Revealing Backdoors, Post-Training, in DNN Classifiers via Novel Inference on Optimized Perturbations Inducing Group Misclassification

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

Abstract

With the 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 whenever the backdoor pattern is present in a test example. Launching backdoor attacks does not require knowledge of the classifier or its training process — it only needs the ability to poison the training set with (a sufficient number of) exemplars containing a sufficiently strong backdoor pattern (labeled with the target class). Defenses against backdoor DP attacks can be deployed before/during training, post-training, or in-flight, i.e. during classifier operation/test time. Here, we address post-training detection of backdoor attacks 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 a trained classifier may be the basis of e.g. a phone app that will be shared with many users. Detecting backdoors post-training may thus reveal a widespread attack. We propose a purely unsupervised anomaly detection (AD) defense against imperceptible backdoor attacks that: i) detects whether the trained DNN has been backdoor-attacked; ii) infers the source and target classes involved in a detected attack; iii) we even demonstrate it is possible to accurately estimate the backdoor pattern. Our AD approach involves learning (via suitable cost function minimization) the minimum size 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 attacked pairs require 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 alternative defenses, for several backdoor patterns, data sets, and attack settings 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.

The first two authors contributed equally to this work.

The authors are with the School of Electrical Engineering and Computer Science, College of Engineering, The Pennsylvania State University, University Park, PA 16802 USA (e-mail: zux49@psu.edu; djmiller@engr.psu.edu; kesidis@engr.psu.edu).
I. 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 [15]. From a broad view, there are three 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. [21] [19] [4] [16]. Data poisoning (DP) attacks insert malicious samples into the training set, often to degrade the classifier’s accuracy [9] [28]. Reverse engineering (RE) attacks learn the decision rule of a black-box classifier through numerous queries to the classifier [22] [18] [26].

Recently, a new form of DP attack — a backdoor attack — was proposed [2] [5] [13]. 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 set so that the classifier learns to decide to the target class whenever the backdoor pattern is present. The backdoor pattern (for images, for example) could be a subtle watermark, or a less subtle but seemingly innocuous object in a scene (such as a tree or a bird in the sky). 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 cannot be reliably used as a basis for detecting them. Moreover, launching backdoor DP attacks only requires the ability to poison the training set (with a relatively modest number of backdoor-poisoned training samples) — no knowledge of the classifier is required. 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). Online learning can also be backdoor-poisoned by attackers. Since backdoor attacks require less attacker knowledge and capabilities than e.g. TTE attacks, which require knowledge of the classifier [21] [19] [4] [16], they are a serious practical threat to the integrity of deployed machine learning solutions. Like many existing works, we focus here on DNN image classifiers for convenience, although backdoor attacks are also studied in other domains such as speech recognition [13].

While backdoor attacks on DNN classifiers appeared only fairly recently, several defenses have been proposed aiming to detect and mitigate them [23] [1] [12] [27]. 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. For general DP attacks, which do seek to degrade classification accuracy, detection of poisoned samples is not such a difficult problem if there is an online training process with an initially clean (backdoor-free) training set. As more data are collected for training, the defender seeks to detect poisoned examples and reject them from being used. New samples whose inclusion in training induces degradation in accuracy (relative to performance of the initial classifier trained on clean data) on a clean validation set can be flagged as poisoned and removed from the training set. This approach has been used e.g. to detect DP attacks on spam filters [17]. However, this approach is only practically feasible if there is an initially clean training set. This defense is not applicable in the more general embedded poisoning case, where there is no initially clean training set [15]. Regardless, such a defense is anyway also not applicable to backdoor attacks, since these attacks are designed not to degrade the classifier’s accuracy on clean examples. Recent backdoor defenses for the before/during training scenario consider this more challenging (embedded) case where, if there is data poisoning, it is embedded in the initial training set. 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). Recently proposed backdoor defenses (e.g. [23][1][27]) for this embedded case first train a network using the potentially poisoned training set and then investigate the distribution of feature vectors produced at the output of the penultimate layer of the DNN, for each class. If a target class has been attacked, the penultimate layer feature vectors corresponding to the backdoor patterns should be distinct (separable) from the clean samples from this class. Among existing works, a state-of-the-art defense [27] first clusters the feature vectors from each class into a Bayesian Information Criterion (BIC)-selected 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 (that may contain a backdoor). The goal of the defender is to detect potential backdoor patterns and then trigger an alarm [15]. To our knowledge, no existing works have devised defenses against backdoor attacks for this scenario. Such an approach might be based on the detection of unusual neuron activations (in the penultimate layer or even more internal layers), corresponding to the backdoor pattern.

In this paper, we focus 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 a

\footnote{It is additionally useful to determine the source and target classes involved in the detected backdoor.}
clean data set is available (no backdoors present) with examples from each of the classes from the domain. This scenario is of strong interest because DNN training can be extremely computationally intensive; hence learning may be outsourced to a third party who may be compromised. 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. Thus, the defender/user possesses the trained DNN but not the data on which it was trained. We assume the user also possesses clean labeled data on which, e.g., it can evaluate the performance of the DNN. 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 (not so easily detected). As the defender, our fundamental goal is to detect whether the DNN has been backdoor-attacked or not, using only the clean data set. If attacked, the DNN should be discarded and replaced by a new one from a more trustworthy source. Beyond that, we also aim to infer the source class(es) and the target class of the detected backdoor attack. Moreover, in cases where a replacement classifier is not available (due e.g. to limited training resources), it is possible that decisions made to classes not involved in the backdoor attack may still be trusted. Additionally, we aim to estimate the backdoor pattern itself, given that an attack has been detected.

