Just How Toxic is Data Poisoning? A Unified Benchmark for Backdoor and Data Poisoning Attacks

Avi Schwarzschild[*]
Department of Mathematics
University of Maryland, College Park
avi1@umd.edu

Micah Goldblum[*]
Department of Mathematics
University of Maryland, College Park
goldblum@umd.edu

Arjun Gupta
Department of Robotics
University of Maryland, College Park
arjung15@umd.edu

John P. Dickerson
Department of Computer Science
University of Maryland, College Park
john@cs.umd.edu

Tom Goldstein
Department of Computer Science
University of Maryland, College Park
tomg@umd.edu

Abstract

Data poisoning and backdoor attacks manipulate training data in order to cause models to fail during inference. A recent survey of industry practitioners found that data poisoning is the number one concern among threats ranging from model stealing to adversarial attacks. However, we find that the impressive performance evaluations from data poisoning attacks are, in large part, artifacts of inconsistent experimental design. Moreover, we find that existing poisoning methods have been tested in contrived scenarios, and they fail in realistic settings. In order to promote fair comparison in future work, we develop unified benchmarks for data poisoning and backdoor attacks.

1 Introduction

Data poisoning is a security threat to machine learning systems in which an attacker controls the behavior of a system by manipulating its training data. This class of threats is particularly germane to deep learning systems because they require large amounts of data to train and are therefore often trained (or pre-trained) on large datasets scraped from the web. For example, the Open Images and the Amazon Products datasets contain approximately 9 million and 233 million samples, respectively, that are scraped from a wide range of potentially insecure, and in many cases unknown, sources [14, 20]. At this scale, it is often infeasible to properly vet content. Furthermore, many practitioners create datasets by harvesting system inputs (e.g., emails received, files uploaded) or scraping user-created content (e.g., profiles, text messages, advertisements) without any mechanisms to bar malicious actors from contributing data. The dependence of industrial AI systems on datasets that are not manually inspected has led to fear that corrupted training data could produce faulty models [8]. In fact, a recent survey of 28 industry organizations found that these companies are significantly more afraid of data poisoning than other threats from adversarial machine learning [13].

[*] Authors contributed equally.

Preprint. Under review.

A spectrum of poisoning attacks exist in the literature. *Backdoor data poisoning* causes a model to misclassify test-time samples that contain a "trigger" – a visual feature in images or a particular character sequence in the natural language setting [3, 4, 22, 27]. For example, one might tamper with training images so that a vision system fails to identify any person wearing a shirt with the trigger symbol printed on it. In this threat model, the attacker modifies data at both train time (by placing poisons) and at inference time (by inserting the trigger). *Triggerless* poisoning attacks, on the other hand, do not require modification at inference time [1, 7, 19, 24, 29]. A variety of innovative backdoor and triggerless poisoning attacks – and defenses – have emerged in recent years, but inconsistent and perfunctory experimentation has rendered performance evaluations and comparisons misleading.

In this paper, we develop a unified framework for benchmarking and evaluating a wide range of poison attacks. Our goal is to address a number of weaknesses in the current literature. First, we observe that the reported success of poisoning attacks in the literature is often dependent on specific (and sometimes unrealistic) choices of network architecture and training protocol, making it difficult to assess the viability of attacks in real-world scenarios. Second, we find that the percentage of training data that an attacker can modify, the standard budget measure in the poisoning literature, is not a useful metric for comparisons. This metric invalidates comparisons because even with a fixed percentage of the dataset poisoned, the success rate of an attack can still be strongly dependent on the dataset size, which is not standardized across experiments. Third, we find that some attacks that claim to be "clean label," such that poisoned data still appears natural and properly labeled upon human inspection, are not.

Our proposed benchmarks measure the effectiveness of attacks in standardized scenarios using modern network architectures. We benchmark from-scratch training scenarios and also transfer learning from a pre-trained network. We report results in the white-box, grey-box, and black-box settings, and we constrain poisoned images to be "clean" in the sense of small perturbations. Furthermore, our benchmarks allow for the direct comparison of both backdoor and triggerless attacks. Our benchmarks are publicly available as a proving ground for existing and future data poisoning attacks.

The data poisoning literature contains attacks in a variety of settings from image classification to facial recognition to text classification [24, 3, 4]. While we acknowledge the merits of studying poisoning in a range of modalities, our benchmark focuses on image classification since it is by far the most common setting in the existing literature.

2 A synopsis of triggerless and backdoor data poisoning

Early poisoning attacks targeted support vector machines and simple neural networks [1, 9]. Since these works, various strategies for triggerless attacks have been developed for deep architectures [24, 29, 7, 19]. Early backdoor attacks were not clean-label and contained triggers in the poisoned data [3, 5, 17]. Subsequent backdoor attacks produce poison examples which don't visibly contain the trigger [3, 22, 27]. Poisoning attacks have also precipitated several defense strategies, but sanitization-based defenses may be overwhelmed by some attacks [2, 10, 15, 21].

We focus on attacks that achieve targeted misclassification. That is, under both the triggerless and backdoor threat models, the end goal of an attacker is to cause a target sample to be misclassified as another specified class. Other objectives, such as decreasing overall test accuracy, have been studied, but less work exists on this topic with respect to neural networks [16, 28]. In both triggerless and backdoor data poisoning, the clean images, called *base images*, that are modified by an attacker come from a single class, the *base class*. This class is often chosen to be precisely the same class into which the attacker wants the target image or class to be misclassified.

There are two major differences between triggerless and backdoor threat models in the literature. First and foremost, backdoor attacks alter their targets during inference by adding a trigger. In the works we consider, triggers take the form of small patches added to an image [22, 27]. Second, these works on backdoor attacks cause a victim to misclassify an entire class rather than a particular sample. Triggerless attacks instead cause the victim to misclassify an individual image called the *target image* [24, 29]. This second distinction between the two threat models is not essential; for example, triggerless attacks could be designed to cause the victim to misclassify a collection of images rather than a single target. To be consistent with the literature at large, we focus on triggerless attacks that target individual samples and backdoor attacks that target whole classes of images.

We focus our attention on the *clean-label backdoor attack* and the *hidden trigger backdoor attack*, where poison examples are the result of optimization procedures and do not themselves contain noticeable patches [22, 27]. For triggerless attacks, we focus on the *feature collision* and *convex polytope* methods as they are the most highly cited attacks of the last two years and have appeared at prominent ML conferences [24, 29]. The following section details the attacks that serve as the subjects of our experiments.

Technical details Before formally describing various poisoning methods, we begin with notation. Let X_c be the set of all clean training data, and let $X_p = \{x_p^{(j)}\}_{j=1}^J$ denote the set of J poison examples with corresponding clean base image $\{x_b^{(j)}\}_{j=1}^J$. Let x_t be the target image. Labels are denoted by y and Y for a single image and a set of images, respectively, and are indexed to match the data. We use f to denote a feature extractor network.

