What Doesn't Kill You Makes You Robust(er): Adversarial Training against Poisons and Backdoors

Jonas Geiping [1] **Liam Fowl** [2] **Gowthami Somepalli** [2] **Micah Goldblum** [2] **Michael Moeller** [1] **Tom Goldstein** [2]

Abstract

Data poisoning is a threat model in which a malicious actor tampers with training data to manipulate outcomes at inference time. A variety of defenses against this threat model have been proposed, but each suffers from at least one of the following flaws: they are easily overcome by adaptive attacks, they severely reduce testing performance, or they cannot generalize to diverse data poisoning threat models. Adversarial training, and its variants, is currently considered the only empirically strong defense against (inference-time) adversarial attacks. In this work, we extend the adversarial training framework to instead defend against (training-time) poisoning and backdoor attacks. Our method desensitizes networks to the effects of poisoning by creating poisons during training and injecting them into training batches. We show that this defense withstands adaptive attacks, generalizes to diverse threat models, and incurs a better performance trade-off than previous defenses.

1. Introduction

As machine learning systems consume more and more data, the data curation process is increasingly automated and reliant on data from untrusted sources. Breakthroughs in image classification (Russakovsky et al., 2015) as well as text processing (Brown et al., 2020) are built on large corpora of data *scraped* from the internet. Automated scraping, in which data is collected directly from online sources, leaves practitioners vulnerable to *data poisoning* in which bad actors tamper with the data so that models trained on this data perform poorly or contain *backdoors* embedded in them (Gu et al., 2019; Shafahi et al., 2018). These attacks present security vulnerabilities that persist even if the data is labeled and checked by crowd-sourced human supervision. In essence, entire machine learning pipelines can be compromised if the input data is modified maliciously - even if the modification appears minor and inconspicuous to a human observer. This mounting threat has instilled fear especially in industry practitioners whose business models rely on powerful neural networks trained on massive volumes of scraped data (Kumar et al., 2020).

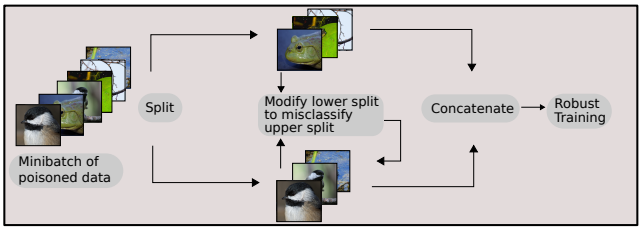


Figure 1. Data poisoning attacks require a new approach to adversarial training to robustify machine learning models against this threat model.

In response to this growing threat, recent works have proposed a number of defenses against data poisoning attacks (Paudice et al., 2018; Ma et al., 2019). Existing defense strategies suffer from up to three primary shortcomings:

1. In exchange for robustness, they trade off test accuracy to a degree that is intolerable to real-world practitioners (Geiping et al., 2021).

2. They are only robust to specific threat models but not to adaptive attacks specially designed to circumvent the defense (Koh et al., 2018; Tan & Shokri, 2020).

3. They apply only to a specific threat model and do not lend a generally applicable framework to practitioners (Wang et al., 2019).

We instead propose a variant of *adversarial training* that harnesses adversarially poisoned data in the place of (test-time) adversarial examples. We show that this strategy exhibits both an improved robustness-accuracy trade-off as well as greater flexibility for defending against a wide range of threats including adaptive attacks.

Adversarial training desensitizes neural networks to test-

[1]Department of Electrical Engineering and Computer Science, University of Siegen; Germany [2]Department of Computer Science, University of Maryland, College Park; United States.

time adversarial perturbations by augmenting the training data with on-the-fly crafted adversarial examples (Madry et al., 2018). Similarly, we modify training data in order to desensitize neural networks to the types of perturbations caused by data poisoning - yet adapting this robust training framework to data poisoning requires special consideration of this new threat model. For example, we must decide how to select targets during training in order to simulate targeted data poisoning. We demonstrate the effectiveness of this framework at defending against a range of data tampering threat models including both targeted data poisoning and backdoor trigger attacks on both *from-scratch* training *transfer learning*, where large models pre-trained on massive datasets are fine-tuned on a small amount of data which may be poisoned. We visualize the impact of the defense in feature space and compare to a range of related defense strategies.

2. Related Work

Data poisoning is a class of threat scenarios focused on malicious modifications to the training data of a machine learning model. See Goldblum et al. (2020) for an overview of dataset security. Data poisoning attacks can either focus on denial-of-service attacks on model *availability* that reduce overall model performance or on backdoor attacks that introduce malicious behavior into an otherwise inconspicuous model which is triggered by a specific visual pattern or target image, thus breaking model *integrity* (Barreno et al., 2010).

In this work, we focus on attacks against model integrity. In comparison to denial-of-service attacks, which can be noticed before deployment, integrity attacks can insert undetectable backdoors even into models that later pass into production and are used and relied upon in real-world scenarios. These attacks can be further distinguished by the nature of their trigger mechanism. In *backdoor trigger attacks* (Gu et al., 2019; Turner et al., 2018), the attack is triggered by a specific backdoor pattern or patch that can be added to target images at test time, whereas *targeted data poisoning* (Shafahi et al., 2018; Zhu et al., 2019) is triggered by a predefined target image. In contrast to targeted poisoning, backdoor trigger attacks can be applied to multiple target images but require target modifications to be active during inference, while targeted attacks are activated by specific, but unmodified targets.

Attacks can be further distinguished by the precise training setup they anticipate their victims to employ. Some attacks assume that the victim will only fine-tune their model on the poisoned data or will train a linear classifier on top of a pre-trained feature extractor (Saha et al., 2020; Zhu et al., 2019; Shafahi et al., 2018). These methods are effective against practitioners engaging in transfer learning with pre-trained models like those found in popular repositories such as Paszke et al. (2017). Other attacks work even when the victim trains their model from scratch on the poisoned data (Huang et al., 2020; Geiping et al., 2021). Generally, the simpler the victim's training procedure, the more readily the attacker can anticipate the effect of their perturbations and thus, the easier the attacker's job.

Here, we briefly detail prominent attacks against which we test our defense:

Feature Collision: Shafahi et al. (2018) present a targeted poisoning attack that perturbs training data so that its deep features collide with those corresponding to a target image. At the same time, the attack penalizes the size of perturbations applied to training data in order to maintain visual similarity between the original and perturbed images. Aghakhani et al. (2020), as opposed to colliding images in feature space, surrounds target images in feature space to increase the success of feature collision. This method improves on the work of Zhu et al. (2019), which constructs a convex polytope around the target based on an ensemble of models.

Bilevel Optimization: MetaPoison (Huang et al., 2020) generates poisoned data based on unrolling the true bilevel objective encountered in targeted data poisoning for several steps and optimizing the unrolled objective over an ensemble of models at different stages in training, leading to an attack that especially robust to new initializations in from-scratch training as well as to changes in model architectures.

Gradient Matching: Witches' Brew (Geiping et al., 2021) instead approximates the bilevel objective by gradient matching, leading to an efficient attack in comparison to MetaPoison (Huang et al., 2020), combining the efficiency of feature collision attacks with the success of bilevel approximations. This attack was already adapted to be effective against data augmentation and differential privacy, showing the need for strong defenses against adaptive attacks.

Patch attacks: Gu et al. (2019) trigger backdoors by adding an ℓ_0 bounded patch to training images in a given class, causing a network trained on these images to associate the patch with the given class. Then, the attacker patches test-time images (of a different class) with the same patch in the hopes that the network mis-classifies the images.

Hidden Trigger Backdoor: Saha et al. (2020) present a backdoor trigger attack wherein the attacker subtly modifies training data to increase the effectiveness of a patch added to target validation images. In this way the attack can be considered a hybrid between targeted data poisoning and backdoor attacks.

Defenses against data poisoning can be broadly classified

into *filter defenses* which attempt to detect and remove or sanitize malicious training data, *robust training* algorithms which use a training routine that yields robust models even on malicious training data, and *model repair* methods that train models on poisoned data and attempt to repair the poisoned models after training. Filter defenses are easy to deploy as they simply add a pre-processing step, but they require extensive hyperparameter tuning and rely on the assumption that only a small fraction of the dataset is poisoned. Furthermore, filter defenses can often be overcome by adaptive attacks (Koh et al., 2018; Tan & Shokri, 2020), mirroring the phenomenon of broken defenses against test-time adversarial attacks (Carlini & Wagner, 2017). Each approach reduces model performance (in terms of validation accuracy); filter defenses reduce performance as a result of training the new model on fewer samples after removing data, robust training methods do so by deviating from standard training practices which are tuned for accuracy in order to increase robustness, and model repair methods harm accuracy by pruning away potentially important neurons.

Numerous options have been proposed for detecting poisoned data. Tran et al. (2018) detect *spectral signatures* associated with backdoor triggers based on their correlation with the top right singular vector of the covariance matrix formed by feature representations, and additional detection scores can be found in Paudice et al. (2018). Peri et al. (2020) detect poisoned data by clustering based on *deep KNN*, re-labeling data based on the nearest neighbors in feature space. Chen et al. (2019) cluster training data based on activation patterns in feature space. Yet another method detects images containing backdoor triggers by flagging images whose corresponding predictions do not change when they are combined with clean samples (Gao et al., 2019). Any measure of "anomaly" that is used to filter images can also be used to generate poisoned data which minimizes the anomalous property, thus adaptively defeating the defense (Koh et al., 2018).