We conjecture (though we do not perform experiments in this paper to confirm) that the backdoor can 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 allowed, the estimated backdoor pattern may still help to detect and thus reject use of the backdoor operationally, i.e. “in-flight”. We also note that there is sufficient interest in post-training backdoor detection that it is the subject of a recent IARPA BAA. 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 to possess in practice and is not needed by our purely unsupervised AD detection approach.

Up until now, few attempts have been made to thwart backdoor attacks post-training. proposed a fine-pruning (FP) approach wherein 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. One limitation of pruning is that the DNN should have sufficient capacity for both the normal classifier operation and the backdoor mapping. Otherwise, for a compact enough DNN, the neurons triggering on backdoor patterns would also trigger on some clean patterns so that any pruning would
necessarily result in loss in classification accuracy. Moreover, FP does not actually detect the presence of backdoor attacks — it is simply expected that pruning will remove backdoor-corrupted neurons, but neurons are pruned even for an unattacked classifier. Finally, the premise behind FP (that there are neurons solely dedicated to performing the backdoor operation) 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.

Our detection approach is based on the observation that a backdoor attack is similar to a TTE attack [21] [19] [4] [16] [14] except that instead of seeking a minimal-size additive perturbation2 to alter a single images decision by the DNN from source class s to target class t, the attacker seeks to alter the decision to class t for every image from class s. Thus, our AD framework is devised by mimicking the attacker, seeking whether we can find a perturbation that is very modest in size/strength 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 (finding such a perturbation effectively amounts to model inversion/reverse-engineering the attack). If so, we detect a backdoor in the DNN involving the class pair (s, t); else the backdoor hypothesis for this pair is rejected.

While developing this work, we became aware of a neural cleanse (NC) defense proposed in [25] that uses similar intuition as ours. They first obtain, for each putative target class, the L1 norm of the minimum-size perturbation inducing misclassification when added to every image from all other classes, by solving an L1-regularized3 objective function minimization problem. Then an “anomaly index” is derived for each class as the L1 norm associated with this class divided by the median absolute deviation (MAD) [7] calculated using the L1 norms from all classes. Detection is achieved by thresholding the anomaly indices — if a class has abnormally large anomaly index, it is detected as the target class of a backdoor attack on the DNN. One main limitation of NC is that their approach assumes that the backdoor has been embedded in training patterns from all classes other than the target class. If the attack actually involves only a single (source, target) class pair, the perturbation their method will require to induce group misclassification of all source classes to the target class is not likely to be small — thus, a single (s, t) backdoor pair (or a small number of pairs) may evade detection by their method, as will be seen by our results. This limitation is crucial in practice because the attacker might avoid choosing too many source classes for the attack.

2Backdoor patterns in the image domain can often be modeled as additive perturbations. However, for the speech domain, backdoor patterns can be additive noises or voice tracks. These are added to the original speech but then may go through a feature extraction process (e.g. cepstral or linear prediction coefficients) before feeding to the classifier. Hence, for the speech domain, assuming the classifier directly takes in the feature vector rather than the raw speech, an additive perturbation attack model may not be suitable.

3The authors wrote L0 norm (of the perturbation ‘mask’) in their problem formulation. But they used the L1 norm in their code posted online.
Otherwise, too many examples with the backdoor pattern will be inserted in the training set, making the attack less evasive. Moreover, their approach generates one decision statistic (anomaly index) per (putative target) class. Thus, assuming a single target class, there are only \((K - 1)\) “null value realizations” (decision statistics guaranteed to \textit{not} be associated with an attack), \(K\) being the number of classes, against which to assess the atypicality of the most extreme statistic as being indicative of a backdoor or not. By contrast, since we consider all class pairs, our method produces \(K(K - 1) - (K - 1) = (K - 1)^2\) such null realizations (the \((K - 1)\) smallest perturbations are considered to be potentially associated with backdoor class pairs and thus are not used in estimating the null distribution). If \(K\) is not too small, we can actually \textit{estimate} a reliable null distribution and evaluate an order statistic p-value for the class pair with the smallest perturbation, based on this null distribution. In addition, the NC defense infers the target class only, while we infer both a source class and the target class, when the attack is detected. Thus, for multiple reasons, our approach yields more reliable and informative detections than NC, as seen in the sequel.

TABOR [6], a recent extension of NC, focuses on detecting localized backdoor patterns. However, they assume that the backdoor pattern is a small icon at the image periphery, and the regularization terms in their objective function penalize for perturbations not satisfying their assumptions about the backdoor pattern — overly large, sparse perturbation masks, or perturbation masks not located in the periphery are penalized. These assumptions about backdoor patterns will not hold in general. We will show in Section [11] that sparse pixel-wise perturbations and global perturbations with small perturbation size are both effective as backdoor patterns. In fact, the backdoor pattern used in the TABOR experiments, which is a Firefox icon, is very easily perceived; hence it is not suitable in practice.

Unlike NC or TABOR, our defense is not limited by the number of source classes involved in the potential backdoor attack. Also, we do not constrain the shape or location of the backdoor pattern. We will show the effectiveness of our defense for different backdoor patterns, data sets, and attack settings, through large-scale experiments. We will also discuss several potential 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. Our contributions in this paper are as follows:

1) We develop one of the first methods (and an approach superior to [12] and [25], as shown in our experiments) for detecting backdoor attacks on DNNs \textit{post-training}, i.e. without the benefit of the poisoned training set, and also without the benefit of additional information such as supervised labeled examples of DNNs with and without backdoors [24], which may be quite unrealistic to possess in practice.

2) Our AD detection approach is related to TTE attack approaches, except in two key respects: i) Instead of inducing misclassification of a \textit{single} pattern, we seek a (small) perturbation that induces misclassification of an entire \textit{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 statistics produced for all $(s, t)$ 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 also make some novel experimental observations about DNN “generalization” induced by backdoors.