Feature Collision (FC) Poisons in this attack are crafted by adding small perturbations to base images so that their feature representations lie extremely close to that of the target [24]. Formally, each poison is the solution to the following optimization problem.

$$x_p^{(j)} = \underset{x}{\operatorname{argmin}} \|f(x) - f(x_b^{(j)})\|_2^2 + \beta \|x - x_b^{(j)}\|_2^2. \tag{1}$$

When we enforce ℓ_∞-norm constraints, we drop the last term in Equation (1) and instead enforce $\|x_p^{(j)} - x_b^{(j)}\|_\infty \leq \varepsilon, \ \forall j$ by projecting onto the ℓ_∞ ball each iteration.

Convex Polytope (CP) This attack crafts poisons such that the target's feature representation is a convex combination of the poisons' feature representations by solving the following optimization problem [29].

$$\begin{aligned} X_p^* = \underset{\{c_j\},\{x_p^{(j)}\}}{\operatorname{argmin}} \quad & \tfrac{1}{2}\frac{\|f(x_t) - \sum_{j=1}^J c_j f(x_p^{(j)})\|_2^2}{\|f(x_t)\|_2^2} \\ \text{subject to} \quad & \textstyle\sum_{j=1}^J c_j = 1 \text{ and } c_j \geq 0 \ \forall\, j, \text{ and } \|x_p^{(j)} - x_b^{(j)}\|_\infty \leq \varepsilon \ \forall j \end{aligned} \tag{2}$$

Clean Label Backdoor (CLBD) This backdoor attack begins by computing an adversarial perturbation to each base image [27]. Formally,

$$\hat{x}_p^{(j)} = x_b^{(j)} + \underset{\|\delta\|_\infty \leq \varepsilon}{\operatorname{argmax}} \mathcal{L}(x_b^{(j)} + \delta, y^{(j)}; \theta), \tag{3}$$

where $\mathcal{L}$ denotes cross-entropy loss. Then, a patch is added to each image in $\{\hat{x}_p^{(j)}\}$ to generate the final poisons $\{x_p^{(j)}\}$. The patched image is subject to an ℓ_∞-norm constraint.

Hidden Trigger Backdoor (HTBD) A backdoor analogue of the FC attack, where poisons are crafted to remain close to the base images but collide in feature space with an image from the target class modified to include the trigger [22]. We let $\tilde{x}_t^{(j)}$ denote a training image from the target class which has been patched (this image is not clean), then we solve the following optimization problem to find poison images.

$$x_p^{(j)} = \underset{x}{\operatorname{argmin}} \|f(x) - f(\tilde{x}_t^{(j)})\|_2^2 \text{ s.t. } \|x - x_b^{(j)}\|_\infty \leq \varepsilon \tag{4}$$

3 Why do we need benchmarks?

Backdoor and triggerless attacks have been tested in a wide range of disparate settings. From model architecture to target/base class pairs, the literature is inconsistent. Experiments are also lacking in the breadth of trials performed, sometimes using only one model initialization for all experiments, or testing against one single target image. We find that inconsistencies in experimental settings have a large impact on performance evaluations, and have resulted in comparisons that are difficult to interpret. For example, in CP the authors compare their ℓ_∞-constrained attack to FC, which is crafted with an ℓ_2 penalty. In other words, these methods have never been compared on a level playing field.

To study these attacks thoroughly and rigorously, we employ sampling techniques that allow us to draw conclusions about the attacks taking into account variance across model initializations and class choice. For a single trial, we sample one of ten checkpoints of a given architecture, then randomly select the target image, base class, and base images. In Section 4, all figures are averages from 100 trials with our sampling techniques.

Disparate evaluation settings from the literature To understand how differences in evaluation settings impact results, we re-create the various original performance tests for each of the methods described above within our common evaluation framework. We try to be as faithful as possible to the original works, however we employ our own sampling techniques described above to increase statistical significance. Then, we tweak these experiments one component at a time revealing the fragility of each method to changes in experimental design.

Establishing baselines For the FC setting, following one of the main setups in the original paper, we craft 50 poisons on an AlexNet variant (for details on the specific architecture, see [12, 24]) pre-trained on CIFAR-10 [11], and we use the ℓ_2-norm penalty version of the attack. We then evaluate poisons on the same AlexNet, using the same CIFAR-10 data to train for 20 more epochs to "fine-tune" the model end-to-end. Note that this is not really transfer learning in the usual sense, as the fine-tuning stage takes places using the same dataset as pre-training, except with poisons inserted [24].

The CP setting involves crafting 5 poisons using a ResNet-18 model pre-trained on CIFAR-10, and then fine-tuning the linear layer of the same ResNet-18 model with a subset of the CIFAR-10 training comprising 50 images per class (including the poisons) [6]. This setup is also not representative of typical transfer learning, as the fine-tuning data is sub-sampled from the pre-training dataset [29]. In this baseline we set $\varepsilon = {}^{25.5}/_{255}$.

One of the original evaluation settings for CLBD uses 500 poisons. We craft these on an adversarially trained ResNet-18 and modify them with a 3×3 patch in the lower right-hand corner. The perturbations are bounded with $\varepsilon = {}^{16}/_{255}$. We then train a narrow ResNet model from scratch with the CIFAR-10 training set (including the poisons) [27].

For the HTBD setting, we generate 800 poisons with another modified AlexNet (for architectural details, see Appendix A.13) which is pre-trained on CIFAR-10 dataset. Then, an 8×8 trigger patch is added to the lower right corner of the target image, and the perturbations are bounded with $\varepsilon = {}^{16}/_{255}$. We use the entire CIFAR-10 dataset (including the poisons) to fine-tune the last fully connected layer of the same model used for crafting. See the left-most bars of Figure 3 for baseline results. Once again, the fine-tuning data in this setup is not disjoint from the pre-training data [22].

Inconsistencies in previous work The baselines defined above do not serve as a fair comparison across methods, since the original works to which we try and stay faithful are inconsistent. Table 1 summarizes experimental settings in the original works. If a particular component (column header) was considered anywhere in the original paper's experiments, we mark a (✓), leaving exes (×) when something was not present in any experiments. We consider data normalization and augmentation as well as optimizers (SGD or ADAM). Table 1 also shows which learning setup the original works considered: frozen feature extractor (FFE), end-to-end fine-tuning (E2E), or from-scratch training (FST), as well as which threat levels were tested, white, grey or black box (WB, GB, BB). We also consider whether or not an ensembled attack was used. The ε values reported are out of 255 and represent the smallest bound considered in the papers; note FC uses an ℓ_2 penalty so no bound is enforced despite the attack being called "clean-label" in the original work. We conclude from Table 1 that experimental design in this field is extremely inconsistent.

4 Just how toxic are poisoning methods really?

In this section, we look at weaknesses and inconsistencies in existing experimental setups, and how these lead to potentially misleading comparisons between methods. We use our testing framework to put triggerless and backdoor attacks to the test under a variety of circumstances, and get a tighter grip on just how reliable existing poisoning methods are.