Robust training algorithms may incorporate strong data augmentations (Borgnia et al., 2020), randomized smoothing (Weber et al., 2020), or may partition data into disjoint pieces and train individual models on the partitions, performing classification via majority voting at test-time (Levine & Feizi, 2021). Another popular robust training strategy harnesses differentially private SGD (Abadi et al., 2016; Ma et al., 2019), as differentially private models are inherently insensitive to small changes on a subset of the training set. Differentially private SGD is applied by clipping and adding noise to the model parameter gradients during training. Hong et al. (2020) note that the addition of noise is the primary factor controlling robustness to poisoning. However, differential privacy is an extreme and general definition of robustness to data manipulations, compared to robustness specific to data poisoning. This strategy consequently incurs a significant performance penalty (Jayaraman & Evans, 2019), and these algorithms can even be adaptively attacked by modifying gradient signals during poison generation in the same manner as in the defense.

Model repair strategies, primarily designed to defend against backdoor attacks, may reconstruct the backdoor trigger and nullify its effects (Wang et al., 2019) or prune away neurons which are inactive on clean samples (Liu et al., 2018). However, adaptive attacks can bypass these defenses by creating poisoned data whose activation patterns mimic those of clean data (Veldanda & Garg, 2020). In order to counteract the loss of performance induced by pruning, some methods fine-tune the pruned model on clean data (Liu et al., 2018; Chen et al., 2020). But this process is only effective when the defender possesses large quantities of trustworthy clean data.

3. Generalizing Adversarial Training to Data Poisoning

Adversarial training (Madry et al., 2018; Sinha et al., 2018) reduces the impact of test-time adversarial attacks and is generally considered the only strong defense against adversarial examples. Adversarial training solves the saddle-point problem,

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathbb{D}} \left[\max_{\Delta \in S} \mathcal{L}_\theta(x+\Delta, y) \right], \tag{1}$$

where $\mathcal{L}_\theta$ denotes the loss function of a model with parameters θ, and the adversary perturbs inputs x from a data distribution $\mathbb{D}$, subject to the constraint that perturbation Δ is in S. Peri et al. (2020) notes that adversarial training against test-time evasion attacks already confers a small degree of robustness against data poisoning at a performance cost. Our proposed strategy is an adaptation of adversarial training to poisoning, resulting in a stronger defense that degrades performance less than differentially private SGD or adversarial training against evasion attacks. In our adversarial training paradigm, two parties engage in a mini-max game; the attacker maliciously poisons the training data to cause the model to mis-classify targets, while the defender trains the model to correctly classify both poisons and targets. As we describe in the previous section, the capabilities of an attacker depend on its knowledge of the defender's training setup, so we now enumerate a series of assumptions concerning the knowledge of the attacker and defender before presenting our framework in precise detail.

Preparing for a strong threat model. In order to harden the model against a wide range of poisons, we train against a strong surrogate attacker. The differences between the surrogate threat model and that of a real-world attacker concern the attacker's access to the defender's training routine. The

surrogate attacker in our training algorithm is aware of the defender's training protocol (e.g. learning rate, optimization algorithm), architecture, and defense strategy but can neither influence training nor intercept random factors such as initialization and mini-batch sampling. In cases where the defender only re-trains a component of a model or fine-tunes the model, the exact baseline pre-trained model, including its parameters, is known to both parties. The attacker's trigger (a target image for targeted poisoning or a specific patch for backdoor attacks) is unknown to the defender, and we do not assume that the defender possesses additional, vetted, clean data. In order to constrain the attacker, the defender chooses a $||\cdot||_p$-norm perturbation budget ε against which they seek robustness.

Since the attacker possesses such strong knowledge concerning the defender's training routine, the threat it poses constitutes a near worst-case analysis. If more factors, such as model definition or parts of the training protocol, are hidden from the attacker, then the quality of the defense can only improve. On the other hand, the defender needs to set a ε bound within which to be robust against attacks - if there were no such limit, then the attacker could arbitrarily modify data. Previous works such as Xu et al. (2020) have considered the existence of clean data or a given clean model, but this presents the obvious problem that if the security of the data pipeline is breached, then there may be no way of knowing a-priori what data is clean and what is not.

3.1. Adversarial Poisoning

Conceptually poisoning attacks differ from evasion attacks through their intermediacy; the attacker modifies some sample x_p of the data distribution $\mathbb{D}$ within constraints S, to change model behavior when evaluated on another sample x_t. As such, the defender needs to train to be invariant to any such modifications. Formally, *adversarial poisoning* thus requires approximating a robust estimation objective given by

$$\begin{aligned} &\min_{\theta} \mathbb{E}_{\substack{(x_p,y_p)\sim\mathbb{D} \\ (x_t,y_t)\sim\mathbb{D}}} \left[\mathcal{L}_\theta(x_p+\Delta_p, y_p) + \mathcal{L}_\theta(x_t+\Delta_t, y_t)\right] \\ &\text{s.t.} \quad \Delta_p, \Delta_t \in \underset{\Delta_p,\Delta_t\in S}{\arg\min} \mathcal{L}_{\text{adv}}(x_p+\Delta_p, x_t+\Delta_t, \theta), \end{aligned} \tag{2}$$

where $\mathcal{L}_\theta$ denotes the loss of a model with parameters θ, and $\mathcal{L}_{\text{adv}}$ denotes the objective function of an arbitrary data poisoning attack. x_p simulates training data (to be poisoned with Δ_p) and x_t simulates a target with possible trigger Δ_t. For example, in gradient matching, Δ_t is zero since such an attack does not modify targets, and $\Delta_p \in \arg\min_{\Delta\in S} \text{sim}\left(\nabla_\theta \mathcal{L}_\theta(x_p+\Delta, y_p), \nabla_\theta \mathcal{L}_\theta(x_t, y_p)\right)$ minimizes the cosine similarity, while for simple backdoor triggers $\Delta_p = \Delta_t$ are non-optimized randomly drawn patches. This robust estimation problem is a strict generalization of adversarial training, which we can recover via $\mathcal{L}_{\text{adv}} = -\sum_{i\in\{p,t\}} \mathcal{L}_\theta(x_i+\Delta_i, y_i)$. Data poisoning attacks such as backdoor triggers, feature collision and gradient alignment achieve success by perturbing $x_p+\Delta_p$ to correlate with $x_t+\Delta_t$ in some metric, a correlation that is then learned by the network, but the robustness objective minimizes the effect of small malicious correlations.

Algorithm 1 Modified iterative training routine.

Input: Split probability $p \in (0,1)$.
repeat
  Sample mini-batch of data $\{x_i, y_i\}_{i=1}^n$,
  Split data randomly into two subsets x_p, x_t with probability p
  Draw malicious labels y_t for x_t
  Apply a data poisoning attack modifying $x_p+\Delta_p$ to reclassify $x_t+\Delta_t$ as y_t
  Concatenate x_p, x_t into a new batch x_m
  Update model based on new data x_m
until training finished

To realize an approximation to this objective, given a mini-batch of data, we first split this batch into two subsets of data, (x_p, y_p) and (x_p, y_p), at random with probability p for an image to be placed in the poison partition, we then run a chosen data poisoning attack with x_p, x_t, and then train the model on the concatenated output, as seen in fig. 1. This way we alternate between both steps in eq. (2) effectively. The full algorithm is summarized in algorithm 1.

Other defenses can be viewed as special cases of adversarial poisoning. This methodology generalizes and explains previous work on defenses against poisoning. In Borgnia et al. (2020), strong data augmentations such as mixup (Zhang et al., 2018) and cutout (DeVries & Taylor, 2017) are proposed as defenses against data poisoning. These defenses are special cases of the proposed adversarial poisoning; when algorithm 1 is used to defend against a watermarking attack (which superimposes the target data onto poison data with low opacity), then the attack is equivalent to mixup data augmentation with mixing factor $\alpha = 1-\varepsilon$. Likewise, implementing algorithm 1 against a backdoor trigger attack reduces to patching randomly selected pairs of data points with a random patch. If this patch is chosen to be uniformly gray, then this defense is exactly equivalent to the cutout data augmentation.

Example: Defending against Gradient Matching. While our methodology can be applied to any data poisoning attack, there are several considerations to make when adapting the attack into a format that is applicable and practical to run in each mini-batch of training. We detail these considerations for a recent attack (Geiping et al., 2021).

The cosine similarity objective of the attack is originally evaluated on a clean surrogate model trained by the attacker and the attack is optimized for a significant number of iterations ($n = 250$ in the original work). To apply it during training, we first replace the clean model used in the attack with the current model in the current state of training - this is actually an advantage for the defender. While the attacker needs to create poisons on a surrogate model, the defender can use the exact model, making it easier to create effective poisons. Secondly, similar to adversarial training, the number of attack iterations can be reduced. In practice, we choose $n = 5$ during the defense, as a compromise between creating a strong attack and spending a limited time budget, as the attack is naturally stronger due to its basis in the current state of training. Third, we need to choose malicious labels y_t. These labels could be chosen entirely at random, however then the average gradient over all target instances would likely be small. However, poisoned data points in Geiping et al. (2021) are in practice chosen from the same class as the target adversarial label, and this choice can be replicated for the randomly chosen subset of poisoned data points x_p with labels y_p by choosing y_t as the label that appears most often in y_p.