II. DETECTION METHODOLOGY

A. 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 a source class $s$ (or one of several such source classes) to a target class $t$ whenever the attacker’s chosen backdoor pattern is added to the original pattern (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 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. The latter requirement may be needed because a human being might inspect a random subset of training set examples and recognize presence of a backdoor pattern in some of them if the attack is not human-imperceptible. 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 class $s$ (or several classes not including class $t$), labeling them as coming from class $t$, and inserting them into the training set. At test time, the attacker can then add the backdoor pattern to an image from class $s$ and, with high probability, elicit a classifier decision to class $t$. Although such attacks do not truly require knowledge about the classifier or the training process itself, 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/energy 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) Again, the backdoor pattern is assumed to be a human-imperceptible additive perturbation applied to an original 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); however, the scene-plausibility of such backdoors is highly dependent on the image domain. Such backdoor attacks may also require much more crafting effort. The defense developed in this paper is not intended to detect backdoors based on innocuous, perceptible patterns – such backdoors are not guaranteed to have small perturbations. Automated detection of perceptible backdoors is expected to be a challenging problem, and a good subject for a future research study.

3) The attacker has no specific knowledge about any defenses that may be deployed to detect the attack. That is, the attack is “black box” with respect to defenses/detectors. However, the attacker does know that the backdoor pattern should be human-imperceptible as human inspection of the training set is always a possibility.

4) The attack involves a single target class and any possible number of source classes (ranging from 1 up to $(K - 1)$). Although more source classes may make the attack less evasive in practice since the number of poisoned samples will grow with the number of involved source classes, we investigate the multiple source class case to demonstrate the detection capability of our AD framework under general attack settings. However, we assume a single target class and a single backdoor pattern for each attack for simplicity. Otherwise, the number of detection hypotheses for the defender to investigate may become too cumbersome. In such cases, the cost to launch the attack is also higher because multiple backdoor patterns need to be carefully designed to simultaneously achieve (multiple) successful attacks, low attack perceptibility, and small impact on clean test set accuracy.

B. Defender’s Goals, Knowledge and Assumptions

The defender 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

4Perturbations with small size/energy are not only imperceptible to humans. In [15] and [27], 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 [1][23].

5Large 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.
possibly be used to mitigate the attack as discussed earlier. The capabilities possessed by the defender are as follows:

1) The defender possesses a labeled, clean data set with images from all classes, 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 interesting and challenging problem, as any information about a possible backdoor is only \textit{latently} available in the learned DNN weights.

2) The defender has full access to the classifier, including its structure and learned parameters, 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 may be too small to adequately retrain the classifier and/or the user may have inadequate computational resources for such retraining (with these in fact the reasons why classifier training may have been outsourced).

3) The defender assumes very little about the backdoor attack — simply 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.

4) There are no labeled example neural networks (labeled as \{“with backdoor”, “without backdoor”\} for the given domain \[24\]. That is, the detection problem we address is purely \textit{unsupervised}. We demonstrate in this paper that strong detection accuracy in this setting is in fact attainable.

C. Notation and Setup

We denote the DNN classifier to be examined for backdoor attacks as \(f(\cdot) : \mathcal{X} \to \mathcal{C}\), where \(\mathcal{X}\) is the input (image) space and \(\mathcal{C} = \{\omega_1, \ldots, \omega_K\}\) is the set of class labels. Conventionally, DNN classifiers follow a “winner take all” rule. That is, for any image \(x \in \mathcal{X}\), a class “score” \(p_\omega(x)\) is obtained for \(\forall \omega \in \mathcal{C}\), with

\[
f(x) = \arg \max_{\omega \in \mathcal{C}} p_\omega(x).
\]

When a softmax activation function is used in the output layer by the DNN classifier, the score of a class is interpretable as its \textit{a posteriori} probability:

\[
p_\omega(x) = \text{prob}[f(x) = \omega | x], \quad \forall \omega \in \mathcal{C}.
\]

We also denote the clean labeled set used for detection as \(\mathcal{D}\), which has a partition \(\mathcal{D} = \bigcup_{\omega \in \mathcal{C}} \mathcal{D}_\omega\), where \(\mathcal{D}_\omega\) contains all images labeled by \(\omega\).

\(^6\)If \(\mathcal{D}\) is unlabeled, we can instead define \(\mathcal{D}_\omega\) as the set of images in \(\mathcal{D}\) classified by the DNN to \(\omega\). Assuming the classifier’s error rate for class \(\omega\) is low, this set will be a good surrogate for the images truly from class \(\omega\).
For the backdoor attack, the designated source class and target class are denoted \( s^* \in C \) and \( t^* \in C \) \((s^* \neq t^*)\), respectively, if the attack involves a single source class. If multiple source classes are involved, the set of such source classes is denoted \( S^* \subset C \) \((t^* \notin S^*)\). The backdoor pattern used to additively poison training samples is denoted \( \nu^* \), with same dimensionality as the input image, \( i.e. \) a poisoned pattern is \( x + \nu^* \). Each element of \( \nu^* \) can be either positive or negative and the absolute value is bounded by the maximum pixel value intensity. Also, since the backdoor pattern is added to a set of images, with each pixel intensity bounded to a prescribed range (\( e.g. [0, 255] \) for a gray scale intensity), we clip a perturbed pixel’s value to the maximum of this range if the perturbed value exceeds the maximum and to the minimum value (\( e.g., 0 \)) if the perturbed value is less than the minimum value. This is an alternative to imposing strict box constraints on the backdoor pattern \[21\]. 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 \( \nu^*_{ij} \), the perturbation for pixel \((i, j)\), to be any real number in the valid range and likewise the clipped version of \( x_{ij} + \nu^*_{ij} \) can be any real number in the valid range.