Table 1: Various experimental designs used in data poisoning research

	Data		Opt.	Transfer Learning			Threat Model				
Attack	Norm.	Aug.	SGD	FFE	E2E	FST	WB	GB	BB	Ensembles	ε
FC	×	×	×	✓	✓	×	✓	×	×	×	-
CP	✓	×	×	✓	✓	×	×	✓	✓	✓	25.5
CLBD	×	✓	✓	×	×	✓	×	×	✓	×	8
HTBD	✓	×	✓	✓	×	×	✓	×	×	×	8

Training without SGD or data augmentation Both FC and CP attacks have been tested with victim models pre-trained with the ADAM optimizer. However, SGD with momentum has become the dominant optimizer for training CNNs. Interestingly, we find that models trained with SGD are significantly harder to poison, rendering these attacks ineffective in practical settings. Moreover, none of the baselines include simple data augmentation such as horizontal flips and random crops. We find that data augmentation, standard in the deep learning literature, also greatly reduces the effectiveness of all of the attacks. For example, FC and CP success rates plummet in this setting to 51.00% and 19.09%, respectively. Complete results including hyperparameters, success rates, and confidence intervals are reported in Appendix A.3. We conclude that these attacks may be significantly less effective against a real practitioner than originally advertised.

Victim architecture matters Two attacks, FC and HTBD, are originally tested on AlexNet variants, and CLBD is tested with a narrow ResNet architecture. These models are not widely used, and they are unlikely to be employed by a realistic victim. We observe that many attacks are significantly less effective against ResNet-18 victims. See Figure 3, where for example, the success rate of HTBD on these victims is as low as 18%. See Appendix A.4 for a table of numerical results. These ablation studies are conducted in the baseline settings but with a ResNet-18 victim architecture. These ResNet experiments serve as an example of how performance can be highly dependent on the selection of architecture.

"Clean" attacks are sometimes dirty Each of the original works we consider purports to produce "clean-label" poison examples that look like natural images. However these methods often produce easily visible image artifacts and distortions due to the large values of ϵ used. See Figure 1 for examples generated by two of the methods. The FC method is tested with an ℓ_2 penalty in the original work, and CP is ℓ_∞ constrained with a large radius of $^{25.5}/_{255}$.

Borrowing from common practice in the evasion attack and defense literature, we test each method with an ℓ_∞ constraint of radius $^{8}/_{255}$ and find that the effectiveness of every attack is significantly diminished [18]. Thus, a standardized constraint on poison examples is necessary for fair comparison of attacks, and these previous attacks are not nearly as threatening under constraints that enforce clean poisons. See Figure 3, and see Appendix A.5 for a table of numerical results.

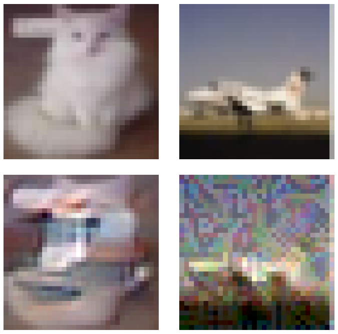

Figure 1: Bases (top) and poisons (bottom). Left: FC perturbs a clean "cat" into an unrecognizable poison. Right: CP generates an extremely noisy poison from a base in the "airplane" class.

Proper transfer learning is vulnerable Of the attacks we study here, FC, CP, and HTBD were originally proposed in settings referred to as "transfer learning." Each particular setup varies, but none are true transfer learning since the pre-training datasets and fine-tuning datasets overlap. For example, FC uses the entire CIFAR-10 training dataset for both pre-training and fine-tuning. Thus, their threat model entails allowing an adversary to modify the training dataset but only for the last few epochs. Furthermore, these attacks use inconsistently sized fine-tuning datasets.

To simulate transfer learning, we test each attack with ResNet-18 feature extractors pre-trained on CIFAR-100. Since we fine-tune on CIFAR-10 data in both cases, we are able to see that these methods actually perform better in the setting with real transfer learning, *i.e.* where the pre-training data and fine-tuning data are not from the same datasets and do not contain the same classes. Figure 3 shows that for every attack aside from CLBD (which is intended for training from scratch), the performance is better when transfer learning is done on data that is disjoint from the pre-training dataset. We conclude that the attacks designed for transfer learning may sometimes benefit from a more realistic setup. See Appendix A.6.

Performance is not invariant to dataset size Existing work on data poisoning measures an attacker's budget in terms of what percentage of the training data they may modify. This begs the question whether percentage alone is enough to characterize the budget. Does the actual size of the training set matter? We find the number of images in the training set has a large impact on attack performance, and that performance curves for FC and CP intersect. When we hold the percentage poisoned constant but change the number of poisons and the size of the training set accordingly, we see no consistent trends in how the attacks are affected. See Figure 2. This observation suggests that one cannot compare attacks tested on different sized datasets by only fixing the percent of the dataset poisoned. See Appendix A.7.

Black-box performance is low Whether considering transfer learning or training from scratch, testing these methods against a black-box victim is surely one of the most realistic tests of the threat they pose. Since, FC, CP and HTBD do not consider the black-box scenario in the original works, we take the poisons crafted using baseline methods and evaluate them on models of different architectures than those used for crafting. The attacks show much lower performance in the black-box settings than in the baselines, in particular FC, CP, and HTBD all have success rates lower than 20%. See Figure 3, and see Appendix A.8 for more details.

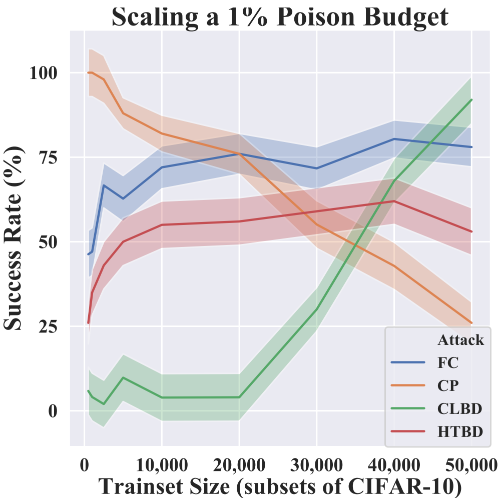

Figure 2: Scaling the dataset size while fixing the poison budget as a percentage of data.

Small sample sizes and non-random targets On top of inconsistencies in experimental setups, existing work on data poisoning often test only on specific target/base class pairs. For example, FC largely uses "frog" as the base class and "airplane" as the target class. CP, on the other hand, only uses "ship" and "frog" as the base and target classes, respectively. Neither work contains experiments where each trial consists of a randomly selected target/base class pair. We find that the success rates are highly class pair dependent and change dramatically under random class pair sampling. Thus, random sampling is critical for performance evaluation. See Appendix A.9 for a comparison of the specific class pairs from these original works with randomly sampled class pairs.

In addition to inconsistent class pairs, data poisoning papers often evaluate performance with very few trials since the methods are computationally expensive. In their original works, FC and CP use 30 and 50 trials, respectively, for each experiment, and these experiments are performed on the same exact pre-trained models each time. And while HTBD does test randomized pairs, they only show results for ten trials on CIFAR-10. These small sample sizes yield wide error bars in performance evaluation. We choose to run 100 trials per experiment in our own work. While we acknowledge that a larger number would be even more compelling, 100 is a compromise between thorough experimentation and practicality since each trial requires re-training a classifier.

