Towards Class-Oriented Poisoning Attacks Against Neural Networks

Bingyin Zhao Yingjie Lao
Department of Electrical and Computer Engineering
Clemson University, SC, 29634, USA
{bingyiz, ylao}@clemson.edu

Abstract

Poisoning attacks on machine learning systems compromise the model performance by deliberately injecting malicious samples in the training dataset to influence the training process. Prior works focus on either availability attacks (i.e., lowering the overall model accuracy) or integrity attacks (i.e., enabling specific instance based backdoor). In this paper, we advance the adversarial objectives of the availability attacks to a per-class basis, which we refer to as class-oriented poisoning attacks. We demonstrate that the proposed attack is capable of forcing the corrupted model to predict in two specific ways: (i) classify unseen new images to a targeted "supplanter" class, and (ii) misclassify images from a "victim" class while maintaining the classification accuracy on other non-victim classes. To maximize the adversarial effect as well as reduce the computational complexity of poisoned data generation, we propose a gradient-based framework that crafts poisoning images with carefully manipulated feature information for each scenario. Using newly defined metrics at the class level, we demonstrate the effectiveness of the proposed class-oriented poisoning attacks on various models (e.g., LeNet-5, Vgg-9, and ResNet-50) over a wide range of datasets (e.g., MNIST, CIFAR-10, and ImageNet-ILSVRC2012) in an end-to-end training setting.

1. Introduction

In recent years, machine learning has demonstrated superior performance in various fields including computer vision [25], natural language processing [11], autonomous vehicle [4], and healthcare [12]. However, it has also been shown that machine learning models are vulnerable to various types of attacks, including evasion attacks [1, 15, 29, 40] backdoor attacks [6, 16, 7, 8, 9, 41, 35] and poisoning attacks [2, 28, 31, 46, 37, 39, 32, 45]. Evasion attacks occur at the inference phase, which causes misclassification without altering the model. Backdoor attacks raise misclassifications on specific inputs embedded with certain triggers, which requires access to both training and inference phase to inject and activate backdoor triggers. In contrast, poisoning attacks corrupt the model by only injecting malicious training data in the training phase, without requiring attackers take control of model inputs during inference. The category of attacks has drawn particular attention under the scenario where attackers are able to provide training data (e.g. online repositories).

Prior research on poisoning attacks can be broadly classified into two categories: availability attacks that aim at degrading overall model accuracy (i.e., denial-of-service attacks) [2, 44, 28, 31, 30, 21, 39, 45, 32] and integrity attacks that seek to cause misclassification on specific instances (i.e., a targeted image) [37, 47, 19]. While various capabilities of integrity attacks on deep neural networks (DNNs) have been comprehensively investigated, most prior studies of availability attacks are in a very constrained setting. Poisoning availability attacks had mainly focused on binary classification tasks until [31] proposed an efficient algorithm for multi-class attack. However, the authors explicitly pointed out that poisoning availability attack against DNN is challenge and the effect of their method is not significant. On the other hand, poisoned data are notoriously hard to craft due to computational complexity of solving the bi-level optimization (see Section 3 for details). Moreover, the major adversarial goal of prior works on poisoning availability attack is only limited to degrading the overall accuracy.

Given these limitations, we extend the **poisoning availability attack** against DNNs to a per-class basis. We advance the adversarial objectives by formulating two attack tasks: (i) forcing the model to classify all new inputs as a targeted class, which is denoted as the *supplanter class* and (ii) corrupting performance of a specific class, which is named as the *victim class*, while retaining the accuracy of other classes. Note that (ii) can be considered as an extension of the targeted poisoning attack that aims to induce the model to make wrong predictions on a victim class. The essential difference is that we minimize the attack impact on non-victim classes simultaneously. We propose a fast and efficient gradient-based framework for poisoned data generation, which reduces the computational complexity and generates more effective poisoned samples.

2. Related Work

Existing literature studied the poisoning availability attack on binary classification tasks against various learning algorithms such as clustering [3], LASSO [43], collaborative filtering [27], SVM [2] and logistic regression [32]. The main challenge of the poisoning attack is the generation of effective poisoned data. Prior works developed a series of gradient-based approaches for poisoned data generation, including substituting the inner minimization problem with stationary Karush-Kuhn-Tucker (KKT) conditions [28], approximating the non-convex and non-differentiable models to influence functions [23] and gradient ascent optimizations [2, 27]. In this work, we approximate the formulated optimization problems to reduce the computational complexity, which will be discussed in Section 4.

The work in [31] firstly proposed back-gradient optimization and extended poisoning availability attack to multi-class classification. However, the attack is less effective against DNNs. [45] expedited the poisoned data generation using generative models and evaluated the effect on MLP and LeNet. [14] leveraged adversarial examples as poisoned data. Prior works mostly focused on indiscriminately degrading the overall accuracy and disregarded particular prediction error of each class. In this work, we propose algorithms that focus on optimizing the feature information of the most important classes, which is able to achieve the class-oriented adversarial goals as well as facilitate the multi-class poisoning availability attack against DNNs. [13] studied an interesting learning problem that is similar to the class-oriented availability attack. However, the essential difference with [13] is that it mainly focuses on adding smallest bounded noises to the entire training data whereas our work focuses on generating a portion of the most effective poisoned data.

Poisoning integrity attacks and backdoor attacks have been extensively studied in the literature [6, 16, 41, 35, 37, 47, 19]. For instance, [6] imposed backdoor data poisoning and caused face recognition systems to misclassify images that contain a "glasses" pattern. [35] proposed hidden trigger backdoor attacks where triggers are more stealthy and imperceptible to human inspection. [37] proposed an integrity attack approach that employed feature collision to cause misclassification on a specific target image and evaluated on ImageNet dataset. [47] enhanced the transferability of integrity attacks by crafting poisoned images surrounded the targeted image and [19] accelerated poisoned data generation of such attacks. However, as we described in Section 1, poisoning availability attack is fundamentally different from poisoning integrity attacks or backdoor attacks. Note that poisoning integrity attacks and backdoor attacks are not class-oriented since the adversarial goal is only on selected instances instead of object classes.