**D. Proposed Detector: Reverse-Engineering the Attack**

1) **Key Idea:** Many works on TTE attacks \[21\][19][4][16] have shown that the trained classifier’s decision for a single image from class \( s \) can be altered to class \( t \) by adding a small, image-customized perturbation. However, altering the class decision for every image (or most images) from class \( s \) to class \( t \) using a common additive perturbation for all the images is expected in general to require a large perturbation size/energy, \( ||\nu^*||_p \). 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 \( \nu^* \) 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 \((s,t) \in C \times C\), define the optimal perturbation \( \nu^*_{st} \) that induces at least \( \pi \)-level group misclassification as the solution to:

\[
\begin{align*}
\text{minimize} & \quad d(\nu) \\
\text{subject to} & \quad \frac{1}{|D_s|} \sum_{x \in D_s} 1(f([x + \nu]_{\mathbb{C}}) = t) \geq \pi,
\end{align*}
\]

(1)

where \( d(\cdot) \) is the metric for measuring the size (\( L_1 \) norm) or the energy (\( L_2 \) norm) of a perturbation \( \nu \), \([\cdot]_{\mathbb{C}} \) represents the clipping operation, and \( 1(\cdot) \) is the indicator function. \( \pi \in (0, 1] \) can be considered the
minimum misclassification rate to deem a backdoor attack “successful” – e.g. we might set $\pi = 0.8$. The choice of $\pi$ 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 $\pi$ values, as will be shown in Section III-C. In practice $\pi$ can be set by the user.

Our detection procedure consists of two steps. First, we estimate $v^*_{st}$ for each class pair $(s, t) \in C \times C$, $s \neq t$. Second, an inference procedure is performed based on the collection of statistics $\{d(v^*_{st})\}$ 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 $d(v^*_{s^*t^*}) \ll d(v^*_{st})$ 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 the sequel.

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 [10], classifier margin, as well as other training objectives [3] 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 the ideal problem (1). While in the sequel we discuss various alternatives, in our experiments we have optimized the perturbation (i.e. estimated putative backdoors) by performing gradient descent on the objective function

$$J_{st}(v) = -\frac{1}{|D_s|} \sum_{x \in D_s} p_t([x + v_c], (2)$$

until the constraint in problem (1) is satisfied, where $p_t$ is defined in (II-C). The algorithm is summarized below.

**Algorithm 1** Perturbation Optimization

1: Initialization: $v \leftarrow 0$, $\rho \leftarrow \frac{1}{|D_s|} \sum_{x \in D_s} 1(f(x) = t)$
2: while $\rho < \pi$ do
3: $v \leftarrow v + \delta \cdot \nabla J_{st}(v)$
4: $\rho \leftarrow \frac{1}{|D_s|} \sum_{x \in D_s} 1(f([x + v_c], (t)$

There are several things to note about Algorithm 1. First, $\rho$ is updated in each iteration as the misclassification fraction induced by the current perturbation $v$. 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 $\pi$-level misclassification could be achieved by minimizing $J_{st}(\cdot)$ subject to an explicit constraint on the perturbation size\(^7\), this would also lead to a Lagrangian 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 $\pi$-level misclassification is achieved yields small perturbations for backdoor pairs relative to those required for non-backdoor pairs. This is all that is needed 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 alternative surrogate functions are identified in the sequel, and can instead be used, within our detection framework. Fourth, $\delta$ is the step size for updating $v$. If $\delta$ is too small, the execution time can be very long. If $\delta$ is too large, the algorithm may terminate in a few steps with a resulting $\rho$ much larger than $\pi$ and with the resulting perturbation much larger in size/energy than that required to induce $\pi$-level group misclassification. In practice, a suitable $\delta$ can be chosen via line search. Fifth, in our actual implementation, we set an upper bound on the size/energy of $v$, with the algorithm terminated when the upper bound is reached even if the $\pi$ target has not been reached – when $\pi$ 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) Detection Inference: The statistics for the detection inference are $\{r_{st}\}$, where $r_{st} = d(v_{st}^*)^{-1}$, 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 $\forall s \in S^*$), if they exist, will cluster 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 the reciprocal statistics 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 distribution. Hence our detection inference for each classifier (whether it is attacked or not) 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.

\(^7\)Or by minimizing the perturbation size subject to a level of $J_{st}(\cdot)$.\)
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 \( K(K - 1) - (K - 1)^2 \) 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 III-C. However, one should not simply naively learn a null density using the \( (K - 1)^2 \) smallest reciprocal statistics. Note in particular 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 \( r_{\text{min}} < r < r_{\text{max}} \), where \( r_{\text{min}} \) is the smallest of the \( (K - 1) \) largest reciprocals and \( r_{\text{max}} \) 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 \( r_{\text{min}} \). 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 \( g_R(r | r < r_{\text{min}}) \) by maximum likelihood estimation (MLE). Once we estimate this conditional density using these \( (K - 1)^2 \) observations, its parameters then uniquely determine the corresponding unconditional null density \( g_R(r) \). That is, \( g_R(r) = g_R(r | r < r_{\text{min}}) \cdot \text{prob}[R < r_{\text{min}}], \ r \geq 0 \). To our knowledge, this is a novel robust density modeling approach we are proposing. In the sequel, we will experimentally 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, $r_{\text{max}}$, i.e.

$$p_{\text{max}} := \text{prob}_{\text{null}}[\max\{R_1, \ldots, R_{K(K-1)}\} \geq r_{\text{max}}]$$

$$= 1 - \text{prob}_{\text{null}}[\max\{R_1, \ldots, R_{K(K-1)}\} \leq r_{\text{max}}]$$

$$= 1 - G_R(r_{\text{max}})^{K(K-1)}$$

where $G_R(\cdot)$ is the null cumulative distribution function. If this order statistic p-value is less than a threshold $\theta$, the null hypothesis is rejected and the classifier is claimed to be attacked. Since p-values under the null hypothesis are uniformly distributed on $[0, 1]$, $\theta$ 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 $r_{\text{max}}$ 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 $\psi^*$ used by the attacker. This (novel to our knowledge) robust null learning and inference strategy is illustrated pictorially in Figure 1.