Attacks are highly specific to the target image Triggerless attacks have been proposed as a threat against systems deployed in the physical world. For example, blue Toyota sedans may go undetected by a poisoned system so that an attacker may fly under the radar. However, triggerless attacks are generally crafted against a specific target image, while a physical object may appear differently under difference real-world circumstances. We upper-bound the robustness of poison attacks by applying simple horizontal flips to the target images, and we find that poisoning methods are significantly less successful when the exact target image is unknown. For example, FC is only successful 7% of the time when simply flipping the target image. See Figure 3 and Appendix A.10.

Ensemblizing boosts the attacker The only original work, of the works we consider, to discuss ensembles is CP. In the original CP work, the authors claim that FC cannot be ensemblized and still remain clean. However, they strangely do not bound FC's perturbation as they only test FC with the ℓ_2 penalty. We find that ensemblizing both attacks helps in the black-box setting and it can be done with clean poisons. The gap between the two, when compared in exactly the same setting, is much smaller than initially reported by [29]. See Table 2 and Appendix A.11.

Backdoor success depends on patch size Backdoor attacks add a patch to target images to trigger misclassification. In real-world scenarios, a small patch may be critical to avoid being caught. The original HTBD attack uses an 8×8 patch, while the CLBD attack originally uses a 3×3 patch [22, 27]. In order to understand the impact on attack performance, we test different patch sizes. We find a strong correlation between the patch size and attack performance, see Appendix A.12. We conclude that backdoor attacks must be compared using identical patch sizes.

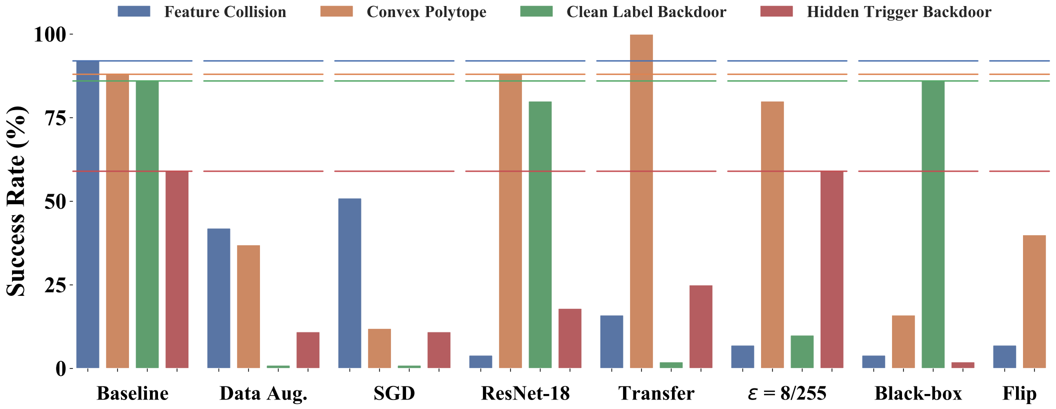

Figure 3: We show the fragility of poisoning methods to experimental design. This figure depicts baselines along with the results of ablation studies. Different methods respond differently to these testing scenarios, lending support to the need for consistent and thorough testing. Horizontal lines denote performance on baselines described in Section 3, and bars represent the results of changing a specific feature in an individual method's baseline. Tables of these results with confidence intervals can be found in the appendices.

5 Unified benchmarks for data poisoning attacks

We propose new benchmarks for measuring the efficacy of **both** backdoor and triggerless data poisoning attacks. We standardize the datasets and problem settings for our three benchmarks below.[2]

Benchmark details Target and base images are chosen from the CIFAR-10 testing and training sets, respectively, according to a seeded/reproducible random assignment. Poison examples crafted from the bases must remain within the ℓ_∞-ball of radius $^8/_{255}$ centered at the corresponding base images. Seeding the random assignment allows us to test against 100 different random choices of base/target, while always using the same 100 choices for each method, thus removing a source of variation from the results. We consider 3 different training modes:

I **Frozen Feature Extractor:** In this transfer learning scenario, a feature extractor is pre-trained on clean CIFAR-100 data. The frozen feature extractor is used while training a linear classification head on a subset of the CIFAR-10 training data that contains poisons. 25 poisons are included in 2,500 training images.

[2] Code is available at `https://github.com/aks2203/poisoning-benchmark`.

II **End-To-End Fine-Tuning:** A feature extractor is pre-trained on CIFAR-100 and then fine-tuned in an end-to-end fashion on a subset of CIFAR-10 training data with poison examples inserted in the training set. 25 poisons are included in 2,500 training images.

III **Training From Scratch:** A network is trained from random initialization on CIFAR-10 with poison examples inserted in the training set. 500 poisons are included in 50,000 training images.

To further standardize these tests, we provide pre-trained architectures to test against: two ResNet-18 models, a VGG11 [25], and a MobileNetV2 [23], all trained on CIFAR-100 data. The parameters of the first ResNet-18 are given to the attacker. We then evaluate the strength of the attacks in white-box, grey-box, and black-box scenarios. For white-box tests in the first two benchmarks, we use the same ResNet-18 given to the attacker for evaluation. In the grey-box scenario, we craft using one ResNet-18 model and test on the other. In the black-box setting, we craft poisons using the ResNet model and test on the VGG11 and the MobileNetV2 models, averaging the results. When training from scratch in the third test, one model of each architecture is trained from a random initialization on the poisoned dataset. We report averages over 100 independent trials for each test. Backdoor attacks can use any 5×5 patch. Note that the number of attacker-victim network pairs is kept small in our benchmark because each of the 100 trials requires re-training (in some cases from scratch), and we want to keep the benchmark within reach for labs with modest computing resources.

Benchmark datasets Our transfer learning tests are done with disjoint sets of pre-training and fine-tuning data, where the fine-tuning data is a subset of CIFAR-10. We choose this subset to be the first 250 images from each class. This amount of data motivates the use of transfer learning, since training from scratch on only 2,500 images yields poor generalization. See Appendix A.13 for examples. We choose to use 25 and 500 base images in our benchmarks. See Appendix A.15 for a case-study in which we investigate how many poisons an attacker may be able to place in a dataset compiled by querying the internet for images.

Benchmark hyperparameters We pre-train all models on CIFAR-100 with SGD for 400 epochs starting with a learning rate of 0.1, which decays by a factor of 10 after epochs 200, 300, and 350. We apply per-channel data normalization, random crops, and horizontal flips, and we use batches of 128 images. We then fine-tune on the poisoned data for 40 epochs with a learning rate that starts at 0.01 and drops to 0.001 after the 30th epoch (this applies to both the frozen feature extractor and end-to-end settings).

When training from scratch, we include the 500 perturbed poisons in the standard CIFAR-10 training set. We use SGD and train for 200 epochs with batches of 128 images and an initial learning rate of 0.1 that decays by a factor of 10 after epochs 100 and 150. Here too, we use data normalization and augmentation as described above.

We find that by using disjoint and standardized datasets for transfer learning, and common training practices like data normalization and scheduled learning rate decay, we overcome the deficits in previous work. Our benchmark can provide useful evaluations of data poisoning methods and meaningful comparisons between them.