3. Problem Settings

3.1. Poisoning Availability Attack Setting

We consider the scenario where a DNN is initialized with pre-trained weights and then updated on a poisoned dataset in a full end-to-end fashion. This scenario is one of the most pervasive poisoning attack settings and widely adopted in the state-of-the-art research literature [31, 45, 37, 47, 36] since pre-trained networks are frequently used in real-world applications. To define the problem, let $\boldsymbol{x} \in \mathcal{X}(\mathcal{X} \in \mathbb{R}^d)$ be a d-dimensional input and $y \in \mathcal{Y}$ be the corresponding label. The objective of the classification task is to build up the mapping $\mathcal{F}$: $\mathcal{X} \rightarrow \mathcal{Y}$. We denote the parameters of pre-trained base classifier as θ. The model parameters are updated to θ^* with the incoming new stream of data: $\theta \xrightarrow{(\boldsymbol{x},y)} \theta^*$.

Poisoning availability attacks are typically formulated as a bi-level optimization problem:

$$\arg\max_{\mathcal{D}_p} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{val}} L\left[\mathcal{F}_{\theta^*}\left(\boldsymbol{x}\right), y, \theta^*\right] \tag{1}$$

$$\textbf{s.t.} \quad \theta^* \in \arg\min_{\theta^*\in\Theta} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{tr}\cup\mathcal{D}_p} L\left[\mathcal{F}_{\theta^*}\left(\boldsymbol{x}\right), y, \theta\right], \tag{2}$$

where D_{tr} is the clean training dataset, D_{val} is the validation dataset, D_p is the poisoned dataset, Θ is the possible parameter space, and $L[\cdot]$ is the loss function. The attack aims to find an optimized poisoned dataset, which will be injected into the clean training data for training the benign model and updating parameters. This training process is expressed by the inner minimization. The outer maximization stands for the adversarial objective, which has to be evaluated on the updated parameters found by solving the inner minimization problem.

3.2. Class-Oriented Adversarial Objectives

Other than only focusing on maximizing the overall loss, we take the first step towards extending the adversarial objective to a per-class basis, as illustrated in Figure 1. We impose two new adversarial objectives in addition to the goal of degrading overall accuracy indiscriminately, which is formulated as two optimization problems accordingly.

Problem 1: class-oriented error-generic (COEG) attack. The goal of this attack is to misclassify all or most inputs as a targeted object class, which we named as *supplanter class*. Hence, the overall accuracy will also be degraded. For instance, in Figure 1, class"4" is selected as the supplanter class. For a broader real-world example, the adversary can apply the class-oriented error-generic attack to compromise a military image classifier to classify all the taken images such as birds and planes as a missile (the supplanter class), raising unnecessary panic or even wrongly activating an anti-missile system. The **COEG** attack problem can be formulated as:

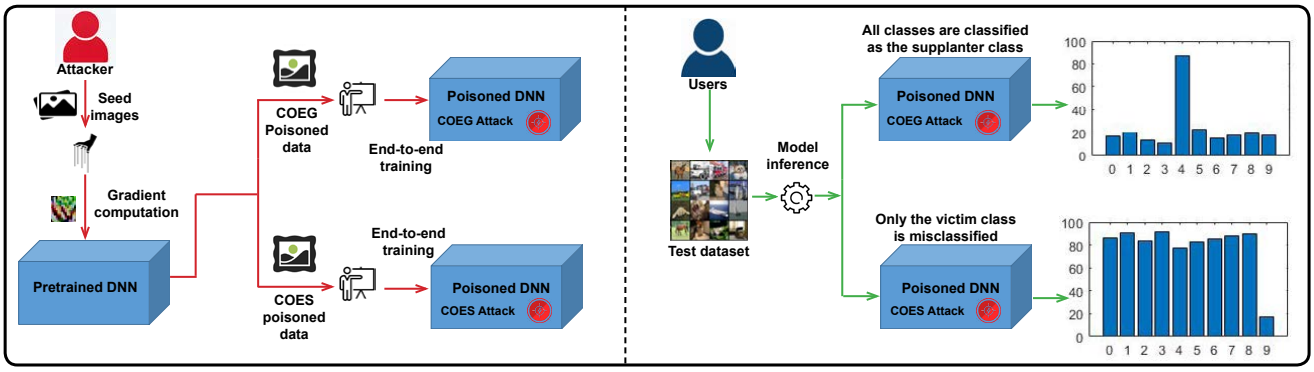

Figure 1. Class-oriented poisoning availability attacks.

$$\arg\max_{\mathcal{D}_p} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{val}} L\left[\mathcal{F}_{\theta^*}\left(\boldsymbol{x}\right), y, \theta^*\right] \tag{3}$$

$$\textbf{s.t.} \quad \theta^* \in \arg\min_{\theta^*\in\Theta} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{tr}\cup\mathcal{D}_p} L\left[\mathcal{F}_{\theta^*}\left(\boldsymbol{x}\right), y_s, \theta\right], \tag{4}$$

where y_s represents the label of the supplanter class.

