- [7] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in *Proc. ICLR*, 2015.
- [8] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” in *Proc. 1st IEEE European Symp. on Security and Privacy*, 2016.
- [9] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “DeepFool: a simple and accurate method to fool deep neural networks,” in *Proc. CVPR*, 2016.
- [10] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in *2017 IEEE Symposium on Security and Privacy (SP)*, May 2017, pp. 39–57.

- [11] H. Z. J. Y. Pin-Yu Chen, Yash Sharma and C.-J. Hsieh, “Ead: Elastic-net attacks to deep neural networks via adversarial examples,” in *AAAI*, 2018.
- [12] L. Huang, A. Joseph, B. Nelson, B. Rubinstein, and J. Tygar, “Adversarial machine learning,” in *Proc. 4th ACM Workshop on Artificial Intelligence and Security (AISec)*, 2011.
- [13] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, and F. Roli, “Support vector machines under adversarial label contamination,” *Neurocomputing*, vol. 160, no. C, pp. 53–62, July 2015.
- [14] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, “Targeted backdoor attacks on deep learning systems using data poisoning,” <https://arxiv.org/abs/1712.05526v1>, 2017.
- [15] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, and J. Zhai, “Trojaning attack on neural networks,” in *Proc. NDSS*, San Diego, CA, Feb. 2018.
- [16] 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.
- [17] C. Liao, H. Zhong, A. Squicciarini, S. Zhu, and D. Miller, “Backdoor embedding in convolutional neural network models via invisible perturbation,” <http://arxiv.org/abs/1808.10307>, 2018.
- [18] B. Tran, J. Li, and A. Madry, “Spectral signatures in backdoor attacks,” in *Proc. NIPS*, 2018.
- [19] Z. Xiang, D. Miller, and G. Kesidis, “A Benchmark Study of Backdoor Data Poisoning Defenses for Deep Neural Network Classifiers and A Novel Defense Only Legitimate Samples,” in *Proc. IEEE MLSP*, Pittsburgh, Oct. 2019.
- [20] —, “Revealing Backdoors, Post-Training, in DNN Classifiers via Novel Inference on Optimized Perturbations Inducing Group Misclassification,” <https://arxiv.org/abs/1908.10498>, 2019.

- [21] 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," <https://arxiv.org/abs/1908.01763>, 2019.
- [22] B. Nelson and B. B. et al., "Misleading learners: Co-opting your spam filter," in *Machine Learning in Cyber Trust: Security, Privacy, and Reliability*, 2009.
- [23] 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," <http://arxiv.org/abs/1811.03728>, Nov. 8, 2018.
- [24] K. Liu, B. Doan-Gavitt, and S. Garg, "Fine-Pruning: Defending Against Backdoor Attacks on Deep Neural Networks," in *Proc. RAID*, 2018.
- [25] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Zhao, "Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks," in *Proc. IEEE Symposium on Security and Privacy*, 2019.
- [26] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. Celik, and A. Swami, "Practical black box attacks against machine learning," in *Proc. Asia CCS*, 2017.
- [27] P. L. P. Koh, "Understanding black-box predictions via influence functions," in *ICML*, 2017.
- [28] C. Yang, Q. Wu, H., and Y. Chen, "Generative poisoning attack method against neural networks," <http://arxiv.org/abs/1703.01340>, 2017.
- [29] G. P. D. B. E. Chou, F. Tramer, "Sentinet: detecting physical attacks against deep learning systems," <https://arxiv.org/abs/1812.00292>, 2018.
- [30] A. Turner, D. Tsipras, and A. Madry, "Clean-label backdoor attack," <https://people.csail.mit.edu/madry/lab/cleanlabel.pdf>, 2018.
- [31] H. P. A. Saha, A. Subramanya, "Hidden trigger backdoor attacks," in *AAAI*, 2020.
- [32] J. Y. M. X. D. K. H. Z. S. Li, Z. Zhao, "Invisible backdoor attacks against deep neural networks," <https://arxiv.org/abs/1909.02742>, 2019.