6 Conclusion

While the threat of data poisoning is at the forefront of fears around emerging ML systems [26], we conclude that many of the methods claiming to do so do not pose a practical threat. Our benchmark evaluations (Table 2) show that every attack we study is less effective than originally claimed when evaluated in our more realistic (and difficult) settings. See Appendix A.16 for tables with confidence intervals. Although the methods we test show no significant performance in the training from scratch setting, it is our hope that this test proves useful as the field matures. Trepidation on the part of practitioners will be matched by the potential harm of poisoning attacks if stronger methods emerge. The advancement of these methods is inevitable, and our benchmark serves the data poisoning community as a standardized test problem on which to evaluate current and future attack methodologies.

Table 2: Benchmark success rates

Frozen Feature Extractor				End-To-End Fine-Tuning			
Attack	WB (%)	GB (%)	BB (%)	Attack	WB (%)	GB (%)	BB (%)
FC	16.0	7.0	3.50	FC	4.0	3.0	3.5
FC-Ens.	13.0	9.0	6.0	FC-Ens.	7.0	4.0	5.0
CP	24.0	7.0	4.5	CP	17.0	7.0	4.5
CP-Ens.	20.0	8.0	12.5	CP-Ens.	14.0	4.0	10.5
CLBD	3.0	6.0	3.5	CLBD	3.0	2.0	1.5
HTBD	2.0	4.0	4.0	HTBD	3.0	2.0	4.0

Training From Scratch				
Attack	ResNet-18	MobileNetV2	VGG11	Average
FC	0%	1%	3%	1.33%
CP	0%	1%	1%	0.67%
CLBD	0%	1%	2%	1.00%
HTBD	0%	4%	1%	2.67%

Broader impact

Data poisoning has been proposed as a way for malicious actors to disrupt machine learning systems. A recent survey of industry practitioners found that data poisoning is the number one concern among threats ranging from model stealing to adversarial attacks. If successful, data poisoning attacks may harm not only organizations which employ deep learning tools but also the wide body of individuals who use them. This work contains a sober discussion concerning the vulnerability of popular deep learning practices to existing poisoning methods. Our benchmarks help practitioners, especially in sensitive domains, to evaluate their own security vulnerabilities. Additionally, our ablation studies inform practitioners on which practices, from optimizers to architectures to data augmentation, may reduce their risk. We believe that data poisoning may achieve a serious degree of threat in the future, and monitoring the capabilities of these methods is critical for anticipating and avoiding damage.

References

[1] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pages 1467–1474, USA, 2012. Omnipress.

[2] H. Chacon, S. Silva, and P. Rad. Deep learning poison data attack detection. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 971–978, 2019.

[3] X. Chen, C. Liu, B. Li, K. Lu, and D. Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[4] J. Dai, C. Chen, and Y. Li. A backdoor attack against lstm-based text classification systems. *IEEE Access*, 7:138872–138878, 2019.

[5] T. Gu, B. Dolan-Gavitt, and S. Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] W. R. Huang, J. Geiping, L. Fowl, G. Taylor, and T. Goldstein. Metapoison: Practical general-purpose clean-label data poisoning, 2020.

[8] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. *arXiv preprint arXiv:1712.05055*, 2017.

[9] P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894. JMLR. org, 2017.

[10] P. W. Koh, J. Steinhardt, and P. Liang. Stronger data poisoning attacks break data sanitization defenses, 2018.

[11] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[13] R. S. S. Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia. Adversarial machine learning–industry perspectives. *arXiv preprint arXiv:2002.05646*, 2020.

[14] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, et al. The open images dataset v4. *International Journal of Computer Vision*, pages 1–26, 2020.

[15] K. Liu, B. Dolan-Gavitt, and S. Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 273–294. Springer, 2018.

[16] S. Liu, S. Lu, X. Chen, Y. Feng, K. Xu, A. Al-Dujaili, M. Hong, and U.-M. Obelilly. Min-max optimization without gradients: Convergence and applications to adversarial ml, 2019.

[17] Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang. Trojaning attack on neural networks. 2017.

[18] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[19] L. Muñoz-González, B. Biggio, A. Demontis, A. Paudice, V. Wongrassamee, E. C. Lupu, and F. 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. ACM, 2017.

[20] J. Ni. *Amazon Review Data*, 2018 (Accessed 2020). `https://nijianmo.github.io/amazon/index.html`.

[21] N. Peri, N. Gupta, W. R. Huang, L. Fowl, C. Zhu, S. Feizi, T. Goldstein, and J. P. Dickerson. Deep k-nn defense against clean-label data poisoning attacks, 2019.

[22] A. Saha, A. Subramanya, and H. Pirsiavash. Hidden trigger backdoor attacks. *arXiv preprint arXiv:1910.00033*, 2019.

[23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[24] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems*, pages 6103–6113, 2018.

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[26] R. S. Siva Kumar, M. Nystrom, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia. Adversarial machine learning - industry perspectives. *SSRN Electronic Journal*, 2020.

[27] A. Turner, D. Tsipras, and A. Madry. Clean-label backdoor attacks. 2018.

[28] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, and F. Roli. Support vector machines under adversarial label contamination. *Neurocomputing*, 160:53–62, 2015.

[29] C. Zhu, W. R. Huang, H. Li, G. Taylor, C. Studer, and T. Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *International Conference on Machine Learning*, pages 7614–7623, 2019.

A Appendix

A.1 Technical setup

We report confidence intervals of radius one standard error, $\mathcal{E} = \sqrt{\hat{p}(1-\hat{p})/N}$, where $\hat{p}$ is the observed probability of success, and N is the number of trials. If there are fewer than five observed successes or failures, we set $\hat{p} = 1/2$ to upper-bound standard error.

Hyperparameters We use one of seven sets of hyperparameters when training models. We refer to these by name throughout this appendix, and Table 3 shows each setup. For all models trained with SGD, we set the momentum coefficient to 0.9. We always use batches of 128 images and weight decay with a coefficient of 2×10^{-4}. The "Decay Schedule" column details the epochs after which the learning rate drops by the corresponding decay factor.

Table 3: Hyperparameters

	Learning rate				
Setup	Initial	Decay Factor	Decay Schedule	Epochs	Optimizer
A	0.001	0.5	32, 64, 96, 128, 160, 192	200	ADAM
B	0.010	0.1	100, 150	200	SGD
C	0.100	0.1	100, 150	200	SGD
D	0.100	0.1	200, 300, 350	400	SGD
E	0.100	0.1	40, 60	100	SGD
F	0.100	0.1	75, 90	100	SGD
G	0.010	0.1	30	40	SGD

A.2 Baselines

Table 4 shows the baseline performance of each attack. This table reports averages over 100 independent trials with confidence intervals of width one standard error. The experimental setups are summarized in Section 3, and we report additional details here. When we say that an experiment uses a particular architecture, we mean that each trial randomly selects one of ten pre-trained models of this type. The average performances for these sets of pre-trained models are reported in Table 15 below where the hyperparameters and training routines are detailed.

Feature Collision The FC baseline uses an AlexNet variant without data normalization or data augmentation. We use the unconstrained version of this attack with the ℓ_2 penalty in the optimization problem. The algorithm presented in the original work has hyperparameters which we set as follows [24]. We add a watermark of the target image with 30% opacity to each base before the optimization and we use a step size of 0.0001 with the maximum number of iterations set to 1,200. When fine tuning on the poisoned data, we train for 20 epochs with ADAM and a fixed learning rate of $0.001 \times 0.5^6 = 0.00015625$, which is the smallest learning rate used when pre-training.