Problem 2: class-oriented error-specific (COES) attack. The goal is to compromise the classification accuracy only for the inputs from a specific class, which is denoted as *victim class*, while retaining the accuracy of other classes. The bottom right chart in Figure 1 shows the poisoned model behavior where class "9" is selected as the victim class. For a broader example again, the adversary can apply the class-oriented error-specific attack to the military image classifier such that only missiles (the victim class) will not be correctly classified, resulting in severe security risks. The **COES** attack problem is formulated as:

$$\arg\max_{\mathcal{D}_p} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{val}} L\left[\mathcal{F}_{\theta^*}\left(\boldsymbol{x}\right), y_v, \theta^*\right] \tag{5}$$

$$\textbf{s.t.} \quad \theta^* \in \arg\min_{\theta^*\in\Theta} \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{tr}\cup\mathcal{D}_p} L\left[\mathcal{F}_{\theta^*}\left(\boldsymbol{x}\right), y_{\bar{v}}, \theta\right], \tag{6}$$

where y_v and $y_{\bar{v}}$ represent the labels of the victim class and non-victim classes, respectively.

3.3. Class-Oriented Evaluation Metric

We propose two class-oriented evaluation metrics to assess the performance of class-oriented poisoning attacks.

Change-to-Target (CTT) rate is designed as an evaluation metric for the **COEG** attack, which indicates the percentage of images that are classified as a targeted supplanter class ($\mathcal{C}_s$) due to the poisoning attack. CTT rate for a class $\mathcal{C}_k$ is formally defined over a validation dataset $\mathcal{D}_{val}$ as:

$$CTT(\mathcal{C}_k) = \frac{1}{N_k} \sum_{\substack{(\boldsymbol{x_i},y_i)\in\mathcal{D}_{val} \\ y_i=y_k}} \left(\mathcal{F}_{\theta^*}(\boldsymbol{x_i})_{y_s} - \mathcal{F}_{\theta}(\boldsymbol{x_i})_{y_s}\right), \tag{7}$$

$$\mathcal{F}_{\theta}(\boldsymbol{x_i})_{y_s} = \begin{cases} 1 & \text{if } \mathcal{F}_{\theta}(\boldsymbol{x_i}) = y_s \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

where N_k is the total number of images in the class $\mathcal{C}_k$ and y_k is the corresponding categorical label, while $\mathcal{F}_{\theta}$ and $\mathcal{F}_{\theta^*}$ are the model inference results before and after the poisoning attack, respectively. Then, the overall CTT rate over $\mathcal{D}_t$ can be calculated by weighted averaging the CTT rates of all non-supplanter classes:

$$CTT = \frac{\sum_{\substack{k=1 \\ k\neq s}}^{K} \left(N_k \cdot CTT(\mathcal{C}_k)\right)}{\sum_{\substack{k=1 \\ k\neq s}}^{K} N_k}, \tag{9}$$

where K is the total number of classes.

Change-from-Target (CFT) rate is specifically used for evaluating the **COES** attack, which shows the percentage of images from a targeted class are misclassified due to the poisoning attack. Similarly, CFT rate for a class $\mathcal{C}_k$ is defined by Equation (10):

$$CFT(\mathcal{C}_k) = \frac{1}{N_k} \sum_{\substack{(\boldsymbol{x_i},y_i)\in\mathcal{D}_{val} \\ y_i=y_k}} \left(\mathcal{F}_{\theta^*}(\boldsymbol{x_i})_{y_k} - \mathcal{F}_{\theta}(\boldsymbol{x_i})_{y_k}\right). \tag{10}$$

3.4. Threat Model

We consider a threat model that is consistent with prior works on poisoning availability attacks [28, 23, 31, 45, 32], which assumes the adversary to have the knowledge of the learning algorithm, hyper-parameters and clean training data. The attack is performed in an end-to-end training setting on a benign model. The adversary is able to inject crafted poisoned data and assign labels to the training dataset, which also holds the same assumption as the prior literature [2, 27, 5, 31, 32]. However, it is worth mentioning that our proposed approach also works under a more practical and strict scenario where the adversary only has limited knowledge of the model where only model architecture and

pre-trained weights are required, while neither the learning algorithm nor the original training dataset are assumed to be known to the adversary.

4. Class-Oriented Poisoning Attack Methods

4.1. COEG Attack

Intuitively, we expect retraining images with the label of the supplanter class, similar to the flipped-label poisoning attack [44], would have the potential to shift predictions of all classes towards the supplanter class. This is the most straightforward approach that does not even require crafting poisoned data. However, our experimental results demonstrate that such attacks, when applied to multi-class classification tasks, neither effectively degrade the overall accuracy nor achieve the class-oriented adversarial goal of the COEG attack for the neural network models (see Section 5).

Alternatively, we can directly solve Equations (3)-(4) to get poisoned data through gradient ascent. However, for non-convex learning problem such as in DNN, it is difficult to compute $\frac{\partial \theta^*}{\partial D_p}$. To simplify the poisoned data generation and improve the effectiveness of poisoning towards the supplanter class, we develop a novel and efficient method to craft poisoned data. We leverage the fact that the probabilities assigned to other object classes of a well-trained model reveal how much feature information of these incorrect classes is associated with the corresponding image by the model [18]. Thus, we hypothesize that if an input image only contains feature information of its ground-truth class, training such an image with the supplanter class label will force the model to expand the decision boundary of the supplanter class to the maximum degree.

We follow the direction of prior works, which exploit the logit outputs to distill knowledge of neural networks [18], catch features [20], and defend against adversarial examples [22], to control the feature information through the logit outputs. Our algorithm starts with a seed image $\boldsymbol{x_o}$ that is arbitrarily picked from any class other than the supplanter class, and then attempts to retain the feature information associated with the ground-truth class and reduce the feature information of other classes by enlarging/dwindling the corresponding logit outputs. We here denote $f(\cdot)$ as the logit output function of the neural network and f_{y_k} as the corresponding logit to the categorical label y_k. The objective of our algorithm can be expressed by the following minimization objective function:

$$L = \lambda_k \cdot L_{\sum f_{y_k}} - L_{f_{y_o}}, \tag{11}$$

$$\begin{aligned} L_{f_{y_o}} &= f_{y_o}(\boldsymbol{x}), \\ L_{\sum f_{y_k}} &= \sum_{\substack{k=1 \\ k \neq o}}^{K} f_{y_k}(\boldsymbol{x}). \end{aligned} \tag{12}$$