![Flow chart for our detection inference procedure.](image)

**E. 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 [3]:

$$J_{st-p}(\psi) = -\frac{1}{|\hat{D}_s(\psi, t)|} \sum_{\hat{x} \in \hat{D}_s(\psi, t)} p_t(\hat{x} + \psi)e,$$  \hspace{1cm} (4)

where $\hat{D}_s(\psi, t) = \{\hat{x} \in D_s : f(\hat{x} + \psi)e \neq t\}$.

Another potential concern with $J_{st}(\psi)$ in Eq. [2] is the use of all images from source class $s$ for perturbation optimization – if the attacker knows the training set and training approach, he 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:

\[ J_{st-c}(v) = -\frac{1}{|\mathcal{D}_s|} \sum_{x \in \mathcal{D}_s, f(x) = s} p_t([x + v]_c), \]  

(5)

where \( \mathcal{D}_s = \{ x \in \mathcal{D}_s : f(x) = s \} \). One can also combine the restrictions in (4) and (5), summing only over \( x \) such that both \( f([x + v]_c) \neq t \) and \( f(x) = s \).

Likewise, one might also hypothesize that the attacker used box constraints [21], rather than clipping, in implementing the backdoor to keep the image intensity values in the proper range. If so, the defender might modify the above surrogate problems to make them consistent with box constraints, rather than clipping.

As noted before, one could also consider an optimization problem that explicitly accounts for/constrains the perturbation size (or minimizes the perturbation size given a constraint on the surrogate misclassification objective). Moreover, one can use a soft error count/discriminative learning objective function akin to that used in [10].

Based on this discussion, one can see that many surrogative objective function variants are possible. In Section III, we do include experimentation 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 \( r_{st^*} \) 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.

F. Alternative Detection Inference

The premise behind our detection inference, as mentioned before, is that the perturbation size/energy required to induce \( \pi \)-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 \( \pi \)-level group misclassification will be smaller than when this initial misclassification fraction is low, i.e. \( d(\mathcal{V}_{st}^*) \) may be abnormally small, possibly resulting in \((s, t)\) being falsely detected.

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

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

where the superscript \(\tau\) 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 \(d(L_{st}^*)\). Intuitively, this correction will increase an “abnormally small” \(d(L_{st}^*)\), making it less likely to be falsely detected. Note that \(d(L_{st}^{(0)}) = 0\) since \(L_{st}^{(0)} = 0\). \(T\) could be chosen as the index of the last iteration such that either \(\rho_{st}^{(T)} \geq \pi\) or such that \(d(L_{st}^{(T)})\) exceeds the upper bound of the perturbation size/norm. Alternatively, we can base the polynomial fit on a smaller number 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 2](image.png)

Fig. 2: An example of correction of perturbation sizes to account for non-zero initial class pair confusion. The orange dots are the (perturbation size/norm, misclassification fraction) sample points for a regime where the perturbation size/norm is small. The blue crosses are the corrected/shifted sample points obtained by adding the correction to the perturbation size/norm. The blue polynomial fits these corrected points.

Figure 2 gives an example of correction to \(d(L_{st}^*)\) 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 (6). 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

\(^8\)In Algorithm 1 we neglect the subscripts for simplicity.
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 \( r_{st}^{c} = [d(v_{st}^{*}) + d_{st}^{(0)}]^{-1} \).

III. EXPERIMENTS

A. Backdoor Patterns

In this paper, we 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 \([2][13][1][5][23][25][12][6]\). 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 \([2][11]\). 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.

![Fig. 3: 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).](image)

Examples of the two types of backdoor patterns for colored images are shown in Figure 3. In the left figure of Figure 3, we show the backdoor pattern from one attack realization used in the experiments in the sequel. Four pixels were randomly selected to be perturbed (in one of the three channels), with
the L2 norm\(^9\) of the perturbation mask equal to 0.6. For visualization purpose, a 0.5 offset was added to each pixel of the image. In the right figure of Figure 3, 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.

### B. Ensemble Experiments

In this experiment, we evaluate the performance of the proposed defense and some of its variants (i.e. surrogate objective functions and confusion-corrected detection inference described in Section II) based on multiple realizations of classifiers under different attack settings and the two types of backdoor patterns mentioned above. The experiment uses the CIFAR-10 data set with 60000 color images \((32 \times 32 \times 3)\) evenly distributed between ten classes. The data set is separated into a training set with 50000 images (5000 per class) and a test set with 10000 images (1000 per class). The DNN classifier uses the ResNet-20 \[^8\] structure with cross-entropy training loss. The training is performed for 200 epochs with mini-batch size of 32, using the Adam optimizer, and with the training data augmentation option. This training achieves an accuracy of 91% on the clean test set when there are no backdoor attacks.

1) **Devising Backdoor Attacks:** Consistent with our description in Section II-A, we evaluated perturbation masks (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 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 selected\(^{10}\) clean training images from the ‘automobile’ (source) class, adding the backdoor pattern to each image, and then clipping as described in Section II-D. These images (with the backdoor pattern) were labeled to the ‘truck’ (target) class and added to the training set of 50000 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 1000 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 10000 clean test patterns is evaluated.

\(^9\)There 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.

\(^{10}\)The 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.
Fig. 4: 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).