3.2. Adaptive Attack Scenarios

A crucial step in the design of new defense algorithms is their ability to withstand adaptive attacks, i.e. strong attacks that can be modified to respond to a novel defense algorithm when the attacker is aware of the defense. While this principle has been well-regarded in literature about adversarial attacks at test-time (Carlini et al., 2019; Tramer et al., 2020), it has not been applied as rigorously for data poisoning.

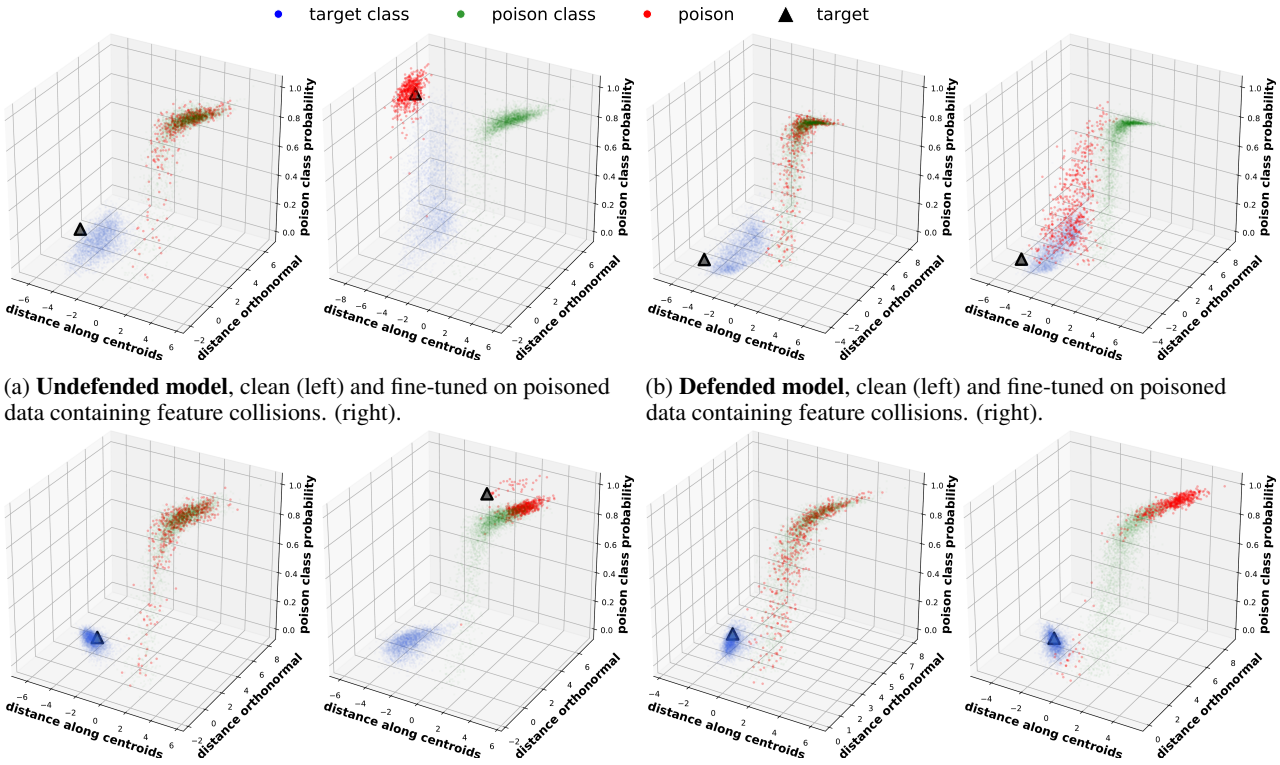
The defense proposed in this work is exceedingly effective against non-adaptive models (evaluating the exemplary case of gradient matching), as the difference in training regimes leads to incorrect perturbations computed by the attacker that relies on a pre-trained surrogate model. However, this would also be the case for most modifications to the training procedure, such as adding data augmentations or changing learning rates or optimizer settings. As such, we find that the optimal way to attack this defense is for the attacker to re-train their pre-trained model with exactly the same defense and the same hyperparameters. The attacker can then more accurately estimate the target gradient (for gradient matching) or target features (for feature collision). We also investigated the possibility of applying algorithm 1 during the optimization of poisoned data itself as an additional stochastic input modification. However, this modification weakens the attack by gradient masking, making it too difficult for the attacker to optimize the poisoned data. This behavior mirrors (test-time) adversarial attacks, where it is non-optimal to add additional perturbations during the creation of an adversarial perturbation. As we will find in the next section, the defense has a major impact on the feature space of a model, which may make it difficult to bypass the defense with other adaptive attacks.

4. Analysis

To understand the effect of the proposed adversarial poisoning scheme qualitatively, we conduct an analysis of feature space visualizations for several attacks. Shafahi et al. (2018), who introduced feature collision attacks, illustrate their poisoning method by visualizing the feature space collisions between poisoned data and the target. These experiments are carried out in the transfer setting, where the feature representation of the model is fixed and known to both parties. We run Bullseye Polytope, a recent and improved feature collision attack (Aghakhani et al., 2020), stronger than in Shafahi et al. (2018), in a strong attack setting of 500 poisoned examples for a ResNet-18 pre-trained on CIFAR-10 and re-trained on the poisoned dataset. We visualize the feature space by plotting the projections of feature vectors of data on to the vector connecting the centroids of the poison (base) class and the target class and it's orthogonal (generated using PCA) in the x-y plane and the softmax output for poison (base) class for each of the points on the z axis. This way we expect to see both classes to form separate clusters in feature space (x-y plane) and to be further separated by the poison class probability, which is low for images from the target class and high for images from the poison class.

Figure 2a shows the effects of an attack on a baseline model. The poisoned images (red) move to collide with the target (black triangle) in feature space as seen by their overlap in the x-y plane, while maintaining their original label (z-axis), subsequently leading to a misclassification of the target image as image from the poison class. Figure 2b however contrasts this collision with the effects of poisons on a defended model. Two effects stand out: First, the poisons, which were optimized to collide with the target, no longer cluster around the target (refer also to the 2D visualization in the supp. material), indicating that straightforward collisions are difficult to achieve against the robust model. Second, poisoned images close to the target are now predicted as the target base class shown by their descend on the z-axis; the defended model is robust enough to assign poisoned images a label that agrees with their feature representation, even though this assignment contradicts the given labels of these images. In essence, the poisoned images are treated like images from the target class but with a noisy label. This property completely reverses the attack. Instead of moving the target into the poison class, the poisoned images are drawn into the target class as it matches their feature representation. This way, the model stays consistent and is able to defend against the strong attack analyzed here.

In addition to feature collision attacks, in fig. 2, we analyze



(a) **Undefended model**, clean (left) and fine-tuned on poisoned data containing feature collisions. (right).

(b) **Defended model**, clean (left) and fine-tuned on poisoned data containing feature collisions. (right).

(c) **Undefended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

(d) **Defended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

Figure 2. Visualization of the effects of poisoning attacks against an undefended and a defended model. Top: Feature collisions via Aghakhani et al. (2020). Bottom: Gradient matching as in Geiping et al. (2021). The target image is marked by a black triangle and is originally part of the class colored blue. The poisoned images are colored red and are part of the class colored green. The x-y axis in each diagram corresponds to a projection of the principal direction separating both classes, while the confidence in the original target class is marked on the z-axis.

the defense against the gradient matching attack of Geiping et al. (2021) in the from-scratch setting, where the model is fully re-trained. The attack can be seen to be effective in fig. 2c, changing the decision boundary of the model to fit the target without collisions by clustering poisons opposite to the target in feature space, significantly moving the target. However, this is prevented by the defense as seen in fig. 2d. The robust model is not modified by the clustering of poisoned images, and outliers seen in the undefended model are again reclassified as the target class leading to a consistent decision. An interesting side effect of the defense for both attacked and clean models is that the model itself is generally less over-confident in its clean predictions. We compute similar outcomes also for other attacks, such as backdoor triggers, which we show in the supp. material.

5. Experiments

This section details a quantitative analysis of the proposed defense for the application of image classification with deep neural networks. To fairly evaluate all attacks and defenses, especially in light of Schwarzschild et al. (2020) discussing the difficulty in comparing attacks across different evaluation settings, we implement all attacks and defenses in a single unified framework, which we will make publicly available. For all experiments, we measure *avg. poison success* over 20 trials, where each trial represents a randomly-chosen attack trigger from a random class and a separately attacked and trained model. The sampling of randomized attack triggers is crucial to estimate the average performance of poisoning attacks, which are generally more effective for related class labels. We discuss additional experimental details in the supp. material.

5.1. Defending in Diverse Scenarios

To evaluate the proposed defense mechanism thoroughly, we consider a variety of attacks in different scenarios. We distinguish three scenarios with increased difficulty for the attacker, *transfer* where the defender only re-trains the last linear layer of a model, *fine-tuning*, where the defender re-trains all layers, and *from-scratch* where the defender trains

Table 1. Quantitative result for several attacks and their defense by adversarial poisoning with $p = 0.75$ for targeted poisoning and $p = 0.5$ for backdoor attacks, showing avg. poison success with standard error (where all trials have equal outcomes, we report the worst-case error estimate 5.59%). Additional details about each attack threat model can be found in the supp. material; Transfer* refers to the explicit setting of Zhu et al. (2019). The proposed defense significantly decreases success rates over a wide range of attacks and scenarios without any hyperparameter changes.