Algorithm 1: COEG Poisoned Data Generation

Input: $\boldsymbol{x_o}$: seed image, y_o: seed image label, y_s: supplanter class label, T: max number of optimization iterations, hyper-parameters λ, ϵ
Output: poisoned image $\boldsymbol{x_p}$, poisoned label y_p
Initialize: $\boldsymbol{x_{p_0}} = \boldsymbol{x_o} - \epsilon \cdot sign\Big(\nabla_{\boldsymbol{x_o}}\big(\lambda \cdot L_{f_{y_s}} - L_{f_{y_o}}\big)\Big)$
while $t < T$ **do**
  Compute the gradient: $\nabla = \nabla \boldsymbol{x_{p_t}}\big(\lambda \cdot L_{f_{y_s}} - L_{f_{y_o}}\big)$
  Update the image: Clip$\{\boldsymbol{x_{p_{t+1}}} = \boldsymbol{x_{p_t}} - \epsilon \cdot sign(\nabla)\}$
  if $f_{y_s}(\boldsymbol{x_{p_{t+1}}}) > f_{y_s}(\boldsymbol{x_{p_t}})$ **or** $f_{y_o}(\boldsymbol{x_{p_{t+1}}}) < f_{y_o}(\boldsymbol{x_{p_t}})$ **then**
    break
  end if
end while
Assign $y_p = y_s$
Return $\boldsymbol{x_p}, y_p$

where $\boldsymbol{x}$ is the image being optimized, which is initialized with the seed image $\boldsymbol{x_o}$. y_o is the corresponding categorical ground-truth label, $f_{y_o}(\boldsymbol{x})$ is the logit output of the ground-truth class, $f_{y_k}(\boldsymbol{x})$ is the logit output of each other classes, and finally $\boldsymbol{x_p}$ stands for the optimized poisoned image. λ_k is used to control the importance of loss terms. We attempt to maximize $L_{f_{y_o}}$ and minimize $L_{\sum f_{y_k}}$ simultaneously. Alternatively, from the perspective of entropy, we expect such optimization would also reduce the following information entropy $H[\cdot]$:

$$H\big[\sigma\big(f(\boldsymbol{x_p})\big)\big] = -\sum_{k=1}^{K} p_k \cdot log(p_k) \to 0, \tag{13}$$

where p_k is the probability for each class that is converted from the logit f_{y_k} using *softmax*.

However, solving the optimization problem can be computationally expensive, especially for large-scale datasets that have hundreds or thousands of classes. Based on the facts that (i) the classification is determined by the largest probability and (ii) only the supplanter class is the target, we consider an approximation that simplifies the task to retain feature information of the ground-truth class and eliminate feature information of the supplanter class. In other words, we only focus on the two most important classes instead of all classes:

$$\begin{aligned} L &= \lambda \cdot L_{f_{y_s}} - L_{f_{y_o}}, \\ L_{f_{y_s}} &= f_{y_s}(\boldsymbol{x}). \end{aligned} \tag{14}$$

By solving the minimization using gradient descent, the poisoned image $\boldsymbol{x_p}$ is updated through one backward pass:

$$\boldsymbol{x_p} = \boldsymbol{x_o} - \epsilon \cdot sign\Big(\nabla_{\boldsymbol{x_o}}\big(\lambda \cdot L_{f_{y_s}} - L_{f_{y_o}}\big)\Big), \tag{15}$$

where $\epsilon > 0$ is the change rate. This update step can also be executed for several rounds to further enhance the poisoning effect. Algorithm 1 presents the details of poisoned data generation for the COEG attack.

4.2. COES Attack

The COES attack is fundamentally more challenging than the COEG attack. It requires not only compromising the accuracy of the targeted victim class but also maintaining the performance of non-victim classes, i.e., a high CFT rate for the victim class while low CFT rates for all the other classes. However, as observed in our single instance attack experiment as well as prior poisoning availability attacks [33], poisoning with data from only one class will shift the distribution of other classes to some extent. Hence, a set of poisoned data from more than a single class is necessary to achieve this adversarial goal. Intuitively, simply training images from all classes except the victim class may achieve this adversarial goal. However, such methods are inefficient in the end-to-end retraining scenario where limited training data and small learning rates are usually applied, as they always require much more training data and training epochs for poisoning.

To this end, we propose another gradient-based algorithm for the COES attack. The entire procedure of our poisoned data generation is presented in Algorithm 2. We craft the poisoned dataset as follows: (i) pick a same number of arbitrary images from each class, (ii) enlarge/dwindle feature information of the corresponding classes for each image, as detailed in Algorithm 2, and (iii) assign the ground-truth labels to the non-victim classes and the targeted label y_p to the victim class. Specifically, for images from the victim class, we apply the same operations as in Algorithm 1, while we only increase the feature information of their ground-truth classes for images from other classes. The objective of COES attack can be expressed as:

$$L = \begin{cases} \lambda \cdot L_{f_{y_s}} - L_{f_{y_o}}, & \text{if } \boldsymbol{x_o} \in \mathcal{C}_v \\ L_{f_{y_o}}, & \text{otherwise} \end{cases} \tag{16}$$

Similarly, the poisoned images $\boldsymbol{x_p}$ are updated through a backward pass:

$$\begin{cases} \boldsymbol{x_o} - \epsilon \cdot sign\Big(\nabla_{\boldsymbol{x_o}}\big(\lambda \cdot L_{f_{y_s}} - L_{f_{y_o}}\big)\Big), & \text{if } \boldsymbol{x_o} \in \mathcal{C}_v \\ \boldsymbol{x_o} + \epsilon \cdot sign\Big(\nabla_{\boldsymbol{x_o}}\big(L_{f_{y_o}}\big)\Big), & \text{otherwise} \end{cases} \tag{17}$$