As depicted in Figure 4, 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 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 4, 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 process\textsuperscript{11} for evaluating detection performance, we conducted ensemble 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 ($||u^*||_2 = 0.6$). The backdoor images were crafted using clean images from a single source class.
- **BD-G-S**: Classifiers were trained on the training set poisoned by 1000 backdoor images with a global perturbation ($||u^*||_2 = 0.2$). The backdoor images were crafted using clean images from a

\textsuperscript{11}In 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.
single source class.

- **BD-G-M**: Classifiers were trained on the training set poisoned by 900 backdoor images with a global perturbation ($\|u^*\|_2 = 0.2$). The backdoor images were crafted using clean images from nine source classes, i.e., 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 classifier under attack, 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 rate and the accuracy on clean test images (across the 25 realizations) for the four groups of classifiers are reported in Table I — 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 5b), global backdoor pattern (Figure 5c) and the original image (Figure 5a) are shown in Figure 5. 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 $2.6 \times 10^{-5}$.

| 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 |

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 $\tilde{s} \in C \setminus \{s^*, t^*\}$ will be classified to $t^*$ with high probability when the same backdoor pattern is 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

12The 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.
Fig. 5: Examples of backdoor patterns applied to CIFAR-10 images: (a) the original automobile image; (b) automobile with sparse, pixel-wise perturbation ($||v^*||_2 = 0.6$); (c) automobile with global perturbation ($||v^*||_2 = 0.2$).

test, we added the backdoor pattern used for attack to the 1000 clean test images from $\overline{s}$, and obtained a “class collateral damage rate” as the fraction of images classified to $t^*$. In Figure 6a and Figure 6b, 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 ($\overline{s}$) 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 $v^*_{s^*t^*}$ is close to the size/energy of $v^*_{s^*t^*}$. This thus gives further insight into why we exclude the $K - 1$ largest statistics in the set $\{|v^*_{s^*t^*}||\}$ from use in estimating the null distribution. This also helps explain why we only infer a single (source, target) class pair based on the most extreme statistic (as illustrated in Figure 1) – “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 “specific”, i.e. if it does not induce
collateral damage to classes not explicitly attacked.

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**: The basic objective function (Eq. (2)) combined with the principal detection inference approach.
- **AD-Jp-P**: The variant of the basic objective function associated with the Perceptron algorithm (Eq. (4)) combined with the principal detection inference approach.
• **AD-Jc-P**: The variant of the basic objective function considering only the clean source samples that are correctly classified (Eq. (5)) combined with the principal detection inference approach.

• **AD-J-C**: The basic objective function (Eq. (2)) combined with the confusion-corrected detection inference approach described in Section II-F.

• **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 \( \pi \), the target misclassification fraction, to be 0.8. However, we have observed experimentally that detection performance is not sensitive to the choice of \( \pi \) (shown in Section III-C). 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 \( \theta \) on the p-values. As mentioned in Section II, \( \theta \) can be set to fix the theoretical false detection rate. Here we use the “classical” statistical significance threshold \( \theta = 0.05 \) as the default. We will also evaluate performance for more conservative and liberal choices of \( \theta \) in the sequel.

The hyperparameters required to be specified for the NC approach are the detection threshold \( \theta_{\text{MAD}} \), and \( \lambda \), the weight on the regularization term. NC detection inference is performed by comparing the anomaly index of each class, derived from the optimized perturbation that induces \( \pi \)-level group misclassification, with the detection threshold. In our experiments, we use the same \( \theta_{\text{MAD}} = 2 \) as in [25], such that any anomaly index greater than \( \theta_{\text{MAD}} \) indicates a detection with \( > 95\% \) confidence, which matches with our choice of \( \theta \) for our detection approach in terms of the significance level. As for the choice of the weight parameter \( \lambda \), we tested both \( \lambda = 1.5 \) and \( \lambda = 1.0 \).[13] We conjecture (without testing) that if \( \lambda \) is selected too large, perturbation optimization for NC may never reach the \( \pi \)-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 \( \pi \)-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 — \( \pi \)-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

---

[13] In the code posted by the authors online, \( \lambda = 1.5 \) is the default.
density. The variants of NC detection that we evaluated are:

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

| TABLE II: 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 II, the detection accuracy of each variant of the proposed AD and NC detection methods for each group of classifiers is shown. The 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 the NC approach, 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 $\theta_{\text{MAD}}$), 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, needing to detect both the source and target classes involved in an attack. For the group of clean networks, the detection accuracy is defined as the fraction of networks inferred to not be attacked.
All the proposed AD variants achieved perfect detection for the BD-G-M and Clean group experiments. The 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 II 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 II by the accuracy for BD-P-S and BD-G-S. Also, the detection power of NC cannot be improved by choosing a smaller \(\theta_{\text{MAD}}\), since the false detection rate on clean classifiers is then made quite non-negligible. In Figure 7, 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 \(\theta_{\text{MAD}}\).

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 \(\theta\) which yield the same (or very similar) performance. In Table III 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 IV only the true detection rate for the
BD-P-S group is reduced slightly. Moreover, we find from Table II and Table III that the false detection rate for the Clean group is much lower than its expectation, which equals the detection threshold $\theta$. 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 III:** 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 IV:** 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 8 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 3. 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 8 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.

Fig. 8: Two examples of estimated backdoor patterns with ground truth a 4-pixel perturbation mask (left) and a global perturbation mask (right), respectively. The estimated global perturbation mask has been scaled by 30 times to be human visualizable.