ATTACK	SCENARIO	UNDEFENDED SUCCESS	DEFENDED SUCCESS
BACKDOOR TRIGGERS	FROM-SCRATCH	87.38% (±2.24)	12.93% (±4.59)
METAPOISON	FROM-SCRATCH	64.00% (±11.76)	11.00% (±4.57))
GRADIENT MATCHING	FROM-SCRATCH	90.00% (±6.71)	0.00% (±5.59)
BULLSEYE POLYTOPE	FINE-TUNING	75.00% (±9.68)	0.00% (±5.59)
BULLSEYE POLYTOPE	TRANSFER	100.00% (±5.59)	10.00% (±6.71)
POISON FROGS	TRANSFER	100.00% (±5.59)	15.00% (±7.98)
GRADIENT MATCHING (SE)	TRANSFER	95.00% (±4.87)	0.00% (±5.59)
CONVEX POLYTOPE	TRANSFER*	90.00% (±10.00)	40.00 % (±16.32)
HIDDEN TRIGGER BACKDOOR	TRANSFER	55.59% (±5.65)	24.78% (±6.82)

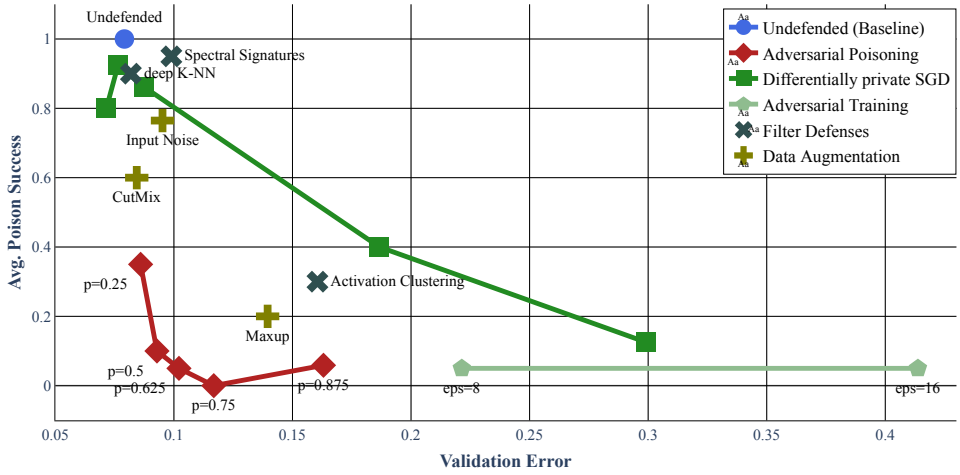


Figure 3. Avg. Poison Success versus validation accuracy for various defenses against the gradient matching attack of (Geiping et al., 2021) in the from-scratch setting. The baseline undefended model is shown in blue, the proposed defense in red. The differentially private SGD is shown for noise values from 0.0001 to 0.01. The proposed defense provides a strong trade-off of robustness and accuracy.

a completely new model.

We first apply the proposed defense against a range of attacks and settings in table 1, choosing $p = 0.75$ for all targeted data poisoning attacks and $p = 0.5$ for all backdoor attacks and no additional modifications. All attacks shown are adaptive, if possible. In the fine-tuning and transfer scenarios, the pre-trained model is defended but known to the attacker exactly. In all cases, we observe that while the attacks are highly effective against an undefended model, our defense steeply reduces the effectiveness of both poisons and backdoors. These encouraging results suggest that the proposed methodology is a strong strategy that can be robustly applied across a range of attacks and may also be applicable to new settings and attacks proposed the future.

5.2. Comparison to Other Defenses

In this subsection, we compare the proposed defense to other existing defense strategies against data poisoning including differentially private SGD, adversarial training, various data augmentations, and filter defenses. For differentially private SGD and adversarial training, we test several noise levels and perturbation budgets, respectively. When comparing to filtering defenses, we allow an optimal hyperparameter choice by supplying the exact number of poisons in the training set, although this information would be unknown in practice. We analyze adversarial poisoning with varying

Table 2. Defenses against feature collision via (Aghakhani et al., 2020) in the transfer setting for a budget of 1% and bound of $\varepsilon = 16$.

Defense	Poison Success	Val. Acc.
None	100.00% (±5.59)	91.97%
Random Noise	90.00% (±6.71)	90.45%
Deep K-NN	75.00% (±9.68)	91.94%
Activation Clustering	0.00% (±5.59)	91.34%
Spectral Signatures	0.00% (±5.59)	92.09%
Diff. Priv. SGD ($n = 0.0001$)	100.00% (±5.59)	92.72%
Diff. Priv. SGD ($n = 0.001$)	100.00% (±5.59)	91.26%
Diff. Priv. SGD ($n = 0.01$)	85.00% (±7.98)	69.78%
Adv. Training ($\varepsilon = 8$)	5.00% (±4.87)	77.80%
Adv. Training ($\varepsilon = 16$)	5.00% (±4.87)	58.98%
Adv. Poisoning ($p = 0.25$)	90.00% (±6.71)	91.24%
Adv. Poisoning ($p = 0.5$)	70.00% (±10.25)	89.83%
Adv. Poisoning ($p = 0.75$)	10.00% (±6.71)	88.64%

levels of p to show the trade-off of performance and security.

We start by conducting our comparison in the common from-scratch setting, where the entire model is re-trained. We test the gradient matching attack proposed in Geiping et al. (2021), for a ResNet-18 trained on CIFAR-10 with budget 1% and $\varepsilon = 16$. While previous defenses were shown to be ineffective in Geiping et al. (2021), we now show in fig. 3 that the proposed adversarial poisoning defense is an extremely effective defense in the from-scratch setting, yielding a much stronger protection than filter defenses, but with only mild trade-off in validation accuracy compared to differential privacy and adversarial training.

Second, we compare adversarial poisoning to other existing defenses in the transfer setting, where only the last layer is re-trained on poisoned data. We test feature collisions via Bullseye Polytope (Aghakhani et al., 2020), also for a ResNet-18 trained on CIFAR-10 with a budget of 1% poisoned data within an ℓ^∞ bound of $\varepsilon = 16$. This is a setting that is ideal for commonly used filtering defenses, as a large number of poisoned data is collided with the target image, which can be detected and filtered, while the setting is difficult for robust training methods, both due to the large perturbations and due to the limitation that only the last layer is re-trained - leaving less control over the model. We record results in table 2, finding that adversarial poisoning can be effective even in a scenario that favors filter defenses, matching filter defenses, while beating adversarial training significantly in the trade-off against validation performance.

5.3. Training on One Attack Yields Robustness to Others

A natural question to ask is whether this defense, which trains against one specific surrogate attack, can be circumvented when the real attacker utilizes a different attack. In the transfer setting, where we can easily evaluate multiple attacks under comparable conditions, we test this hypothesis and report the results in table 3. We find that using gradient matching, a strong attack, in adversarial poisoning successfully defends against a range of other attacks.

Table 3. Defending against various attacks in the transfer setting using only gradient matching as defense, with $p = 0.75$.

Attack	Poison Success	Val. Acc.
Poison Frogs	15.79% (±8.37)	87.86%
Gradient Matching (SE)	6.67% (±6.44)	87.85%
Bullseye	5.26% (±5.12)	87.90%
Hidden Trigger	3.32% (±0.79)	87.94%

5.4. Releasing Robust Models

So far, we have considered scenarios in which the defense described in algorithm 1 is always active, even during the fine-tuning procedure in the transfer setting. However, especially in the transfer setting, we are interested in the inherent robustness of models and its transferability. We thus analyze a setting in which the base model is trained robustly via adversarial poisoning, but the last layer is re-trained non-robustly on poisoned data. In the same setting as shown in table 2, this approach leads to an avg. poison success of 20.00%(±8.94) and validation accuracy of 88.66 when using a base model trained robustly with adversarial poisoning via $p = 0.75$. Figure 4 visualizes that the target confidence remain consistent, even if the fine-tuning is non-robust. This implies that robustness is an inherent quality of the model, and may make it attractive to release models robust to poisoning.

6. Conclusions

In this work, we adapt adversarial training to defend against data poisoning and backdoor attacks. In addition to demonstrating the strong defensive capabilities of our method, adversarial poisoning, we analyze the feature space of defended models and observe mechanisms of defense. We further evaluate the proposed defense against a variety of attacks on deep neural networks for image classification, successfully adapting to and defending against feature collisions, gradient matching, backdoor triggers and hidden trigger backdoors. We stress that we believe this strategy to be a general paradigm for defending against data tampering attacks that can extend to novel future attacks.

Acknowledgements

This work was supported by the DARPA GARD and DARPA YFA programs. Additional support was provided

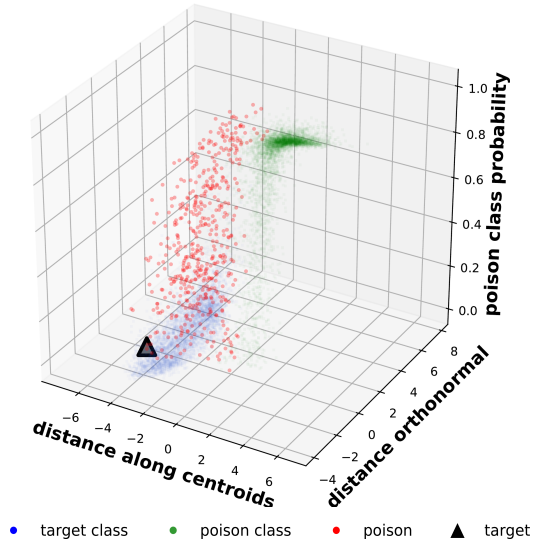


Figure 4. Feature Defense. A robust base model can withstand feature collision attacks even when fine-tuning non-robustly on poisoned data.

by DARPA QED and the National Science Foundation DMS program.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pp. 308–318, Vienna, Austria, October 2016. Association for Computing Machinery. ISBN 978-1-4503-4139-4. doi: 10.1145/2976749.2978318.

