Not All Poisons are Created Equal: Robust Training against Data Poisoning

Yu Yang [1] **Tian Yu Liu** [1] **Baharan Mirzasoleiman** [1]

Abstract

Data poisoning causes misclassification of test time target examples, by injecting maliciously crafted samples in the training data. Existing defenses are often effective only against a specific type of targeted attack, significantly degrade the generalization performance, or are prohibitive for standard deep learning pipelines. In this work, we propose an efficient defense mechanism that significantly reduces the success rate of various data poisoning attacks, and provides theoretical guarantees for the performance of the model. Targeted attacks work by adding bounded perturbations to a randomly selected subset of training data to match the targets' gradient or representation. We show that: (i) under bounded perturbations, only a number of poisons can be optimized to have a gradient that is close enough to that of the target and make the attack successful; (ii) such effective poisons move away from their original class and get isolated in the gradient space; (iii) dropping examples in low-density gradient regions during training can successfully eliminate the effective poisons, and guarantees similar training dynamics to that of training on full data. Our extensive experiments show that our method significantly decreases the success rate of state-of-the-art targeted attacks, including Gradient Matching and Bullseye Polytope, and easily scales to large datasets[1].

1. Introduction

The impressive success of modern machine learning systems is highly dependent on the quality of their large training data. Many large datasets are scraped from the internet, or other public and user-provided sources. Models trained on such datasets are susceptible to data poisoning attacks, wherein an adversary places specially-constructed poisoned examples into the training data with the intention of manipulating the behavior of the system at test time. These attacks create security vulnerabilities that cannot be detected even if the data is labeled and checked by human supervision. This makes data poisoning arguably one of the most concerning threats to deep learning systems deployed in security- and safety-critical applications, such as financial services, security cameras, autonomous cars, and medical devices.

Various types of poisoning attacks have been proposed in recent years. Most attacks fall into one of two main categories: backdoor or triggerless poisoning. Backdoor data poisoning augments the training data by a set of poisoned examples that contain a (not necessarily visible) trigger pattern (Gu et al., 2017; Turner et al., 2018; Souri et al., 2021). Finetuning the model on the augmented training data causes a model to misclassify test-time samples containing the trigger. On the other hand, triggerless poisoning attacks work by crafting small per-example perturbations so that the perturbed training examples collide with the adversarially labeled target in the feature or gradient space (Shafahi et al., 2018; Zhu et al., 2019; Huang et al., 2020; Geiping et al., 2021b; Aghakhani et al., 2021). Triggerless poisoning attacks cause misclassification of particular instances and do not require modification at inference time. In both cases, the poisoned examples may be seemingly innocent and properly labeled, and hence are hard to be detected by expert observers.

Existing defense mechanisms against data poisoning attacks mainly rely on either anomaly detection based on nearest neighbors, training loss, singular-value decomposition, feature and activation clustering (Cretu et al., 2008; Steinhardt et al., 2017; Tran et al., 2018; Chen et al., 2019; Peri et al., 2020), or robust training based on strong data augmentation, randomized smoothing, ensembling, and adversarial training (Weber et al., 2020; Levine & Feizi, 2020; Abadi et al., 2016; Ma et al., 2019; Li et al., 2021; Tao et al., 2021). However, such methods either drastically degrade the generalization performance of the model (Geiping et al., 2021a), or can only protect the model against certain types of poisoning attacks (Koh et al., 2018; Tran et al., 2018), or are computationally prohibitive for standard deep learning pipelines (Geiping et al., 2021a). Importantly, these methods do not provide any theoretical guarantee for the performance of the model (Weber et al., 2020; Levine &

[1]Department of Computer Science, University of California, Los Angeles, United States. Correspondence to: Yu Yang <yuyang@cs.ucla.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

[1]Code is available at https://github.com/YuYang0901/Epic.

Feizi, 2020; Abadi et al., 2016; Geiping et al., 2021a).

We develop an efficient and principled defense framework that effectively prevents various types of targeted poisoning attacks, and provide theoretical guarantee for the performance of the model. To successfully prevent poisoning attacks, we make the following key observation: not all poisons are effective in making the attack successful. In particular, targeted attacks add bounded perturbations to randomly selected subsets of training data to match the gradient of the adversarially labeled target. We show that for a poison to be effective, it needs to fall close enough to the target in the gradient space. However, under bounded perturbations, only a small number of poisons can be optimized to get close enough to the target and make the attack successful. Such effective poisons get far away from their original class and get isolated in the gradient space. Eliminating the *effective poisons* can successfully break various types of attacks.

To prevent data poisoning while maintaining the generalization performance of the network, we aim to identify and eliminate the effective poisons. We show that effective poisons can be identified as isolated *medoids* of each class, in the gradient space. Medoids are the most centrally located examples of a dataset, that minimize the sum of dissimilarity between every data point to its nearest medoid. The set of medoids can be efficiently extracted by maximizing a submodular function. To eliminate effective poisons, we iteratively find medoids of every class in the gradient space during the training. Then, we assign every data point to the closest medoid in its class, and drop the medoids to which no other data point is assigned. We show that our Effective Poison IdentifiCation (EPIC) method can successfully eliminate effective poisons. We also prove that training on large gradient clusters of each class guarantees similar training dynamics to that of training on the full data.