- [33] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, “How to backdoor federated learning,” <http://arxiv.org/abs/1807.00459>, 2018.
- [34] C. Xie, K. Huang, P.-Y. Chen, and B. Li, “Dba: Distributed backdoor attacks against federated learning,” in *International Conference on Learning Representations*, 2020.
- [35] V. Smith, C. Chiang, M. Sanjabi, and A. Talwalkar, “Federated multi-task learning,” in *Proc. NIPS*, 2017.
- [36] J. Dai and C. Chen, “A backdoor attack against lstm-based text classification systems,” <https://arxiv.org/abs/1905.12457>, 2019.
- [37] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 618–626.
- [38] R. Feinman, R. Curtin, S. Shintre, and A. Gardner, “Detecting adversarial samples from artifacts,” <https://arxiv.org/abs/1703.00410v2>, 2017.
- [39] D. Miller, Y. Wang, and G. Kesidis, “When not to classify: Anomaly detection of attacks (ada) on dnn classifiers at test time,” *Neural Computation*, vol. 31, no. 8, pp. 1624–1670, Aug 2019.
- [40] X. Li and F. Li, “Adversarial examples detection in deep networks with convolutional filter statistics,” in *Proc. ICCV*, 2017.
- [41] F. R. Hampel, “The influence curve and its role in robust estimation,” *Journal of the American Statistical Association*, vol. 69, 1974.
- [42] R. Duda, P. Hart, and D. Stork, *Pattern classification*. Wiley, 2001.
- [43] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, “Cats and dogs,” in *Proc. CVPR*, June 2012.

- [44] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, 2016.
- [45] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *ICLR*, 2015.
- [46] P. Sermanet, S. Chintala, and Y. LeCun, “Convolutional neural networks applied to house numbers digit classification,” in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, Nov 2012.
- [47] S. Albanie, “Pretrained vgg-face model,” <http://www.robots.ox.ac.uk/~albanie/pytorch-models.html>.
- [48] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in *Proc. CVPR*, 2017.