Aghakhani, H., Meng, D., Wang, Y.-X., Kruegel, C., and Vigna, G. Bullseye Polytope: A Scalable Clean-Label Poisoning Attack with Improved Transferability. *arXiv:2005.00191 [cs, stat]*, April 2020.

Barreno, M., Nelson, B., Joseph, A. D., and Tygar, J. D. The security of machine learning. *Machine Language*, 81(2):121–148, November 2010. ISSN 0885-6125. doi: 10.1007/s10994-010-5188-5.

Borgnia, E., Cherepanova, V., Fowl, L., Ghiasi, A., Geiping, J., Goldblum, M., Goldstein, T., and Gupta, A. Strong Data Augmentation Sanitizes Poisoning and Backdoor Attacks Without an Accuracy Tradeoff. *arXiv:2011.09527 [cs]*, November 2020.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language Models are Few-Shot Learners. In *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, December 2020.

Carlini, N. and Wagner, D. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17, pp. 3–14, Dallas, Texas, USA, November 2017. Association for Computing Machinery. ISBN 978-1-4503-5202-4. doi: 10.1145/3128572.3140444.

Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., Madry, A., and Kurakin, A. On Evaluating Adversarial Robustness. *arXiv:1902.06705 [cs, stat]*, February 2019.

Chen, B., Carvalho, W., Baracaldo, N., Ludwig, H., Edwards, B., Lee, T., Molloy, I., and Srivastava, B. Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering. In *SafeAI@AAAI*, January 2019.

Chen, X., Wang, W., Bender, C., Ding, Y., Jia, R., Li, B., and Song, D. REFIT: A Unified Watermark Removal Framework for Deep Learning Systems with Limited Data. *arXiv:1911.07205 [cs]*, January 2020.

DeVries, T. and Taylor, G. W. Improved Regularization of Convolutional Neural Networks with Cutout. *arXiv:1708.04552 [cs]*, August 2017.

Gao, Y., Xu, C., Wang, D., Chen, S., Ranasinghe, D. C., and Nepal, S. STRIP: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, ACSAC '19, pp. 113–125, New York, NY, USA, December 2019. Association for Computing Machinery. ISBN 978-1-4503-7628-0. doi: 10.1145/3359789.3359790.

Geiping, J., Fowl, L. H., Huang, W. R., Czaja, W., Taylor, G., Moeller, M., and Goldstein, T. Witches' Brew: Industrial Scale Data Poisoning via Gradient Matching. In *International Conference on Learning Representations*, 2021.

Goldblum, M., Tsipras, D., Xie, C., Chen, X., Schwarzschild, A., Song, D., Madry, A., Li, B., and Goldstein, T. Dataset Security for Machine Learning: Data Poisoning, Backdoor Attacks, and Defenses. *arXiv:2012.10544 [cs]*, December 2020.

Gong, C., Ren, T., Ye, M., and Liu, Q. MaxUp: A Simple Way to Improve Generalization of Neural Network Training. *arXiv:2002.09024 [cs, stat]*, February 2020.

Gu, T., Liu, K., Dolan-Gavitt, B., and Garg, S. BadNets: Evaluating Backdooring Attacks on Deep Neural Networks. *IEEE Access*, 7:47230–47244, 2019. ISSN 2169-3536. doi: 10.1109/ACCESS.2019.2909068.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015.

Hong, S., Chandrasekaran, V., Kaya, Y., Dumitraş, T., and Papernot, N. On the Effectiveness of Mitigating Data Poisoning Attacks with Gradient Shaping. *arXiv:2002.11497 [cs]*, February 2020.

Huang, W. R., Geiping, J., Fowl, L., Taylor, G., and Goldstein, T. MetaPoison: Practical General-purpose Clean-label Data Poisoning. In *Advances in Neural Information Processing Systems*, volume 33, Vancouver, Canada, December 2020.

Jaderberg, M., Simonyan, K., Zisserman, A., and kavukcuoglu, k. Spatial Transformer Networks. In *Advances in Neural Information Processing Systems 28*, pp. 2017–2025. Curran Associates, Inc., 2015.

Jayaraman, B. and Evans, D. Evaluating Differentially Private Machine Learning in Practice. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pp. 1895–1912, 2019. ISBN 978-1-939133-06-9.

Koh, P. W., Steinhardt, J., and Liang, P. Stronger Data Poisoning Attacks Break Data Sanitization Defenses. *arXiv:1811.00741 [cs, stat]*, November 2018.

Kumar, R. S. S., Nyström, M., Lambert, J., Marshall, A., Goertzel, M., Comissoneru, A., Swann, M., and Xia, S. Adversarial Machine Learning-Industry Perspectives. In *2020 IEEE Security and Privacy Workshops (SPW)*, pp. 69–75, May 2020. doi: 10.1109/SPW50608.2020.00028.

Levine, A. and Feizi, S. Deep Partition Aggregation: Provable Defenses against General Poisoning Attacks. In *International Conference on Learning Representations*, 2021.

Liu, K., Dolan-Gavitt, B., and Garg, S. Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks. In *Research in Attacks, Intrusions, and Defenses*, Lecture Notes in Computer Science, pp. 273–294, Cham, 2018. Springer International Publishing. ISBN 978-3-030-00470-5. doi: 10.1007/978-3-030-00470-5_13.

Ma, Y., Zhu, X., and Hsu, J. Data Poisoning against Differentially-Private Learners: Attacks and Defenses. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 4732–4738, Macao, China, August 2019. International Joint Conferences on Artificial Intelligence Organization. ISBN 978-0-9992411-4-1. doi: 10.24963/ijcai.2019/657.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*, February 2018.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. In *NIPS 2017 Autodiff Workshop*, Long Beach, CA, 2017.

Paudice, A., Muñoz-González, L., Gyorgy, A., and Lupu, E. C. Detection of Adversarial Training Examples in Poisoning Attacks through Anomaly Detection. *arXiv:1802.03041 [cs, stat]*, February 2018.

Peri, N., Gupta, N., Huang, W. R., Fowl, L., Zhu, C., Feizi, S., Goldstein, T., and Dickerson, J. P. Deep k-NN Defense Against Clean-Label Data Poisoning Attacks. In *Computer Vision – ECCV 2020 Workshops*, Lecture Notes in Computer Science, pp. 55–70, Cham, 2020. Springer International Publishing. ISBN 978-3-030-66415-2. doi: 10.1007/978-3-030-66415-2_4.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, December 2015. ISSN 1573-1405. doi: 10.1007/s11263-015-0816-y.

Saha, A., Subramanya, A., and Pirsiavash, H. Hidden Trigger Backdoor Attacks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):11957–11965, April 2020. ISSN 2374-3468. doi: 10.1609/aaai.v34i07.6871.

Schwarzschild, A., Goldblum, M., Gupta, A., Dickerson, J. P., and Goldstein, T. Just How Toxic is Data Poisoning? A Unified Benchmark for Backdoor and Data Poisoning Attacks. *arXiv:2006.12557 [cs, stat]*, June 2020.

Shafahi, A., Huang, W. R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., and Goldstein, T. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 6106–6116, Red Hook, NY, USA, December 2018. Curran Associates Inc.

Sinha, A., Namkoong, H., and Duchi, J. Certifying Some Distributional Robustness with Principled Adversarial Training. In *International Conference on Learning Representations*, February 2018.

Tan, T. J. L. and Shokri, R. Bypassing Backdoor Detection Algorithms in Deep Learning. In *2020 IEEE European Symposium on Security and Privacy (EuroS P)*, pp. 175–183, September 2020. doi: 10.1109/EuroSP48549.2020.00019.

Tramer, F., Carlini, N., Brendel, W., and Madry, A. On Adaptive Attacks to Adversarial Example Defenses. In *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, Vancouver, Canada, December 2020.

Tran, B., Li, J., and Madry, A. Spectral Signatures in Backdoor Attacks. In *Advances in Neural Information Processing Systems 31*, pp. 8000–8010. Curran Associates, Inc., 2018.

Turner, A., Tsipras, D., and Madry, A. Clean-Label Backdoor Attacks. *openreview*, September 2018.

Veldanda, A. and Garg, S. On Evaluating Neural Network Backdoor Defenses. *arXiv:2010.12186 [cs]*, October 2020.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., and Zhao, B. Y. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 707–723, San Francisco, CA, USA, May 2019. IEEE. ISBN 978-1-5386-6660-9. doi: 10.1109/SP.2019.00031.

Weber, M., Xu, X., Karlas, B., Zhang, C., and Li, B. RAB: Provable Robustness Against Backdoor Attacks. *arXiv:2003.08904 [cs, stat]*, June 2020.