Finally, we evaluate the FP approach using the classifiers from BD-P-S, BD-G-S and BD-G-M. Note that FP does not provide any detection inference about whether a classifier has been backdoor attacked or not. Thus, for each classifier, we pruned the neurons in the penultimate layer in increasing order of their average activation over the clean test set, and recorded the attack success rate when the accuracy on the same test set first drops by 2% and 5%, respectively. The histograms of the attack success rate for the classifiers from the three groups at 2% and 5% clean test set accuracy degradation are shown in Figure 9. From the results, only a small portion of the classifiers are successfully pruned such that the backdoor effect is mitigated. This is likely due to the fact that, for most classifiers, penultimate layer (and, in fact all) neurons are functionally shared by clean patterns and backdoor patterns. Thus, successfully removing the backdoor by pruning will entail inevitable (above 5%) degradation in the usability of the classifier.

C. Other Design Choices

1) Choice of $\pi$: Unlike NC, the proposed detection approach is not very sensitive to the choice of $\pi$. 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 $\pi$ from 0.4 to 0.9. We use the default detection threshold $\theta = 0.05$. As shown in Figure 10, the detection accuracy does increase, but not dramatically with $\pi$. The minimum detection accuracy across the range of choices of $\pi$ is 0.8, which can be further improved by use of a more liberal detection threshold. This phenomenon can be understood from Figure 11, in which we plot the sequences of $(\|U_{st}^{(r)}\|_2, \rho_{st}^{(r)})$ 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,
Fig. 9: The histograms of attack success rate for classifiers from (a&b) BD-P-S, (c&d) BD-G-S, and (e&f) BD-G-M, at 2% and 5% absolute classification accuracy degradation, respectively.

and is clearly separated from the sequences for the non-backdoor pairs. In other words, there is a huge range for the choice of $\pi$ (not exceeding 0.9) to achieve correct detection for this example. For any choice of $\pi$ in such range, the $(\|v_\tau^{(s,t)}\|_2, \rho^{(s,t)}_\tau)$ sequence for each $(s,t)$ pair is truncated at the minimum $\tau$ such that $\rho^{(s,t)}_\tau$ first exceeds $\pi$. The resulting $\|v_\tau^{(s,t)}\|_2$ 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.

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 12, 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.
Fig. 10: Detection accuracy of AD-J-P on the BD-P-S group, with a range of choices of $\pi$.

Fig. 11: Sequences of $(||L_{st}^{(\tau)}||_2, \rho_{st}^{(\tau)})$ 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.

per class. Comparing with the size of the training set (5000 images per class) and the number of backdoor images required to launch the attack (1000 images in total\textsuperscript{14}), the detection cost is relatively “cheap” in terms of the required number of clean images. Also, even if there are abundant clean images available to the defender, he can always use only a subset of images to reduce the defender’s computational effort.

3) Fitting Null Distribution Using All Statistics: As described in Section II-D3, 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 V, we

\textsuperscript{14}For backdoor patterns with innocuous objects, poisoning the training set with a few to a few tens of backdoor images yields a successful attack\textsuperscript{2}. For backdoor patterns with human-imperceptible perturbations, hundreds of backdoor images should be used for poisoning the training set to make the attack successful\textsuperscript{23}.
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.

**TABLE V**: 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 accuracy | 0.48   | 0.44   | 0.52   | 1.00  |

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 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 $0.101 \pm 0.017$, while the L2 norm of the optimized perturbation for all non-backdoor class pairs across the 25 realizations is $1.243 \pm 0.321$. That is, the “estimated” backdoor pattern will have the minimum norm necessary to achieve the $\pi$ 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.
D. Other Data Sets

In this part, we evaluate the performance of our detector on several other data sets. For each data set, we train one classifier with no attacks and one attacked classifier using the “chess board” backdoor pattern with $||\mathbf{v}_a|| = 0.2$ (i.e. the same backdoor pattern used for the BD-G-S group). For each classifier, the AD-J-P variant and the AD-J-C variant (with correction using the confusion matrix information) are applied for detection using a clean data set (not used for training) containing 100 images per class.

MNIST contains 70000 $28 \times 28$ gray scale handwritten digit images evenly distributed in 10 classes. The training set contains 60000 images evenly distributed in the 10 classes, and the test set contains the remaining 10000 images. The backdoor pattern is added to 1000 clean images from class ‘0’. These images are then labeled to class ‘9’ and used for poisoning the training set. The structure of the classifier is shown in Table VI. The training was performed for 30 epochs with mini-batch size 32 and Adam optimizer. No data augmentation was applied during the training.

| Layer | Filter/Node | Stride | Activation |
|-------|-------------|--------|------------|
| conv1 | 64 $\times$ 3 $\times$ 3 | 1 | ReLU |
| conv2 | 64 $\times$ 3 $\times$ 3 | 1 | ReLU |
| pool1 | max, 2 $\times$ 2 | 2 | \ |
| fc1  | 128 | \ | ReLU |
| fc2  | 10 | \ | Softmax |

CIFAR-100 consists of 60000 $32 \times 32$ color images from 100 classes, 600 images per class. The training set contains 50000 images evenly distributed in the 100 classes; the test set contains 10000 images. The training set is poisoned using 1000 images with the same “chess board” pattern. These images are created using clean images from the “beaver”, “dolphin”, “otter”, “seal” and “whale” classes, 200 per class, and are labeled to the “orchid” class. The structure of the classifier is ResNet-32 [8]. The training is performed for 200 epochs, with mini-batch size of 32, using the Adam optimizer, and with the training data augmentation option.

We also created a “CIFAR-50” data set, which consists of training and test images from the first 50 classes of the CIFAR-100 data set. Again, the attack is devised using the same source classes, target class and number of images as the attack for CIFAR-100. The structure of the classifier and the training settings are also the same as for CIFAR-100.

Lastly, we reconsider CIFAR-10, but do not allow sufficient training; hence the accuracy on clean test images will be low, with or without attacks. This is achieved by using the same training settings as described in Section II-B for training, but with the training data augmentation option disabled. Here
we simulate practical scenarios where the training resources are limited, or where the training is poorly optimized. The attack for this test is crafted in the same way as for the BD-G-S group.