Convex Polytope The CP baseline uses a ResNet-18 with data normalization (without data augmentation). In the poison crafting procedure, we use the ADAM optimizer with a learning rate of 0.04 for a maximum of 4,000 iterations or when the CP loss is less than or equal to 1×10^{-6}. We bound the perturbations with $\varepsilon = 25.5/255$. Then, we fine tune the model with ADAM for 10 epochs on the poisoned data with a learning rate of 0.1.

Clean Label Backdoor The CLBD baseline is a training from scratch scenario. The model used for crafting is an adversarially trained ResNet-18, and we use 20-step PGD with a step size of $4/255$ and ε of $16/255$ to compute the adversarial perturbations [18]. Then we train a narrow ResNet model (see [27] for architectural details) from scratch using hyperparameter set E as defined in Table 3.

Hidden Trigger Backdoor The HTBD baseline uses a modified AlexNet model with data normalization. Poisons are crafted using SGD for a maximum of 5,000 iterations with initial learning rate of 0.001, which decays by a factor of 0.95 every 2,000 iterations with $\epsilon = 16/255$. The target image is patched with an 8×8 patch in the bottom right corner. Then we fine-tune the last linear layer of the network using SGD for 20 epochs with initial learning rate of 0.5, which decays by a factor of 0.1 after epochs 5, 10, and 15.

Table 4: Baseline performance

Attack	Success Rate (%)
FC	92.00 ± 2.71
CP	88.00 ± 3.25
CLBD	86.00 ± 3.47
HTBD	59.00 ± 4.92

A.3 Training without SGD or data augmentation

We add data normalization and augmentation to the pre-training processes in each attack. For FC and CP, which were originally tested with ADAM, we show results from experiments where normalization and augmentation are used with ADAM as well as when we pre-train with SGD.

Table 5: Data normalization and augmentation + ADAM

Attack	Success Rate (%)	Diff. From Baseline (%)
FC	42.00 ± 4.94	-50.00
CP	37.00 ± 4.83	-51.00

Table 6: Data normalization and augmentation + SGD

Attack	Success Rate (%)	Diff. From Baseline (%)
FC	51.00 ± 5.00	-41.00
CP	12.00 ± 3.25	-76.00
CLBD	1.00 ± 5.00	-85.00
HTBD	11.00 ± 3.13	-48.00

A.4 Victim architecture matters

We test each method on ResNet-18 victims. Note that CP shows no change from the baseline, as our baseline set-up for CP uses a ResNet-18 victim model.

Table 7: ResNet-18 victims

Attack	Success Rate (%)	Diff. From Baseline (%)
FC	4.00 ± 5.00	-88.00
CP	88.00 ± 3.25	0.00
CLBD	80.00 ± 4.00	-6.00
HTBD	18.00 ± 3.84	-41.00

A.5 "Clean" attacks are sometimes dirty

We test each attack with an ℓ_∞-norm constrained perturbation with $\varepsilon = {}^{8}\!/_{255}$. Note that HTBD shows no change form the baseline since this was the ε values used in our baseline for this attack. See Table 8.

A.6 Properly transfer learned models are vulnerable

We use feature extractors pre-trained to classify CIFAR-100 data to craft the poisons. Then, we use those same feature extractors in the fine-tuning stage when we train the models to classify CIFAR-10 data. See Table 9.

Table 8: Poisons crafted with $\varepsilon = {}^{8}/_{255}$

Attack	Success Rate (%)	Diff. From Baseline (%)
FC	7.00 ± 2.55	-85.00
CP	80.00 ± 4.00	-8.00
CLBD	10.00 ± 3.00	-76.00
HTBD	56.00 ± 4.96	-3.00

Table 9: Transfer learned victims

Attack	Success Rate (%)	Diff. From Baseline (%)
FC	16.00 ± 3.67	-76.00
CP	100.00 ± 5.00	$+12.00$
CLBD	2.00 ± 5.00	-84.00
HTBD	25.00 ± 4.33	-34.00

A.7 Performance is not invariant to dataset size

We study the effect of scaling the dataset size while holding the percentage of data that is poisoned constant. We test each attack with 5 poisons and 500 training images and increment both the poison budget and the training set size until we reach 500 poisons and 50,000 training images (the entire CIFAR-10 training set). For every training set size, we allow the attacker to perturb 1% of the data and we see that the strength of poisoning attacks does not scale with any generality – in some cases we see success rates drop with increase in dataset size, and some attacks are more successful with more data. See Table 10 for complete numerical results with confidence intervals of width one standard error. This experiment, whose results are perhaps best presented in Figure 2, shows that discussing the poison budget only as a percentage of the data does not allow for fair comparison.

Table 10: Success rates (%) with varying dataset sizes and number of poisons

	Number of Poisons				
Attack	5	10	25	50	100
FC	46.30 ± 6.79	47.06 ± 6.99	66.67 ± 6.60	62.75 ± 6.77	72.00 ± 6.35
CP	100.00 ± 7.07	100.00 ± 7.07	98.00 ± 7.07	88.00 ± 4.60	82.00 ± 5.43
CLBD	5.88 ± 7.00	4.08 ± 7.14	2.00 ± 7.07	9.80 ± 7.00	3.92 ± 7.00
HTBD	26.00 ± 4.39	35.00 ± 4.77	43.00 ± 4.95	50.00 ± 5.00	55.00 ± 4.97

	Number of Poisons			
Attack	200	300	400	500
FC	76.00 ± 6.04	71.70 ± 6.19	80.39 ± 5.56	78.00 ± 5.86
CP	76.00 ± 6.04	55.10 ± 7.11	42.86 ± 7.07	26.00 ± 6.20
CLBD	4.00 ± 7.07	30.00 ± 6.48	68.00 ± 6.60	92.00 ± 7.07
HTBD	56.00 ± 4.96	59.00 ± 4.92	62.00 ± 4.85	53.00 ± 4.99

A.8 Black-box performance is low

When tested in the black-box setting, all methods except for CLBD show dramatically lower performance. CLBD is intended for use in the training from scratch case, so the baseline is black-box. For FC, CP, and HTBD we craft poisons on the architectures used in the baselines. The black-box victims for FC and HTBD are ResNet-18 models, whereas the CP baseline used a ResNet-18 victim, so we use a MobileNetV2 for the black-box victim. See Table 11.

Table 11: Black-box victim

Attack	Success Rate (%)	Diff. From Baseline (%)
FC	4.00 ± 5.00	-88.00
CP	16.00 ± 3.67	-72.00
CLBD	86.00 ± 3.47	0.00
HTBD	2.00 ± 5.00	-57.00

A.9 Small sample sizes and non-random targets

We test FC and CP with the specific target/base class pairs studied in the original works. We find the performance of each attack measured only on these classes differs from our baseline. See Table 12. This fact alone is sufficient evidence that the comparisons done in the poison literature are lacking consistency, and that this field needs a benchmark problem.