Xu, X., Wang, Q., Li, H., Borisov, N., Gunter, C. A., and Li, B. Detecting AI Trojans Using Meta Neural Analysis. *arXiv:1910.03137 [cs]*, October 2020.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6023–6032, 2019.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. Mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations*, February 2018.

Zhu, C., Huang, W. R., Li, H., Taylor, G., Studer, C., and Goldstein, T. Transferable Clean-Label Poisoning Attacks on Deep Neural Nets. In *International Conference on Machine Learning*, pp. 7614–7623. PMLR, May 2019.

A. Appendix

This appendix contains additional details, detailing the experimental setup in general, adding experimental details for all attacks and defenses and additional visualizations for Poison-Frogs, Backdoor Triggers, and gradient matching. We also replicate all three-dimensional visualizations in two dimensions.

B. Visualizations

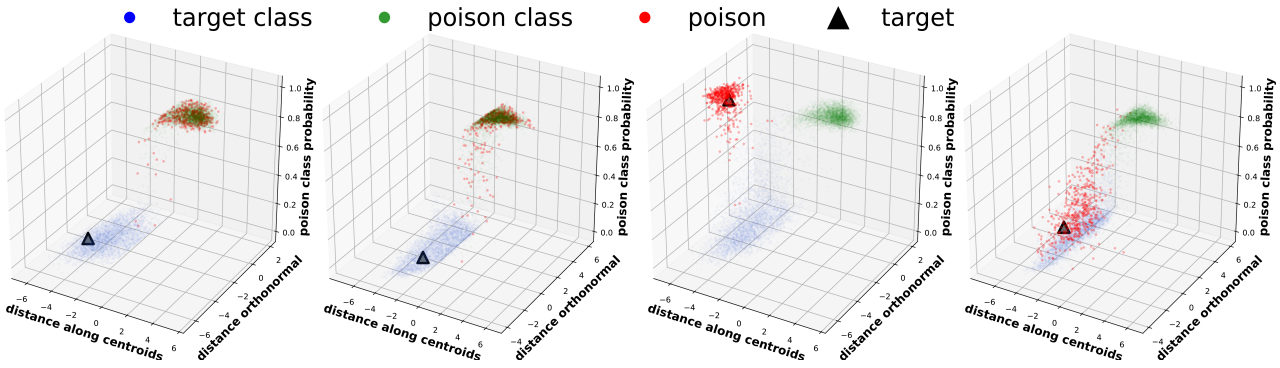
We repeat the three-dimensional visualizations shown in Figure 2 in the main work for additional attacks. For easy comparison, we also repeat the figures appearing in the main work. We show feature collisions via Poison Frogs in fig. 5 and repeat Bulleye Polytope in fig. 6. We then repeat gradient matching in fig. 7, in comparison to backdoor trigger in fig. 8 and gradient matching (SE) in fig. 9.

All three-dimensional visualizations are also shown in two dimensions, showing Poison Frogs in fig. 10, Bullseye Polytope in fig. 11, gradient matching in fig. 12, backdoor triggers in fig. 13 and gradient matching (SE) in fig. 14.

C. Experimental Setup

In general terms, the goal of our experimental setuo is to standardize the experimental conditions encountered in various works in data poisoning to a degree that allows for convenient comparisons across attacks. Furthermore previous works have focused on showcasing the smallest possible adversarial modifications that still achieve a substantially malicious effect. However, such setups "on the edge" are broken too easily by any defenses, so that we generally consider stronger attacks in this work than in their original implementations. On the other hand, there is an upper limit to this design because attacks have to be sufficiently realistic, modifying only parts of the dataset within limits - unlimited adversarial modifications would allow for unlimited attack strength.

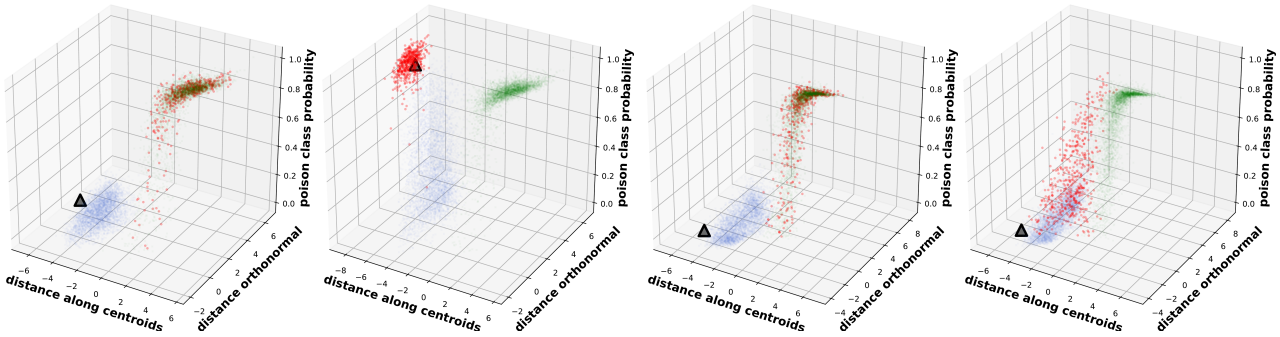
For all experiments shown in this work, we standardize the machine learning model that is attacked to a deep neural network for image classification, namely always a ResNet-18 model trained on the CIFAR-10 dataset. The ResNet-18 model follows (He et al., 2015), with the customary CIFAR-10 modification of replacing the initial 7x7 convolution and max-pooling with a 3x3 convolution. We train this model using SGD with Nesterov momentum ($m = 0.9$) with a batch size of 128 for 40 epochs with an initial learning rate of 0.1, which is reduced by a factor of 10 after $\frac{3}{8}$, $\frac{5}{8}$ and $\frac{7}{8}$ of all epochs. The model is additionally regularized by weight decay with weight 5×10^{-4}. The CIFAR-10 dataset is augmented with horizontal flips and continuous random crops from images with a zero padding of 4.



(a) **Undefended model**, clean (left) and fine-tuned on poisoned data containing feature collisions. (right).

(b) **Defended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

Figure 5. 3D Visualization of a feature collision attack (via **Poison-Frogs**) against an undefended and a defended model. The defended model significantly hinders feature collisions. The target image is marked by a black triangle and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green.



(a) **Undefended model**, clean (left) and fine-tuned on poisoned data containing feature collisions. (right).

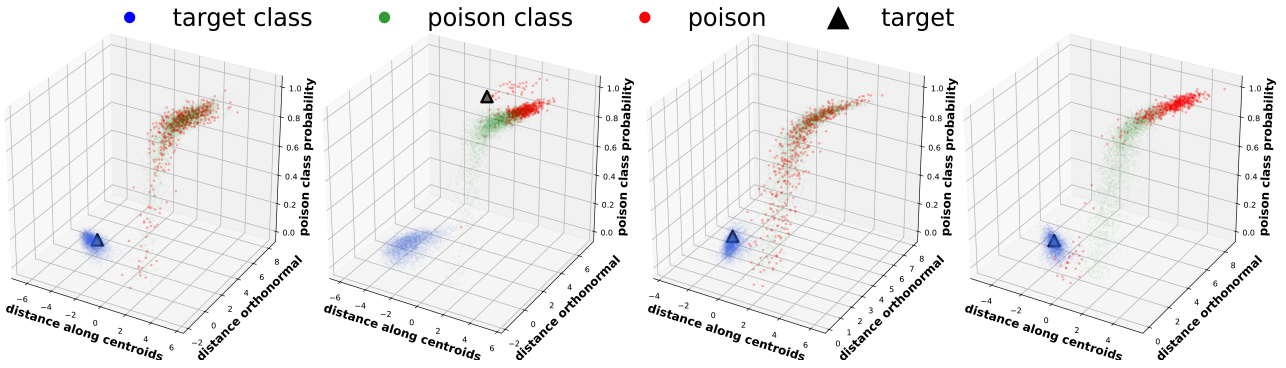
(b) **Defended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

Figure 6. 3D Visualization of the effects of a feature collision (**Bullseye Polytope**) attack against an undefended and a defended model. The defended model significantly hinders feature collisions. The target image is marked by a black triangle and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green. Notably the strong collision seen in the baseline is inhibited by the defense.

We directly train with these hyperparameters in the FROM-SCRATCH setting. For the TRANSFER experiments we first train a ResNet-18 with this setup, which we call the base model, and then freeze its feature representation. We then retrain the linear layer from a random initialization with the same hyperparameters. For the FINE-TUNING experiments we drop the learning rate by 0.001 before fine-tuning from the base model, likewise reinitializing the linear layer, but not freezing the feature representation. This setup for fine-tuning and transfer is arguably easier to attack than a transfer to an unknown dataset (as investigated for example in (Shafahi et al., 2018)), as all features are already optimized to be relevant to the given task, and as such we do not recommend it as the only evaluation of an attack, but believe it is an appropriate worst-case setting for the defense experiments considered in this work. Note that we apply a slightly different setting for the convex polytope attack of Zhu et al. (2019) which we will detail together with the attack in the next section and mark by TRANSFER* in the main table.