TABLE VII: Accuracy on clean test images when the classifier is not attacked, attack success rate, and accuracy on clean test images for the classifier under attack, for the four data sets.

| Data Set       | Test accuracy (w/o attack) | Attack success rate | Test accuracy (w/attack) |
|----------------|----------------------------|--------------------|--------------------------|
| MNIST          | 0.994                      | 1.00               | 0.994                    |
| CIFAR-50       | 0.791                      | 0.980              | 0.783                    |
| CIFAR-100      | 0.731                      | 0.986              | 0.723                    |
| CIFAR-10 (no data aug.) | 0.843                  | 0.973              | 0.837                    |

In Table VII we show that the attack is successful (high attack success rate and negligible degradation in clean test accuracy) on all four data sets.

TABLE VIII: Order statistic p-value when applying AD-J-P and AD-J-C to each of the two classifiers (one attacked and the other not) for each data set, respectively.

| Data Set       | AD-J-P w/o attack | AD-J-P w/attack | AD-J-C w/o attack | AD-J-C w/attack |
|----------------|-------------------|-----------------|-------------------|-----------------|
| MNIST          | 0.890             | U.F.            | 0.797             | U.F.            |
| CIFAR-50       | $1.76 \times 10^{-2}$ | U.F.          | 0.291             | U.F.            |
| CIFAR-100      | $2.75 \times 10^{-4}$ | U.F.          | 0.222             | U.F.            |
| CIFAR-10 (no data aug.) | $6.39 \times 10^{-2}$ | $6.72 \times 10^{-9}$ | 0.476             | $1.65 \times 10^{-9}$ |

In Table VIII for each of the two classifiers (one attacked and the other not attacked) associated with each data set, we show the order statistic p-value when the AD-J-P and AD-J-C variants are applied. For both detection variants, the order statistic p-value, when the classifier has been attacked, is very small (and even causes underflow $^{15}$ (U.F.) for most data sets). For classifiers not attacked, except for MNIST, the order statistic p-value achieved by the AD-J-C variant is clearly larger than the AD-J-P variant — thus, false detections will be rare for AD-J-C with a detection threshold of $\theta = 0.05$. For AD-J-P, there is still a vast range of thresholds that will detect attacks and reject unattacked classifiers, but a threshold smaller than $\theta = 0.05$ is necessary. Note also that reduced classifier accuracy (last row) does not inhibit reliable discrimination of attack from no-attack using a large range of p-value thresholds. As mentioned

$^{15}$ A positive numbers less than $10^{-323}$ will be rounded to 0.
in Section II-F, if there is high confusion between class $s$ and $t$, the perturbation size/energy required to induce $\pi$-level misclassification from $s$ to $t$ may be small. The AD-J-C variant corrects the size/energy of the optimized perturbation using the confusion matrix information. As seen in Table VIII this results in much larger p-values under clean and roughly the same (small) p-values under attack, for all four data sets. This result vindicates our confusion matrix correction scheme. Finally, note that even though the number of classes range from 10 up to 100 across the four data sets, for our method, there is a large, common range for $\theta$ over which strong detection performance is achieved. This is in contrast to there being no good choice for $\theta_{MAD}$ for NC in Figure 7.

E. 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 $\pi$-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.

IV. 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 defenses (post-training) for several backdoor patterns, data sets, and attack settings. Our defense was experimentally shown to be more effective and robust over a range of detection scenarios.
There are several issues that should be considered by future work. 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, 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. This is a challenging problem and a good subject for future work.

There are also several related topics that can be explored in our future work. 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, for an attacker, 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] 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.
[2] 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.
[3] R.O. Duda, P.E. Hart, and D.G. Stork. Pattern classification. Wiley, 2001.
[4] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In Proc. ICLR, 2015.
[5] 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.
[6] 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.

[7] F. R. Hampel. The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, 69, 1974.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016.

[9] 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 (AlSec)*, 2011.

[10] B.-H. Juang and S. Katigiri. Discriminative learning for minimum error classification. *IEEE Trans. on Signal Processing*, 1992.

[11] C. Liao, H. Zhong, A. Squicciarini, S. Zhu, and D.J. Miller. Backdoor embedding in convolutional neural network models via invisible perturbation. CoRR, 2018.

[12] K. Liu, B. Doan-Gavitt, and S. Garg. Fine-Pruning: Defending Against Backdoor Attacks on Deep Neural Networks. In *Proc. RAID*, 2018.

[13] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, and Zhai J. Trojating attack on neural networks. In *Proc. NDSS*, San Diego, CA, Feb. 2018.

[14] 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.

[15] 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.

[16] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. DeepFool: a simple and accurate method to fool deep neural networks, 2016.

[17] B. Nelson and B. Barreno et al. Misleading learners: Co-opting your spam filter. In *Machine Learning in Cyber Trust: Security, Privacy, and Reliability*, 2009.

[18] 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.

[19] 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.

[20] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[21] C. Szegedy, W. Zaremba, I Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *Proc. ICLR*, 2014.

[22] F. Tamer, F. Zhang, A. Juels, M. Reiter, and T. Ristenpart. Stealing machine learning models via prediction apis. In *Proc. USENIX Security Symposium*, 2016.

[23] B. Tran, J. Li, and A. Madry. Spectral signatures in backdoor attacks. In *Proc. NIPS*, 2018.

[24] Trojans in artificial intelligence (TrojAI). [https://www.iarpa.gov/index.php/research-programs/trojai](https://www.iarpa.gov/index.php/research-programs/trojai)

[25] 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.

[26] 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.

[27] 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.

[28] 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.