5. Experiments

5.1. Experimental Settings

As described above, we consider a similar setting as in prior works [45, 37]. We apply our proposed class-oriented poisoning availability attacks to multi-class image classification tasks using three widely-used datasets (MNIST, CIFAR-10, and ImageNet-ILSVRC2012) against popular neural network models (LeNet-5 [26], Vgg-9 [38], and ResNet-50 [17], respectively). Model details are presented in the appendix. For comparison with prior works, we implement the flipped-label (FL) attack [44] and the direct gradient method (DGM) [45] as the baseline attacks for MNIST and CIFAR-10. We also examine our poisoning availability attacks on ImageNet against a ResNet-50 model, which we hope to serve as a baseline comparison for future works. To illustrate the effectiveness of our methods under a fair comparison, we minimize the impact of significant model shifts due to large learning rates by setting the initial learning rate of the poisoning attack close to the final learning rate of the base model and applying the same decay strategy during each attack. All networks are implemented with TensorFlow and experiments are run on NVIDIA Tesla V100 GPUs.

Algorithm 2: COES Poisoned Data Generation

Input: $\boldsymbol{x_k} \in \boldsymbol{X}$: the set of seed images,
$y_k \in Y$: the set of labels associated to X,
y_p: poisoned label, T: max number of iterations,
hyper-parameters λ, ϵ
Output: poisoned dataset $\boldsymbol{X_p}$, poisoned label set Y_p
if $\boldsymbol{x_k} \in \mathcal{C}_v$ **then**
  Apply Algorithm 1
  Assign $y_k = y_p$
else
  Initialize: $\boldsymbol{x_{p_0}} = \boldsymbol{x_o} + \epsilon \cdot sign\Big(\nabla_{\boldsymbol{x_o}}\big(L_{f_{y_o}}\big)\Big)$
  while $t < T$ **do**
    Compute the gradient: $\nabla = \nabla \boldsymbol{x_{p_t}}\big(L_{f_{y_o}}\big)$
    Update the image:
    Clip$\{\boldsymbol{x_{p_{t+1}}} = \boldsymbol{x_{p_t}} + \epsilon \cdot sign(\nabla)\}$
  end while
end if
Assign $\boldsymbol{X_p} = \boldsymbol{X}$; $Y_p = Y$
Return $\boldsymbol{X_p}$, Y_p

5.2. Experimental Results of COEG Attack

5.2.1 Single instance attack.

We first evaluate the effectiveness of our COEG attack under the single instance attack setting [45] that implements the poisoning attack by only training with a sole poisoned point. Hyper-parameters in Algorithm 1 are set as: $\lambda = 1, \epsilon = 0.3$ for MNIST and CIFAR-10. We present the results of MNIST and CIFAR-10 along with baseline methods (FL and DGM) in Figure 2.

It can be seen that our proposed attack is highly effective in degrading the overall model accuracy and increasing the CTT rate for the supplanter class. For example, we increase

Table 1. Comparison of CTT and accuracy before and after the COEG attack on ImageNet.

	Top-1 Accuracy	Top-5 Accuracy	CTT
Vanilla ResNet-50	74.87%	92.02%	–
Poisoned by Our Attack	6.73%	15.00%	85.60%

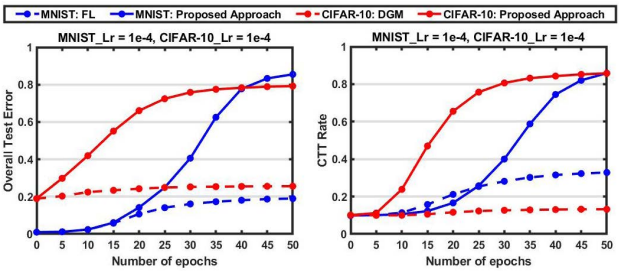

Figure 2. Error and CTT rate comparison of single instance attack. Classes '4' and 'deer' are selected as the supplanter class for MNIST and CIFAR-10, respectively.

the test error from 20% to ∼70% and the CTT rate from 10% to ∼60% on the CIFAR-10 dataset within 20 epochs. Besides, with the increase of training epochs, our method can still consistently achieve higher test errors and CTT rates than baseline attacks, which only increase the test error and CTT rate by ∼8% and ∼0.5%, respectively, even after 50 training epochs. The result for ResNet-50 on ImageNet with hyper-parameters $\lambda = 1, \epsilon = 0.5$ is presented in Table 1. Our proposed method achieves a CTT rate of 85.60% within 20 epochs in the single instance attack. One may argue that the success is due to the bias yielded from training with a single data point. However, for a robust benign model, the accuracy drop and CTT rate of DGM and FL attacks using the same training strategy with a single instance are limited, as shown in Figure 2, while our attack achieves much better performance.

For scenarios where the adversary has no knowledge of the training process, it is also essential to study the impact of different learning rates on the attack. Our conclusion is consistent with prior work [42] that lower learning rates yield less effectiveness for the attack. However, our method still outperforms the baseline attacks when the learning rate is low. We consider a practical adversary model that attackers can only inject the poisoned data but have no knowledge of the learning algorithms and hyper-parameters. To this end, it is worth studying the effect of poisoning attacks with different hyper-parameter settings. Since we have achieved a superior effect with a higher learning rate in Figure 2, we further study the impact of a lower learning rate for the single instance attack. We experiment on both MNIST and CIFAR-10 datasets with a learning rate of 5×10^{-5} and use the same baseline attacks for comparison. We find that a lower learning rate reduces the effect of the poisoning attack in this setting, the alteration of the decision boundary provided from each poisoned sample is decreased. However, while it requires more attack iterations to arrive at the maximum poisoning effect, our method still outperforms the baseline attacks when the learning rate is lower, as indicated in Figure 3.