Table 12: Success with specific class pairs

Attack	Target	Base	Success Rate (%)	Diff. From Baseline (%)
FC	plane	frog	80.00 ± 4.00	-12.00
CP	frog	ship	83.00 ± 3.76	-5.00

A.10 Attacks are highly specific to the target image

We consider the case where the target object is photographed in a slightly different environment than in the particular image the attacker uses while crafting poisons. Perhaps, the attacker is trying to keep their own red car from being classified as a car. In reality, the deployed model may see a different image than the specific photograph to which the attacker has access. We are unable to get new photographs of the exact objects in CIFAR-10 images, so we choose to upper bound performance on highly modified images by simply flipping the target images horizontally during evaluation. In this setting, we observe that triggerless attacks are severely impaired, supporting our conclusion that they pose less practical threat in physical settings than suggested in previous work. See Table 13.

Table 13: Success on flipped targets

Attack	Success Rate (%)	Diff. From Baseline (%)
FC	7.00 ± 2.55	-85.00
CP	40.00 ± 4.90	-48.00

A.11 Ensemblizing boosts the attacker

We study the impact of ensemblizing attacks, where the attacker uses several architectures while crafting poisons. This was suggested and tested with CP in the original work [29]. In that study however, the comparison between CP and FC is incomplete. We show that ensemblizing helps both attacks and that FC outperforms CP in the white-box setting with enough poisons (both in the single model and the ensemblized attacks). See Tables 19, 20, and 22.

A.12 Backdoor success depends on patch size

In order to determine the effect of the particular size of the patch used in backdoor attacks, we test the backdoor methods with a variety of patch sizes. We see a strong correlation between patch size and success rate. See Table 14. Note that dashes correspond to an attack's baseline performance, see Table 4.

Table 14: Success rates (%) of backdoor attacks with varying patch sizes

	Patch Size		
Attack	3×3	5×5	8×8
HTBD	20.00 ± 4.0	33.00 ± 4.70	-
CLBD	-	97.00 ± 5.00	100 ± 5.0

A.13 Model training and performances

Models trained for our experiments In Tables 15 and 16, we show the training setups, including references to sets of hyperparameters outlined in Table 3, and the training and testing accuracy of the models we use in this study. Each row in these two tables shows averages of ten models we trained from random intializations with identical training setups. Note that the models called "AlexNet" are the variants introduced in the original FC paper, see that work for details [24]. Models named "HTBD AlexNet" are the modified AlexNet architecture we used for the HTBD experiments and the details are below.

Architectures we use Four of the five architectures we use in this study are widely used and/or detailed in other works. For the modified AlexNet used in FC experiments, see [24]. For ResNet-18 architecture details, see [6]. For MobileNetV2, see [23]. For VGG11, see [25].

HTBD AlexNet The model used in the original HTBD work was a simplified version of AlexNet. But for our baseline experiments we adapt the ImageNet AlexNet model to CIFAR-10 dataset. We modify the kernel size and stride in the first convolution layer to 3 and 2, respectively, in order to take $32 \times 32 \times 3$ input images. For deeper layers we use a stride of 1. The width of the network at then end of the convolutional layers is 256.

Table 15: CIFAR-10 models

Model	Norm.	Aug.	Hyperparam.	Train Acc. (%)	Test Acc. (%)
AlexNet	×	×	A	99.99	73.96
AlexNet	✓	×	A	99.99	74.45
AlexNet	✓	✓	A	90.35	82.36
AlexNet	✓	✓	B	98.77	85.91
HTBD AlexNet	✓	×	B	100.00	77.35
HTBD AlexNet	✓	✓	B	98.80	84.30
ResNet-18	×	×	C	100.00	87.05
ResNet-18	✓	×	C	100.00	87.10
ResNet-18	✓	✓	C	99.99	94.96
MobileNetV2	×	×	C	99.99	82.11
MobileNetV2	✓	×	C	99.99	82.04
MobileNetV2	✓	✓	C	99.88	93.36

Table 16: CIFAR-100 models

Model	Norm.	Aug.	Hyperparam.	Train Acc. (%)	Test Acc. (%)
ResNet-18	✓	✓	D	99.97	74.37
MobileNetV2	✓	✓	D	99.95	71.81
VGG11	✓	✓	D	99.97	67.87

Transfer learning We also train models of each architecture from scratch on the first 250 images per class of CIFAR-10. By comparing these models to transfer learned models on the same data, we see the benefit of transfer learning for this quantity of data. Each row of Table 17 shows averages of 10 models. For the transfer learned models we use exactly the feature extractors from the benchmark, and the pre-trained models' performances are in Table 16. It is clear from Table 17 that with so little data, training from scratch leads to less-than-optimal test accuracy. This motivates the transfer learning benchmark tests since transfer learning does improve performance in this setting.

Table 17: CIFAR-10 models with Trainset size 2500

	Model	Norm.	Aug.	Hyperparam.	Train Acc. (%)	Test Acc. (%)
From Scratch	ResNet-18	✓	✓	F	99.12	59.71
	MobileNetV2	✓	✓	F	98.99	68.55
	VGG11	✓	✓	C	97.98	60.72
Transfer learned	ResNet-18	✓	✓	G	99.98	80.38
	MobileNetV2	✓	✓	G	99.97	79.53
	VGG11	✓	✓	G	99.93	74.95

A.14 Additional experiments

Backdoor triggers Different backdoor attacks use different patch images as a trigger. Does the performance of the attack depends on the patch used? In order to study the impact, we swap the patches of CLBD and HTBD. We resize the CLBD patch to 8×8 and the HTBD patch to 3×3 which are the baseline sizes, see Figure 4. We observe that patch image does matter and it can have a significant effect on the performance of an attack. See Table 18.

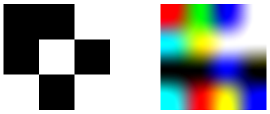

Figure 4: CLBD patch (left), and HTBD patch (right).

Table 18: Success rates (%) of backdoor attacks with swapped patch images

Attack	Success Rate (%)	Diff. From Baseline (%)
HTBD w/ CLBD patch	51.00 ± 4.99	-8.00
CLBD w/ HTBD patch	2.00 ± 5.00	-84.00

Poison budget We conduct an additional experiment to assess the impact of the budget in our benchmark. We test each attack in the same setting at the benchmark, where we do 100 trials each with 50, 100, and 250 poisons. See Tables 19 and 20. We see the expected rise in success rate with increased budget, however we note that these increases are almost always small. We choose to use 25 poisons in the benchmark tests for the following two reasons. First, we want the evaluation of an attack to be accessible to those with modest computing resources. Second, as discussed in Appendix A.15 we find 25 images out of 250 images per class to be a large budget in realistic settings.