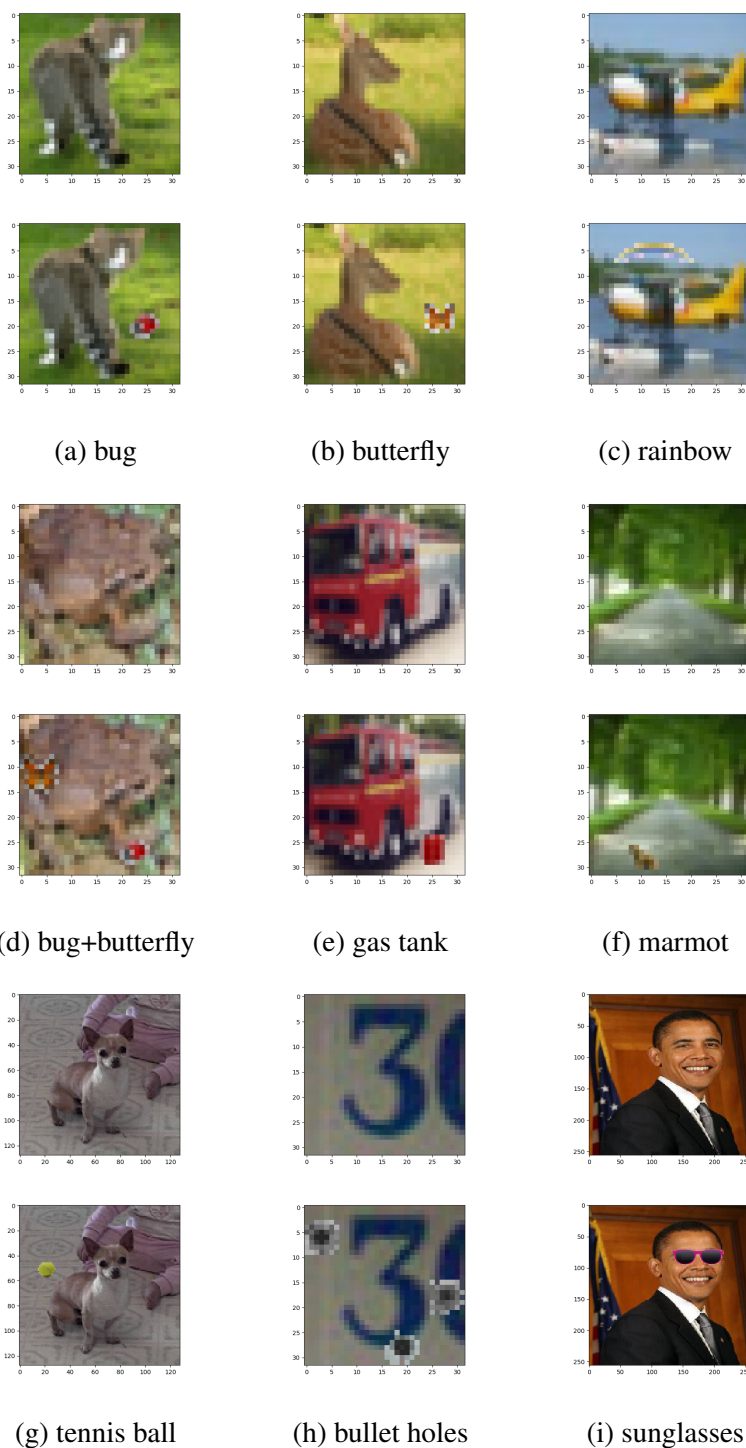


Figure 4: Example backdoor image and the originally clean image for Attack A–I. Subcaptions describe the object(s) added as the perceptible backdoor pattern for each attack.