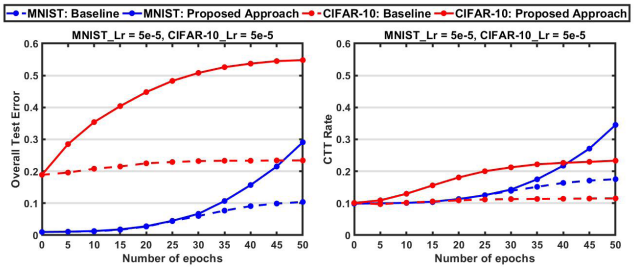

Figure 3. Error and CTT rate comparison of single instance attack with smaller learning rate at 5×10^{-5}. Classes '4' and 'deer' are selected as the supplanter class for MNIST and CIFAR-10. Seed images are from class '6' and 'frog', respectively.

Attack with a set of poisoned data. A more general poisoning attack scenario would allow the attacker to inject a fraction of poisoned data into the clean training dataset, which indeed is a setting often adopted in prior studies [37, 31, 32]. We evaluate the effect of our attack on CIFAR-10 by using 1000 images for training and 9000 images for testing. The number of poisoned samples are controlled by the fraction parameter α. For example, when $\alpha = 0.1$, 100 images are poisoned and 900 images remain clean. In the experiment, class "airplane" is set as the supplanter class. The poisoning attack is conducted at learning rate of 1×10^{-5} over 20 epochs and batch size of 128. We also implement the FL attack as our baseline.

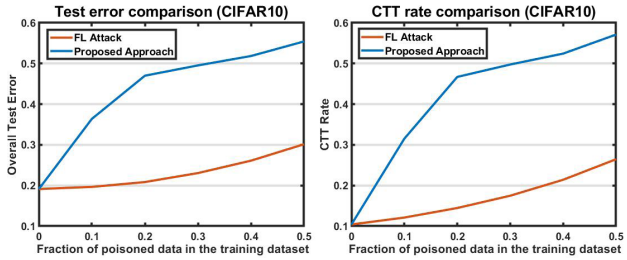

Figure 4. Error and CTT rate comparison of general poisoning attack with different poisoning fraction.

As shown in Figure 4, our attack outperforms the FL attack for all the α values. We achieve over 50% for both CTT rate and test error, which are 25% higher than those of the baseline attack. Note that the proposed attack not only achieves an overall better CTT rate but also performs better on a per-class basis, as indicated in Figure 5. The first column of each confusion matrix represents the number of test images classified as the supplanter class after the poisoning attack. Our approach has a darker color (higher CTT rate)

for every class than the baseline attack. Another interesting observation is that some classes, e.g., "automobile" and "ship", are hard to change towards the supplanter class. This is possibly due to the structural similarity of these classes are less distinguishable to the targeted supplanter class. A recent study finds that different seed/target class pairs and training set size may have significantly different poisoning effect [36]. We also examine our attack's performance from this aspect by experimenting with different seed/target pairs, whose results are presented in the appendix. We find that our attack is particularly effective for smaller training set size and outperforms the FL attack with all the seed/target pairs.

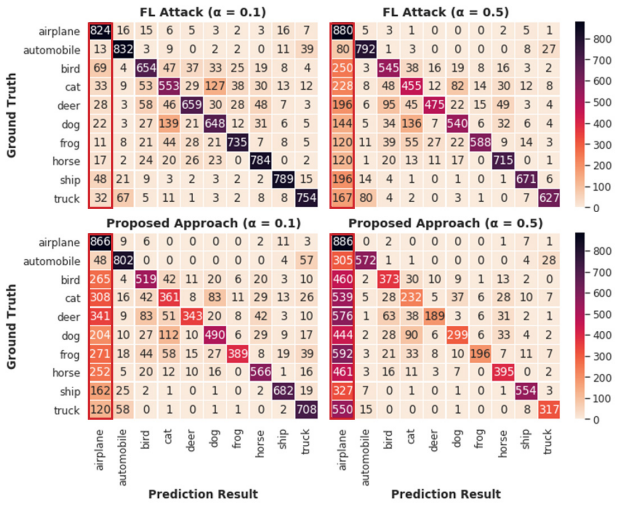

Figure 5. Confusion matrix with different poisoning fractions. Class "airplane" is selected as the supplanter class. Darker color indicates larger change into the supplanter class.

We also study the impact of different learning rates for the general poisoning attack scenario with more poisoned data. Since we achieve the best performance with a dataset size of 500, we keep the same size for this experiment. We increase the learning rate from 1×10^{-5} to 1×10^{-4} and keep the remaining settings the same as the previous experiments. Experiment results are shown in Figure 6. For the FL attack, both the test error and CTT rate are proportional to the learning rate. In contrast, the proposed attack with a lower learning rate achieves a higher test error for $\alpha = 0.2 \sim 0.4$, while yielding a higher CTT rate for $\alpha > 0.1$. This may be attributed to the fact that higher learning rate also amplifies the impact of clean data and partially offsets the effect of poisoned data. Despite this, our proposed attack still achieve better performance compared to the baseline attack in both criteria.

Comparison with existing works. We compare our approach with state-of-the-art poisoning availability attacks. Since prior works are not class-oriented, we only compare the accuracy drop. Taking the different experimental settings

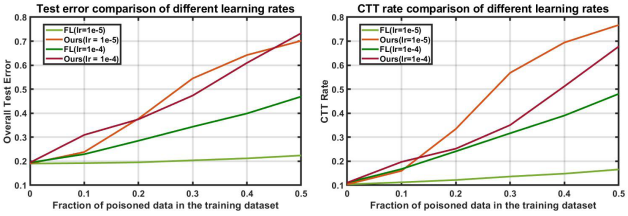

Figure 6. Comparison of different learning rates for the attack with a set of poisoned data.

such as victim model and model accuracy into account, we refer to the best results reported in these papers for fair comparison. Also note again that this paper, to the best of our knowledge, is the first work to evaluate poisoning availability attack on ImageNet. So we only compare the performance on MNIST and CIFAR-10. As shown in Table 2, our proposed approach shows superior adversarial capability.