C.1. Measuring poison effectiveness

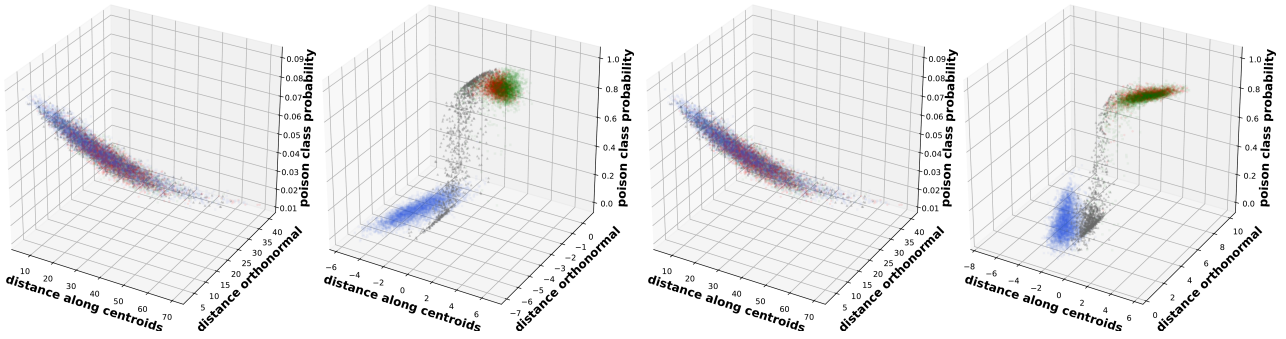
We run data poisoning attacks with the goal of maliciously classifying a single target image for all targeted data poisoning attacks, and with the goal of maliciously classifying 1000 images patched with a single target trigger for backdoor attacks. In both cases these images are drawn from the validation set without replacement. We report success for an attack if the target image, or patched image is classified with the adversarial label, we do not count mis-classifications into a third label. In all experiments, we then report *avg. poison success*. This metric represents the average success over N random trials, where each trial consists of a



(a) **Undefended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

(b) **Defended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

Figure 7. 3D Visualization of the effects of a gradient matching attack (**Witches' Brew**) against an undefended and a defended model. The defended model significantly hinders feature collisions. The target image is marked by a black triangle and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green.



(a) **Undefended model**, clean (left) and retrained on poisoned data containing backdoor patches. (right).

(b) **Defended model**, clean (left) and retrained on poisoned data containing backdoor patches (right).

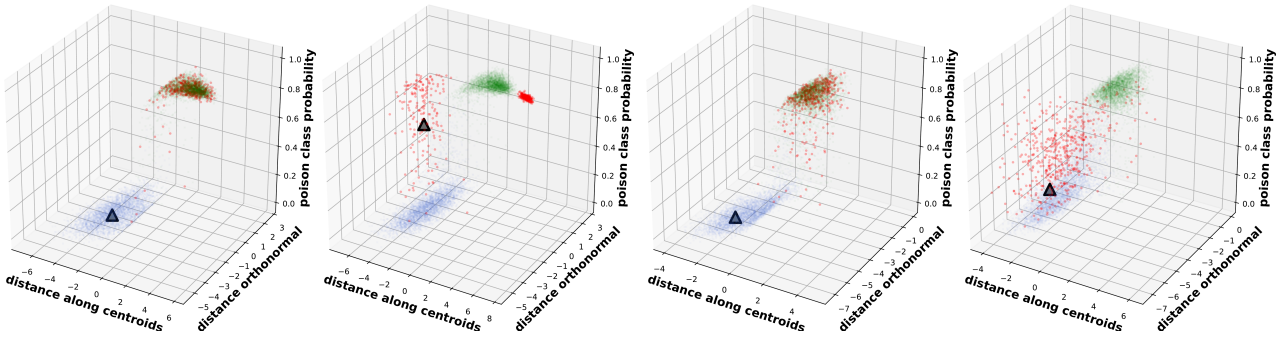
Figure 8. 3D Visualization of the effects of a **backdoor trigger** patch attack against an undefended and a defended model. The target trigger is applied to a number of target images shown in black and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green. Note how the black datapoints are associated with the poison class in the undefended case, but correctly associate with the target class in the defended case.

randomly drawn target trigger or image, randomly chosen adversarial class, randomly chosen subset of images to be poisoned (from the adversarial class) and random model initializations. We control these trials with by specifying their random seed. All experiments in this work are based on the same 20 fixed trials, which we list by their seed within the supplemented code submission and as such comparable.

For all attacks we consider an ℓ^∞ bound of $\varepsilon = 16$ for targeted data poisoning attacks. For backdoor triggers, we allow triggers with a size of 6 by 6 pixels, i.e. a rectangular arrangment of the ℓ^0 bound of $\varepsilon = 36$. If not otherwise noted we allow a budget of 1% of the dataset to be modified.

C.2. Attack settings

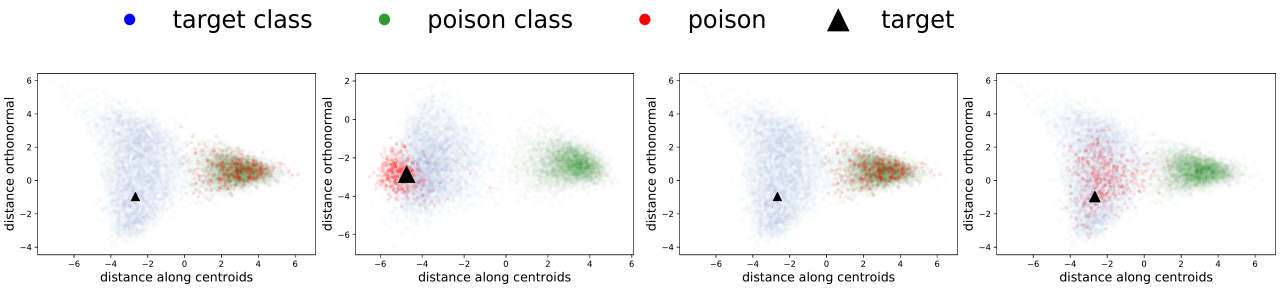
For all attacks we optimize the adversarial perturbation through projected descent (PGD). We found signed Adam with a step size of 0.1 to be a robust first-order optimization tool to this end which we run for 240 steps, reducing the step size by a factor of 10 after $\frac{3}{8}$, $\frac{5}{8}$ and $\frac{7}{8}$ of total steps. Basic data poisoning attacks are often brittle when encountering simple data augmentations (Schwarzschild et al., 2020). As we include data augmentations in our experimental setup, we also include them during the attack algorithm for all iterative attacks, sampling a random augmentation in every update step and differentiating through the resulting transformation via grid sampling (Jaderberg et al., 2015). Some works such as Geiping et al. (2021) consider multiple restarts of the attack algorithm, however we consistently run all attacks with a single restart, mostly due to computational constraints, and given that restarts appear to confer only a minor benefit.



(a) **Undefended model**, clean (left) and fine-tuned on poisoned data containing gradient matching attacks (right).

(b) **Defended model**, clean (left) and fine-tuned on poisoned data containing gradient matching attacks (right).

Figure 9. 3D Visualization of the effects of a gradient matching attack (**Witches' Brew** with squared loss) against an undefended and a defended model. The defended model significantly hinders feature collisions. The target image is marked by a black triangle and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green.
We see that the attack effectively moves the decision boundary opposite of the target in feature space. However this is completely prevented in the defended model.



(a) **Undefended model**, clean (left) and fine-tuned on poisoned data containing feature collisions. (right).

(b) **Defended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

Figure 10. 2D Visualization of a feature collision attack (via **Poison-Frogs**) against an undefended and a defended model. The defended model significantly hinders feature collisions. The target image is marked by a black triangle and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green.

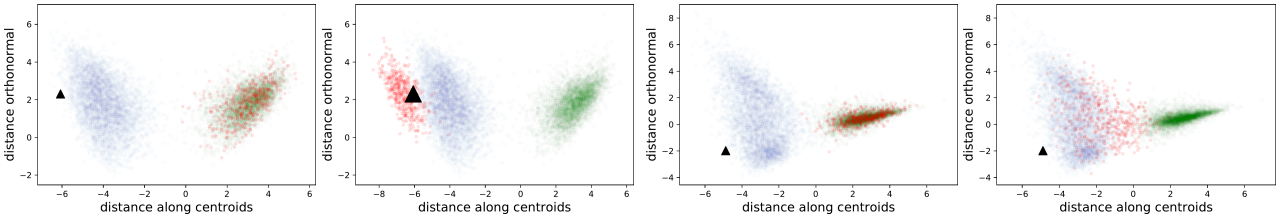
Poison Frogs. We implement the feature collision objective as proposed in Shafahi et al. (2018). However, while perturbation bounds were weakly enforced in the original version by an additional penalty, we instead optimize the objective directly by projected (signed) gradient descent in line with other attacks. We find this to be at least equally effective.

Convex Polytope. The TRANSFER* attack in table 1 uses the setup from Schwarzschild et al. (2020). Poisoned data created by Convex Polytope can be brittle in terms of the amount of training data and optimizer settings. Therefore, to get an idea of how well our defense works against this attack, we implement a modified setting wherein the attack succeeds. This includes using a feature extractor trained on CIFAR-100, and training the last linear layer for only 10 epochs using Adam optimizer with lr$= 0.1$. We otherwise applied the attack as proposed in Zhu et al. (2019).

Bullseye Polytope. We directly re-implement the attack based on eq.(2) in Aghakhani et al. (2020).

Witches' Brew. We implement gradient matching as in Geiping et al. (2021). However, we modify the original attack in the transfer setting. The original attack is posed for from-scratch attacks on large models and the objective of cosine similarity of parameter gradients does not scale well to small models. For small models (such as the transfer case, where only the last linear layer is retrained), we instead measure similarity in the squared Euclidean norm. We refer to this variant as gradient matching with squared error, e.g. Gradient Matching (SE) in table 1.