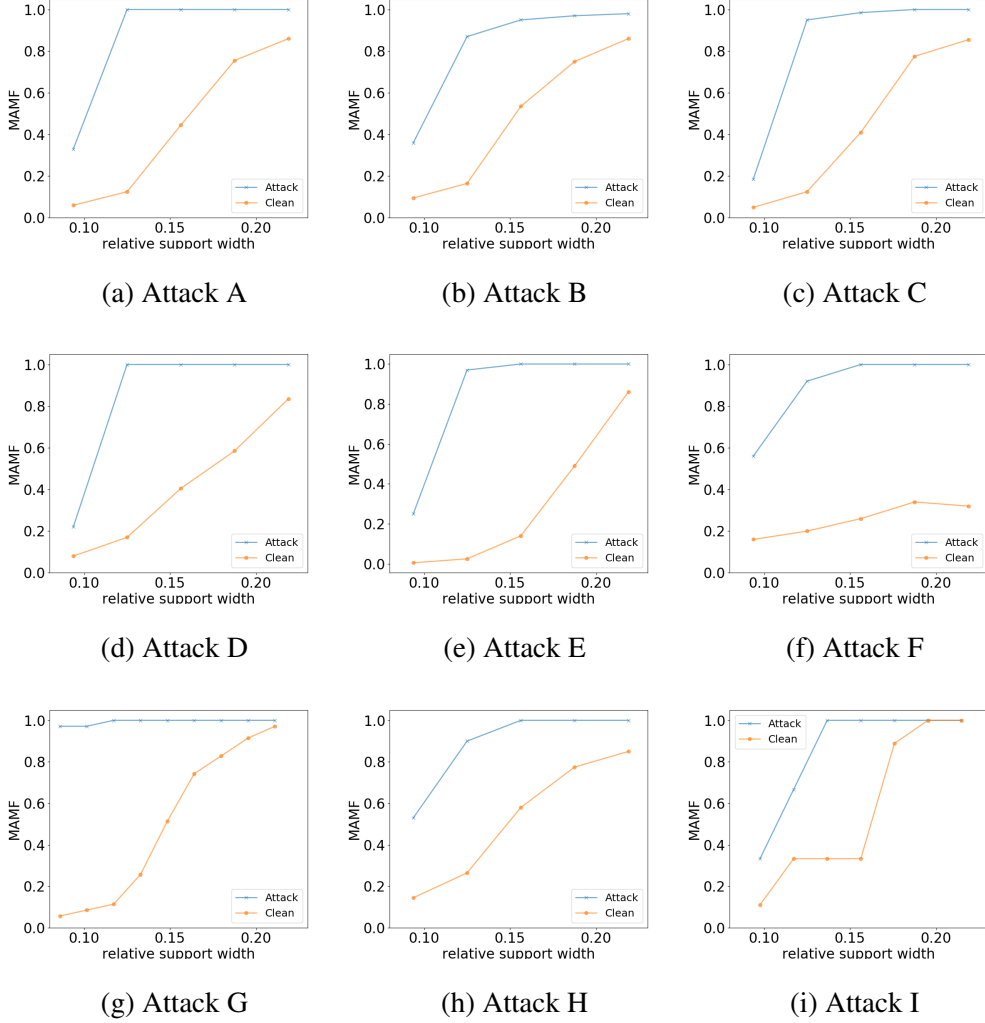


Figure 5: Maximum achievable misclassification fraction (MAMF) statistics for the class pair with the largest average MAMF, for both attacked DNN and unattacked (clean DNN), for Attack A–I. Relative support range for pattern estimation is (0.08, 0.22).

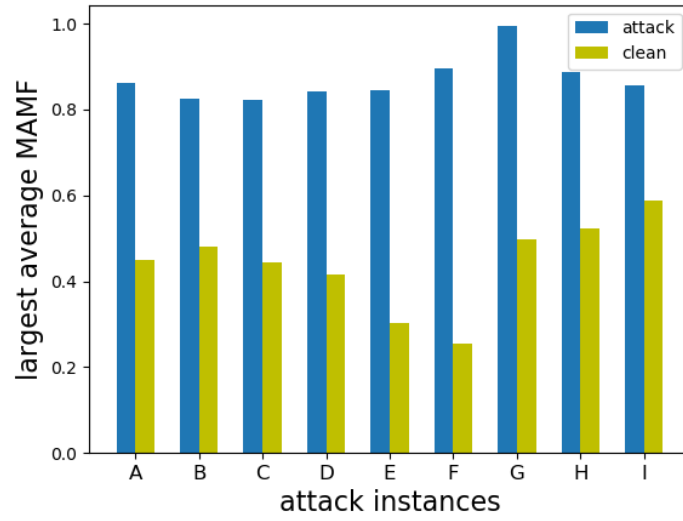


Figure 6: Largest average maximum achievable misclassification fraction (MAMF), i.e. ρ^* , over all class pairs for both the attacked DNN and unattacked (clean) DNN, for Attack A–I.

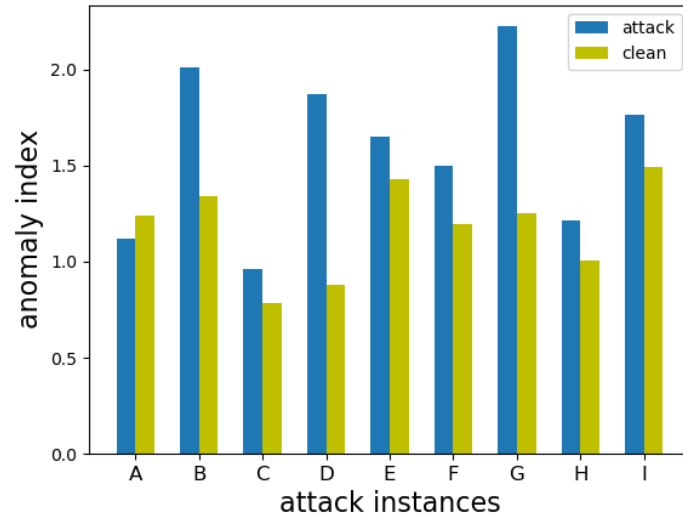


Figure 7: Anomaly indices when applying NC to the DNN being attacked and the clean DNN of Attack A–I.