Table 2. Accuracy drop comparison with prior works.

Approach	Dataset	Accuracy Drop	Victim Model
[31]	MNIST	∼10%	LR classifier
[45]	MNIST	∼80%	2 layers NN
Ours	MNIST	∼80%	LeNet
[32]	CIFAR-10	∼12%	DNN
Ours	CIFAR-10	∼30%	DNN

5.3. Experimental Results of COES Attack

Since the adversarial goal of the COES attack is to subvert only one class without degrading the performance of other classes, there are two important metrics for this task: 1) CFT rate of the victim class should be as high, and 2) CFT rates of the non-victim classes should be as low as possible. Note that the highest achievable CFT rate is upper-bounded by the accuracy of the victim class in the base model. Since the single image attack naturally contradicts to the CEOS adversarial goal (as explained in *Methods Section*), we only consider the general poisoning attack and keep the same training settings and hyper-parameters as in the COEG attack. We present the CFT rate of each class for poisoning attacks on CIFAR-10 in Table 3, where class "truck" is selected as the victim class and "airplane" is selected as the poisoned label. $\alpha = 0.5$ is used. We also evaluate the effect of different values of α in the appendix. Two different types of FL attacks are implemented for comparison: FL-1 flips the label of all poisoned images; FL-2 only flips the label of poisoned images from the victim class.

It can be seen that FL-2 is able to keep the CFT rates of all non-victim classes relatively low; however, the CFT rate of the victim class is only 8.01%. While the FL-1 attack achieves a CFT rate of 18.42% for the victim class, it also largely increases the CFT rates of non-victim classes, which indeed verifies the difficulty of achieving the class-oriented

Table 3. CFT rate of each CIFAR-10 class by poisoning.

Class / Attack	airplane	automobile	bird	cat	deer
FL-1	-3.90%	3.87%	16.80%	16.20%	32.65%
FL-2	-3.01%	-1.44%	-2.05%	-2.29%	2.57%
Ours	-7.07%	0.88%	-0.41%	2.29%	5.27%
Class / Attack	dog	frog	horse	ship	truck(victim)
FL-1	19.09%	22.44%	8.34%	15.45%	18.42%
FL-2	2.67%	-0.59%	-0.11%	-0.78%	8.01%
Ours	3.32%	-0.47%	0.11%	0.22%	51.14%

adversarial goal by using prior poisoning attack methods. Compared to the baseline attacks, our proposed approach can effectively increase the CFT rate of the victim class to 51.14%, which significantly surpasses the performance of both baseline attacks. Meanwhile, our method is able to retain the CFT rates of the non-victim classes to be less than 5.27%. In some cases, the proposed approach even improves the accuracy of some non-victim classes, which is indicated by the negative CFT rates in our experimental results. Such performance is beyond our expectation, which however does not contradict the adversarial objective of COES attack, i.e., only degrading the accuracy of the victim class. Similar to the COEG attack, we find that the proposed COES attack method is more resilient to the variation of learning rate than baseline attacks. We present these results in the appendix.

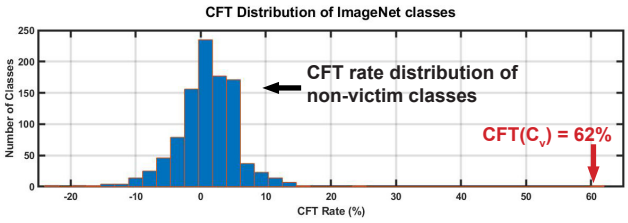

Figure 7. The CFT rate distribution of ImageNet classes by the COES attack.

For ImageNet, we inject 100 poisoned images into 1000 clean training images ($\alpha = 0.1$). Due to the large number of object classes, we present the distribution of the final CFT rates, as shown in Figure 7. Our method achieves a CFT rate of 62% for the victim class, while successfully maintaining low CFT rates for non-victim classes. Compared to the performance on CIFAR-10, although we still accomplish the adversarial goal, we find our attack is slightly less effective on ImageNet (i.e. CFT of non-victim classes are more difficult to control). A possible rationale behind this phenomena is that it becomes harder to completely decouple the feature information of one class from all other classes during the poisoning, with the number of classes scaling up. On the other hand, our experimental results also reveal the importance of studying poisoning availability attack on large-scale/dimensional dataset, which lacks a systematic study yet in the existing literature.

6. Possible Defenses

Since the main objective of this paper is to extend the adversarial capability of poisoning availability attack to a per-class basis on deep neural networks, we expect the proposed attacks to have similar performance as prior poisoning availability attacks in general when evaluated under possible defenses.

Data sanitization is a defensive technique against poisoning attack that works by distinguishing and removing outliers (poisoned data) from the training dataset [10, 34]. However, it has been shown that a broad range of data sanitization can be easily compromised or bypassed [24]. Therefore, we can also leverage such techniques for our proposed attacks to evade detection. In fact, in most recent works on poisoning attacks, data sanitization is no longer considered a certified defensive strategy [32, 21, 42].

Alternatively, a possible countermeasure is to periodically check the accuracy and/or loss of the learning models [31, 45]. Although expensive in terms of cost and time, these approaches are intuitively effective based on the fact that poisoning availability attack aims at degrading the accuracy. Since the poisoned data are tailored to influence the learning model's training process maliciously, we suggest exploiting averaged stochastic gradient classifier [42] and combinational models such as bagging [27], where the classification results are no longer dependent on a single model, to defend against the poisoning attack. However, the overhead for deploying multiple classifiers should also be carefully considered.