A.15 How many images

When scraping data from Google, the sources of images are diverse. If each source is responsible for very few images, then an adversary may have a difficult time poisoning a significant amount of data scraped by their victim. In order to investigate the diversity of sources, we query Google Images for each CIFAR-10 class label and measure how many images come from each source. Table 21 shows how many images the first through fifth most represented source are each responsible for in the first 100 search results for each class. Entries represent the number of images in a particular class coming from a particular source within the first 100 search results for that query. The first column represents the source responsible for the most images. The second column represents the source responsible for the second most images, etc. We find that sources generally are not highly dominant, and each source is responsible for few images. Poisoning methods which only perturb data from the target class may only be able to poison a very small percentage of the victim's total data, especially when the number of classes is high. For example, in a 1000-class problem like ImageNet, even if the attacker could

Table 19: Frozen feature extractor tests with varying budget

		Success Rates (%)		
Attack	Budget	White-box	Grey-box	Black-box
FC	50	23.0 ± 4.21	8.0 ± 2.71	7.0 ± 1.80
	100	59.0 ± 4.92	10.0 ± 3.03	5.0 ± 1.56
	250	93.0 ± 2.55	27.0 ± 4.44	17.0 ± 3.76
FC-Ens.	50	17.0 ± 3.76	8.0 ± 2.71	19.5 ± 2.80
	100	40.0 ± 4.90	15.0 ± 2.55	33.0 ± 3.34
	250	79.0 ± 4.07	38.00 ± 4.85	69.0 ± 4.62
CP	50	24.0 ± 4.27	7.0 ± 2.55	8.0 ± 1.92
	100	38.0 ± 4.85	8.0 ± 2.71	2.5 ± 5.00
	250	49.0 ± 5.00	9.0 ± 2.86	5.0 ± 1.54
CP-Ens.	50	22.0 ± 4.14	7.0 ± 2.55	22.0 ± 2.93
	100	33.0 ± 4.70	12.0 ± 3.25	28.0 ± 3.17
	250	47.0 ± 4.99	18.0 ± 3.84	37.0 ± 4.83
CLBD	50	5.0 ± 5.00	2.0 ± 5.00	3.0 ± 1.20
	100	2.0 ± 5.00	2.0 ± 5.00	2.0 ± 5.00
	250	1.0 ± 5.00	2.0 ± 5.00	2.0 ± 5.00
HTBD	50	7.0 ± 2.55	7.0 ± 2.55	7.0 ± 1.80
	100	2.0 ± 5.00	5.0 ± 5.00	6.5 ± 1.74
	250	10.0 ± 3.00	4.0 ± 5.00	6.0 ± 2.37

Table 20: End-to-end tests with varying budget

		Success Rates (%)		
Attack	Budget	White-box	Grey-box	Black-box
FC	50	9.0 ± 2.86	6.0 ± 2.37	4.5 ± 1.47
	100	12.0 ± 3.25	8.0 ± 2.74	5.0 ± 1.55
	250	32.0 ± 4.66	11.0 ± 3.13	13.59 ± 3.38
FC-Ens.	50	10.0 ± 3.00	5.0 ± 5.00	8.0 ± 1.92
	100	9.0 ± 2.86	4.0 ± 5.00	6.5 ± 1.74
	250	21.0 ± 4.07	13.00 ± 3.36	19.00 ± 3.92
CP	50	14.0 ± 3.47	6.0 ± 2.37	3.5 ± 1.30
	100	19.0 ± 3.92	7.0 ± 2.55	4.0 ± 1.39
	250	15.0 ± 3.57	2.0 ± 5.00	0.0 ± 5.00
CP-Ens.	50	17.0 ± 3.76	8.0 ± 2.71	10.0 ± 2.12
	100	15.0 ± 3.57	8.0 ± 2.71	11.0 ± 2.21
	250	17.0 ± 3.76	6.0 ± 2.37	15.0 ± 3.57
CLBD	50	4.0 ± 5.00	2.0 ± 5.00	3.0 ± 1.20
	100	1.0 ± 5.00	0.0 ± 5.00	2.0 ± 5.00
	250	0.0 ± 5.00	0.0 ± 5.00	2.0 ± 5.00
HTBD	50	3.0 ± 5.00	2.0 ± 5.00	4.0 ± 1.39
	100	3.0 ± 5.00	3.0 ± 5.00	5.0 ± 1.54
	250	0.0 ± 5.00	0.0 ± 5.00	0.0 ± 5.00

poison 10% of the target class, this would only represent 0.01% of the total dataset. This percentage is far smaller than the percentages studied in the data poisoning literature.

Table 21: Google Images case study

	Number of Images				
Search term	Source 1	Source 2	Source 3	Source 4	Source 5
airplane	7	5	5	3	3
automobile	9	7	6	6	3
bird	7	5	4	4	4
cat	8	8	5	4	4
deer	6	6	5	4	4
dog	14	9	6	4	3
frog	5	4	4	4	3
horse	9	5	4	3	3
ship	9	6	5	5	4
truck	9	6	6	5	4

A.16 Benchmark results

We present complete benchmark results with confidence intervals in Table 22.

Table 22: Complete benchmark results

	Frozen Feature Extractor		
Attack	WB (%)	GB (%)	BB (%)
FC	16.0 ± 3.67	7.0 ± 2.55	3.5 ± 1.30
FC-E	13.0 ± 3.36	9.0 ± 2.86	6.0 ± 1.68
CP	24.0 ± 4.27	7.0 ± 2.55	4.5 ± 1.47
CP-E	20.0 ± 4.00	8.0 ± 2.71	12.5 ± 2.34
CLBD	3.0 ± 5.00	6.0 ± 2.37	3.5 ± 1.30
HTBD	2.0 ± 5.00	4.0 ± 5.00	4.0 ± 1.39

	End-To-End Fine Tuning		
Attack	WB (%)	GB (%)	BB (%)
FC	4.0 ± 5.00	3.0 ± 5.00	3.5 ± 1.30
FC-E	7.0 ± 2.55	4.0 ± 5.00	5.0 ± 1.54
CP	17.0 ± 3.76	7.0 ± 2.55	4.5 ± 1.47
CP-E	14.0 ± 3.47	4.0 ± 5.00	10.5 ± 2.17
CLBD	3.0 ± 5.00	2.0 ± 5.00	1.5 ± 5.00
HTBD	3.0 ± 5.00	2.0 ± 5.00	4.0 ± 1.39

		Training From Scratch		
Attack	ResNet-18 (%)	MobileNetV2 (%)	VGG11 (%)	Average (%)
FC	0.0 ± 5.00	1.0 ± 5.00	3.0 ± 5.00	1.3 ± 5.00
CP	0.0 ± 5.00	1.0 ± 5.00	1.0 ± 5.00	0.7 ± 5.00
CLBD	0.0 ± 5.00	1.0 ± 5.00	2.0 ± 5.00	1.0 ± 5.00
HTBD	0.0 ± 5.00	4.0 ± 5.00	1.0 ± 5.00	2.7 ± 5.00

A.17 Run times and hardware

We measure the run time of our benchmark tests. We note that the amount of time required to craft poisons is highly attack/algorithm dependent. However, once a batch of poisons has been crafted, we measure the run times of our benchmark tests. These tests each comprise some number of training epochs on CIFAR-10 data. (In the case of the FFE and E2E tests, the data is only 2,500 images, whereas the FST test uses all 50,000 images in the CIFAR-10 trainset.) In Table 23 we report the average wall time to compute 100 trials of each test using Nvidia GeForce RTX 2080 Ti GPUs.

Table 23: Benchmark test run times

Benchmark	Run time (hrs)
FFE	4.54
E2E	7.35
FST	455.12