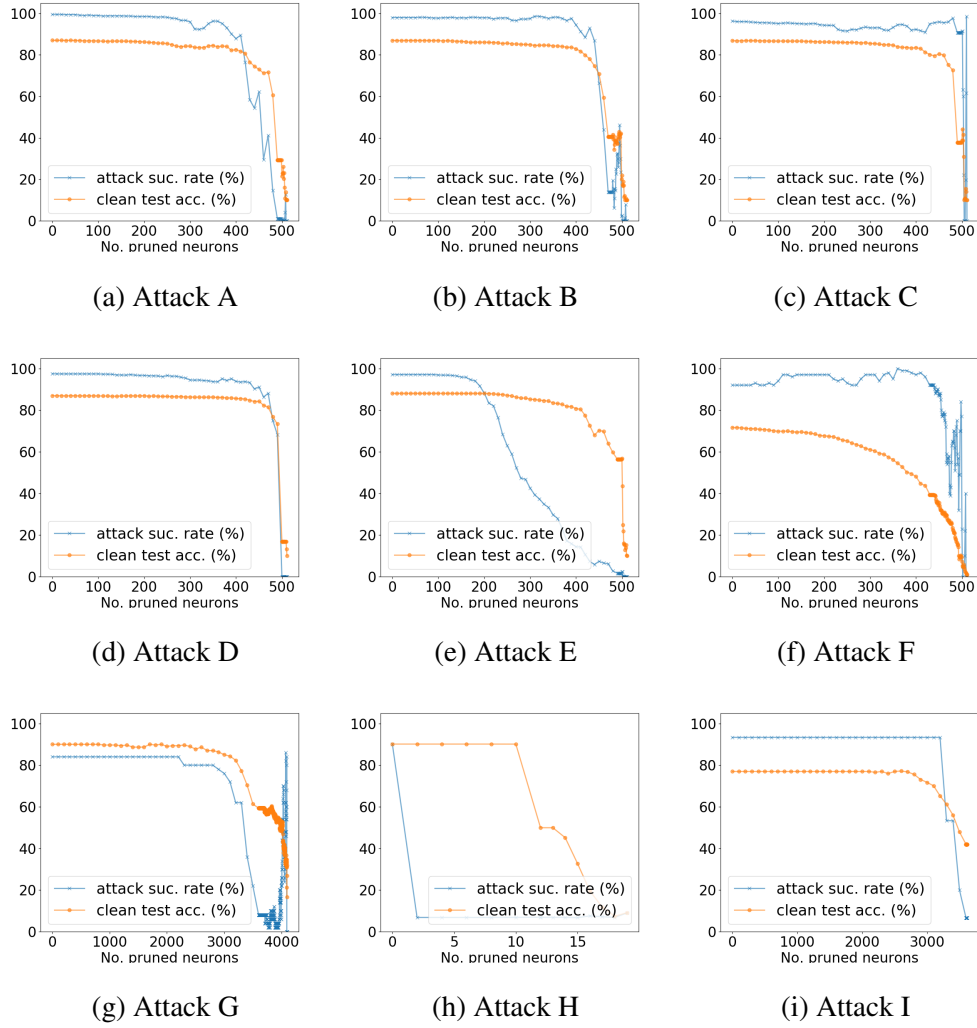


Figure 8: Attack success rate and accuracy on clean test images as the number of penultimate layer neurons being pruned is increased, for each DNN being attacked.