7. Conclusions

This paper introduced the concept of class-oriented poisoning attack. We formulated two attack problems, i.e., "COEG" and "COES", which seek to compromise the model behavior on a per-class basis. Accordingly, we defined two new metrics to evaluate the performance of poisoning attacks at the class level. Our proposed gradient-based algorithms successfully achieved the class-oriented adversarial objectives through manipulating the feature information in images for poisoned data generation. The effectiveness of the proposed methods is comprehensively studied in our experiments.

Acknowledgment

This work is partially supported by the National Science Foundation award 2047384.

References

[1] Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, pages 274–283, 2018.

[2] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning, ICML*, 2012.

[3] Battista Biggio, Ignazio Pillai, Samuel Rota Bulò, Davide Ariu, Marcello Pelillo, and Fabio Roli. Is data clustering in adversarial settings secure? In *Proceedings of the 2013 ACM workshop on Artificial intelligence and security*, pages 87–98, 2013.

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016.

[5] Cody Burkard and Brent Lagesse. Analysis of causative attacks against svms learning from data streams. In *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, pages 31–36, 2017.

[6] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *CoRR*, abs/1712.05526, 2017.

[7] Joseph Clements and Yingjie Lao. Backdoor attacks on neural network operations. In *2018 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2018, Anaheim, CA, USA, 2018*, pages 1154–1158. IEEE, 2018.

[8] Joseph Clements and Yingjie Lao. Hardware trojan attacks on neural networks. *CoRR*, abs/1806.05768, 2018.

[9] Joseph Clements and Yingjie Lao. Hardware trojan design on neural networks. In *IEEE International Symposium on Circuits and Systems, ISCAS 2019, Sapporo, Japan, 2019*, pages 1–5. IEEE, 2019.

[10] Gabriela F Cretu, Angelos Stavrou, Michael E Locasto, Salvatore J Stolfo, and Angelos D Keromytis. Casting out demons: Sanitizing training data for anomaly sensors. In *2008 IEEE Symposium on Security and Privacy*, pages 81–95. IEEE, 2008.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[12] Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115, 2017.

[13] Ji Feng, Qi-Zhi Cai, and Zhi-Hua Zhou. Learning to confuse: Generating training time adversarial data with auto-encoder. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*, pages 11971–11981, 2019.

[14] Liam Fowl, Micah Goldblum, Ping-Yeh Chiang, Jonas Geiping, Wojtek Czaja, and Tom Goldstein. Adversarial examples make strong poisons. *CoRR*, abs/2106.10807, 2021.

[15] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *3rd International Conference on Learning Representations, ICLR*, 2015.

[16] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[18] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

[19] W Ronny Huang, Jonas Geiping, Liam Fowl, Gavin Taylor, and Tom Goldstein. Metapoison: Practical general-purpose clean-label data poisoning. *arXiv preprint arXiv:2004.00225*, 2020.

[20] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136, 2019.

[21] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy*, pages 19–35. IEEE, 2018.

[22] Harini Kannan, Alexey Kurakin, and Ian J. Goodfellow. Adversarial logit pairing. *CoRR*, abs/1803.06373, 2018.

[23] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, pages 1885–1894, 2017.

[24] Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger data poisoning attacks break data sanitization defenses. *CoRR*, abs/1811.00741, 2018.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[26] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[27] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in neural information processing systems*, pages 1885–1893, 2016.

[28] Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[29] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to

fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[30] Mehran Mozaffari-Kermani, Susmita Sur-Kolay, Anand Raghunathan, and Niraj K Jha. Systematic poisoning attacks on and defenses for machine learning in healthcare. *IEEE journal of biomedical and health informatics*, 19(6):1893–1905, 2014.

[31] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 27–38, 2017.

[32] Luis Muñoz-González, Bjarne Pfitzner, Matteo Russo, Javier Carnerero-Cano, and Emil C. Lupu. Poisoning attacks with generative adversarial nets. *CoRR*, abs/1906.07773, 2019.

[33] Sangdon Park, James Weimer, and Insup Lee. Resilient linear classification: an approach to deal with attacks on training data. In *Proceedings of the 8th International Conference on Cyber-Physical Systems*, pages 155–164, 2017.

[34] Benjamin IP Rubinstein, Blaine Nelson, Ling Huang, Anthony D Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J Doug Tygar. Antidote: understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*, pages 1–14, 2009.

[35] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI*, pages 11957–11965, 2020.

[36] Avi Schwarzschild, Micah Goldblum, Arjun Gupta, John P. Dickerson, and Tom Goldstein. Just how toxic is data poisoning? A unified benchmark for backdoor and data poisoning attacks. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 9389–9398. PMLR, 2021.

[37] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems*, pages 6103–6113, 2018.

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR*, 2015.

[39] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. In *Advances in neural information processing systems*, pages 3517–3529, 2017.

[40] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR*, 2014.

[41] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Label-consistent backdoor attacks. *CoRR*, abs/1912.02771, 2019.

[42] Yizhen Wang and Kamalika Chaudhuri. Data poisoning attacks against online learning. *CoRR*, abs/1808.08994, 2018.

[43] Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? In *International Conference on Machine Learning*, pages 1689–1698, 2015.

[44] Han Xiao, Huang Xiao, and Claudia Eckert. Adversarial label flips attack on support vector machines. In *20th European Conference on Artificial Intelligence.*, pages 870–875, 2012.

[45] Chaofei Yang, Qing Wu, Hai Li, and Yiran Chen. Generative poisoning attack method against neural networks. *CoRR*, abs/1703.01340, 2017.

[46] Bingyin Zhao and Yingjie Lao. Resilience of pruned neural network against poisoning attack. In *13th International Conference on Malicious and Unwanted Software, MALWARE 2018, Nantucket, MA, USA, 2018*, pages 78–83. IEEE, 2018.

[47] Chen Zhu, W. Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *Proceedings of the 36th International Conference on Machine Learning, ICML*, pages 7614–7623, 2019.