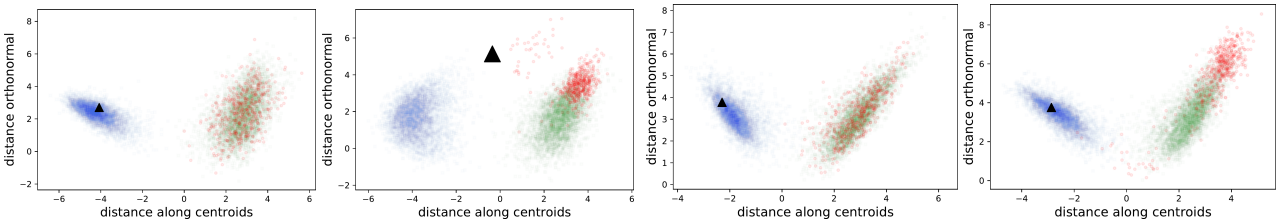
MetaPoison We download premade poisoned datasets for MetaPoison from `https://github.com/wronnyhuang/metapoison`, validate their effectiveness and and then deploy our proposed defense using our own surrogate attack at the batch level, which we unroll



(a) **Undefended model**, clean (left) and fine-tuned on poisoned data containing feature collisions. (right).

(b) **Defended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

Figure 11. 2D Visualization of the effects of a feature collision (**Bullseye Polytope**) attack against an undefended and a defended model. The defended model significantly hinders feature collisions. The target image is marked by a black triangle and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green. Notably the strong collision seen in the baseline is inhibited by the defense.



(a) **Undefended model**, clean (left) and retrained on poisoned data containing gradient matching attacks. (right).

(b) **Defended model**, clean (left) and retrained on poisoned data containing gradient matching attacks (right).

Figure 12. 2D Visualization of the effects of a gradient matching attack (**Witches' Brew**) against an undefended and a defended model. The defended model significantly hinders feature collisions. The target image is marked by a black triangle and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green.

for 2 steps as also proposed for the attacks in Huang et al. (2020). We defend using only the current estimate of the model as discussed in previous sections (instead of replicating the ensemble of 24 models used to create the poisoned dataset in some fashion). We train the usual ResNet-18 model on the downloaded poisoned CIFAR-10 for 80 epochs with data augmentations, conforming to the training setup based on which the poisons were created. We download poisoned datasets for a budget of 1% for the bird-dog setting with bird target ids 0 to 9 with perturbations bounded by $\varepsilon = 8$ in ℓ^∞-norm and perturbed by 4% in color space.

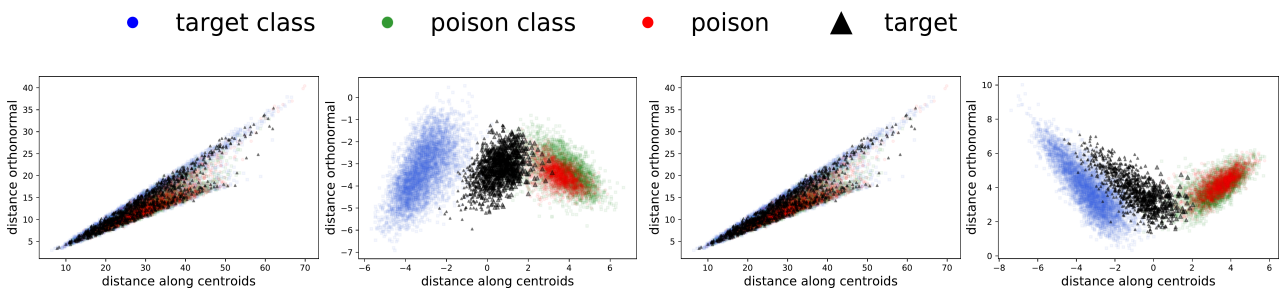
Backdoor Triggers Also referred to as "patch attacks", this attack was introduced in Gu et al. (2019). For this attack we sample a checkerboard pattern, independently for all three colors channels, with a size of 6x6 pixels. We then imprint this patch in the lower right corner of all images in the poison set. During evaluation we imprint the same patch in the same location for 1000 target images. We run this attack with a budget of 5% of the training set, to increase its success in comparison to targeted attacks.

Hidden Trigger Backdoor For the hidden trigger backdoor attack of Saha et al. (2020) we use triggers identical to the ones used in the original work (see `https://github.com/UMBCvision/Hidden-Trigger-Backdoor-Attacks`). The number of poisons we use is indicated in the main body experiments, and is on same order as the number used in the original work. Specifically, we evaluate the attack on 1000 patched target images and choose a budget of 5%. The adversarial perturbations to the subset of poisoned data are optimized as described in the general attack settings, minimizing the hidden trigger objective of matching poison features to features of patched images.

C.3. Defense settings

Input Noise As a sanity check we include a comparison to input noise. We draw random noise from the boundary of the set of allowed perturbations by independently sampling from a Bernoulli distribution for each value, and assigning either $-\varepsilon$ or ε to each value.

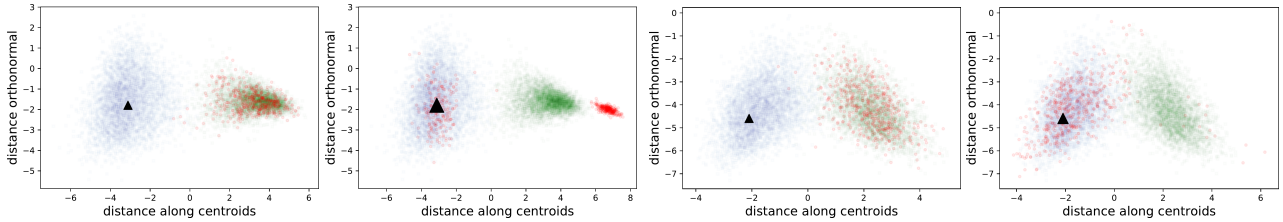
CutMix and Maxup We apply CutMix (Yun et al., 2019) as a defense against data poisoning as proposed in Borgnia et al. (2020). We attack this defense adaptively by creating poisoned data based on a clean model trained with CutMix as well. The same considerations apply for Maxup (Gong



(a) **Undefended model**, clean (left) and retrained on poisoned data containing backdoor patches. (right).

(b) **Defended model**, clean (left) and retrained on poisoned data containing backdoor patches (right).

Figure 13. 2D Visualization of the effects of a **backdoor trigger** patch attack against an undefended and a defended model. The target trigger is applied to a number of target images shown in black and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green. Note how the black datapoints are associated with the poison class in the undefended case, but correctly associate with the target class in the defended case.



(a) **Undefended model**, clean (left) and fine-tuned on poisoned data containing gradient matching attacks. (right).

(b) **Defended model**, clean (left) and fine-tuned on poisoned data containing gradient matching attacks (right).

Figure 14. 2D Visualization of the effects of a gradient matching attack (**Witches' Brew** with squared error) against an undefended and a defended model. The defended model significantly hinders feature collisions. The target image is marked by a black triangle and is originally part of the class marked in blue. The poisoned images are marked in red and are part of the class marked in green.
We see that the attack effectively moves the decision boundary by placing poisoned data opposite of the target in feature space. However this is completely prevented in the defended model.

et al., 2020). We use Cutout (DeVries & Taylor, 2017) as a base augmentation for maxup, and select the worst-case augmentation from four examples.

Spectral Signatures We implement the defense as proposed in Tran et al. (2018), using the provided overestimation factor of 1.5. We supply the attack budget as additional info for this defense.

Deep K-NN We implement the defense as proposed in Peri et al. (2020), using the provided overestimation factor of 2. We supply the attack budget as additional info for this defense.

Activation Clustering We run the defense of Chen et al. (2019), clustering the available training data into two clusters.

Differentially private SGD We implement a variant of differentially private SGD with gradient clipping to a value of 1 on a mini-batch level (as suggested in Hong et al. (2020)), and varying levels of Gaussian noise applied to the mini-batch gradient. Attacks can adapt to this defense by adding gradient noise to their surrogate estimation of gradients (this is mostly relevant for gradient matching where surrogate gradients appear explicitely).

Adversarial Training We implement straightforward adversarial training, starting from a randomly initialized perturbation and maximizing cross entropy for 5 steps via signed descent.

Adversarial Poisoning For all implementations of adversarial poisoning we replicate the original objective of the attack in the mini-batch setting, but optimize for only 5 steps, based on features or gradients from the current model. The surrogate attacks are optimized via signed Adam descent with the same parameters as described in the attack section. For backdoor triggers, we do not need to optimize and sample a random checkerboard pattern with a random rectangular shape within $\ell^0 < 45$ (overestimating the actual ℓ^0 bound for a gray-box setting) as well as a random loca-

tion. We sample such a patch for every class in the dataset and then apply them to randomly chosen pairs of classes, replicating the attack without knowing the targeted class. For the hidden trigger backdoor attack we draw patches in the same way, but optimize the perturbations for 5 steps as described for targeted attacks.

C.4. Hardware

We run all reported experiments using `Nvidia GEFORCE RTX 2080 Ti` GPUs, using one GPU per experimental run. We also ran select ablation experiments using a `Nvidia V100` setup.