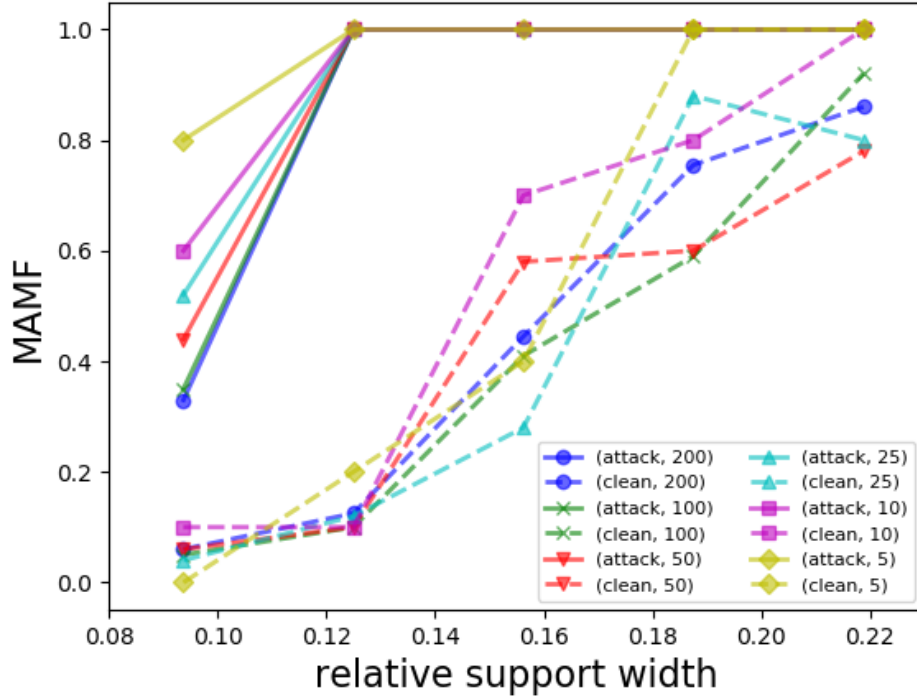


Figure 9: Maximum achievable misclassification fraction (MAMF) ρ^* for a range of choices of relative support width r , for both DNNs being attacked and unattacked for Attack A, when using 5, 10, 25, 50, 100 and 200 clean images per class for detection, respectively.

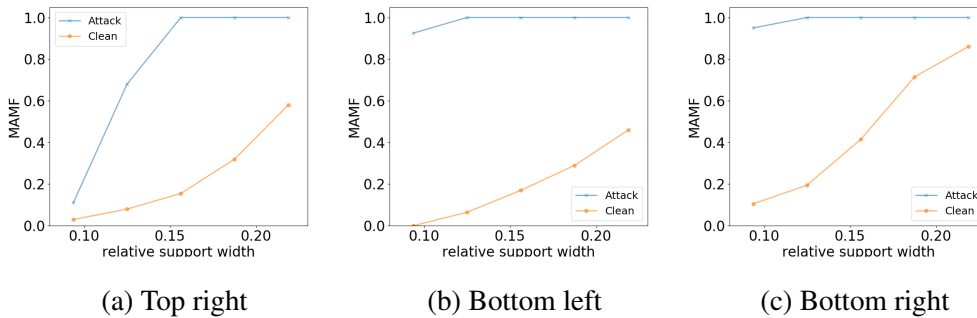


Figure 10: Maximum achievable misclassification fraction (MAMF) statistics for the class pair with the largest average MAMF, for both the DNN being attacked and the DNN not attacked under Attack B. The spatial support for pattern estimation is fixed to cover a) the top left corner, b) the bottom left corner, and c) the bottom right corner.

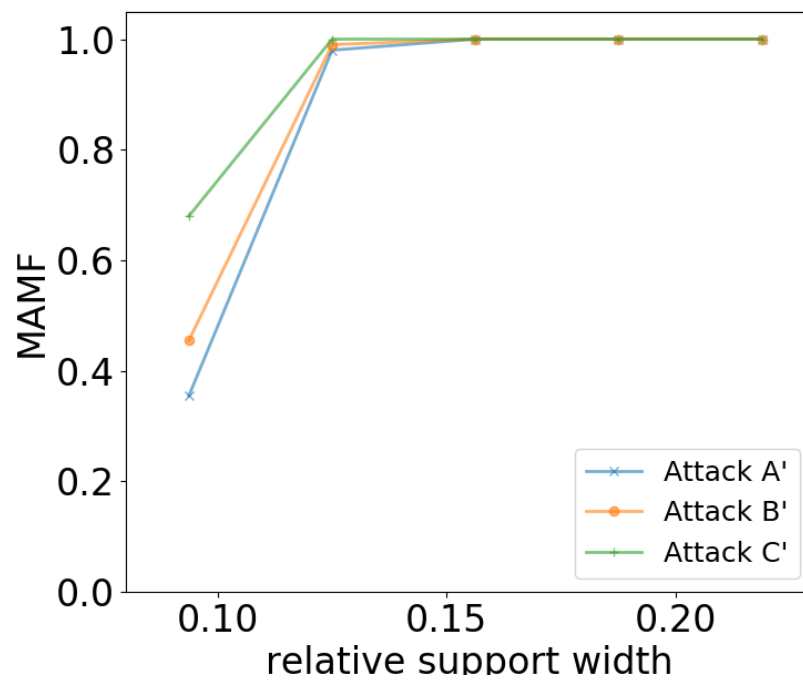


Figure 11: Maximum achievable misclassification fraction (MAMF) ρ^* for a range of choices of relative support width r , for Attack A', B', and C'.