Compared to existing defense strategies, our method does not require a pre-trained clean model, is not attack specific, can be applied very efficiently during the training, and provides quality guarantee for the performance of the trained model. Our extensive experiments show that our method renders state-of-the-art targeted attacks, including Gradient Matching, Bullseye Polytope, and Feature Collision ineffective, with only a slight decrease in the performance. We note that, EPIC is the only effective defense method against state-of-the-art attacks that can efficiently scale to standard deep learning pipelines. Compared to the state-of-the-art (Geiping et al., 2021a), EPIC is 6.9x faster, and maintains similarly high test accuracy and low attack success rate.

2. Related Work

2.1. Targeted Data Poisoning

Attacks on deep networks can be generally divided into triggered and triggerless attacks. Triggered or backdoor attacks augment the training data with a small set of examples that contain a trigger patch and belong to a specific target label. Models trained on the augmented data will misclassify test examples with the same patch. While early backdoor attacks were not clean-label (Chen et al., 2017; Gu et al., 2017; Liu et al., 2017; Souri et al., 2021), recent backdoor attacks produce poison examples which do not contain a visible trigger (Turner et al., 2018; Saha et al., 2019). Triggerless poisoning attacks add small adversarial perturbations to base images to make their feature representations or gradients match that of the adversarially labeled target (Shafahi et al., 2018; Zhu et al., 2019; Huang et al., 2020; Geiping et al., 2021b; Aghakhani et al., 2021). Such poisons are very similar to the base images in the input space, cannot be detected by observers, and do not require modification to targets at inference time. The most prominent poisoning attacks we test our defense against are:

Feature Collision (FC) crafts poisons by adding small perturbations to base examples so that their feature representations collide with that of the target (Shafahi et al., 2018).

Bullseye Polytope (BP) is similar to FC, but instead crafts poisons such that the target resides close to the center of their convex hull in feature space (Aghakhani et al., 2021).

Gradient Matching (GM) produces poisons by approximating this bilevel objective using "gradient alignment", encouraging gradients of the clean-label poisoned data to align with that of the adversarially labeled target (Geiping et al., 2021b). This attack is shown to be effective against data augmentation and differential privacy.

Sleeper Agent (SA) is a hidden-trigger backdoor attack that also craft poisons based on the "gradient alignment" between patched poisons and targets (Souri et al., 2021).

2.2. Defense Strategies

Commonly used data sanitization defenses work by detecting anomalies that fall outside a spherical radius in the feature space (Steinhardt et al., 2017), spectrum of the feature covariance matrix (Tran et al., 2018), or activation space (Chen et al., 2019). They may also filter points that are labeled differently from their nearest neighbors in the feature space (Peri et al., 2020). Such defense mechanisms rely on the assumption that poisons are far from the clean data points in the input or feature space. Hence, they can be easily broken by stronger data poisoning attacks that place poisoned points near one another, or by optimization methods that craft poisons to evade detection (Koh et al., 2018; Shafahi et al., 2018; Saha et al., 2019).

Robust training methods rely on strong data augmentation (Borgnia et al., 2021), apply randomized smoothing (Weber et al., 2020), use an ensemble of models for prediction (Levine & Feizi, 2020), or bound gradient magnitudes and minimize differences in orientation (Hong et al., 2020). Such methods often incur a significant performance penalty

(Jayaraman & Evans, 2019), and can even be adaptively attacked by modifying gradient signals during poison crafting (Veldanda & Garg, 2020). Other identify backdoor attacks early in training and revert their effect by gradient ascent (Li et al., 2021), use adversarial training (Madry et al., 2018; Tao et al., 2021), or create poisons during training and inject them into training batches (Geiping et al., 2021a).

Existing defense methods either drastically degrade the the model's performance (Geiping et al., 2021a), only protect the model against certain type of poisoning attack (Koh et al., 2018; Tran et al., 2018), are prohibitive for larger datasets (Geiping et al., 2021a), or do not provide any theoretical guarantee for the performance of the model (Weber et al., 2020; Levine & Feizi, 2020; Abadi et al., 2016; Geiping et al., 2021a). On the other hand, our method is fast and scalable, and successfully eliminates various poisoning attacks while allowing the model to effectively learn from the clean examples with rigorous theoretical guarantees.

3. Robust Training against Data Poisoning

Let $\mathcal{D}_c = \{(x_i, y_i)\}_{i=1}^n$ be the set of all clean training data, where $x_i \in \mathbb{R}^m$. Targeted data poisoning attacks aim to change the prediction of a target image x_t in the test set to an adversarial label y_{adv}, by modifying a fraction (usually less than 1%) of data points in the training data within an l_∞-norm ϵ-bound. We denote by $V = \{1, \cdots, n\}$, and $V_p \subset V$ the index set of the entire training data and poisoned data points, respectively. For small ϵ, this constraint enforces the perturbed images to visually look similar to the original example. Such attacks remain visually invisible to human observer and are called clean-label attacks. Targeted clean label data poisoning attacks can be formulated as the following bilevel optimization problem:

$$\min_{\delta \in \mathcal{C}} \mathcal{L}(x_t, y_{\text{adv}}, \theta(\delta)) \quad s.t. \tag{1}$$
$$\theta(\delta) = \arg\min_\theta \sum_{i \in V} \mathcal{L}(x_i + \delta_i, y_i, \theta),$$

where $\mathcal{C} = \{\delta \in \mathbb{R}^{n \times m} : \|\delta\|_\infty \leq \epsilon, \delta_i = 0 \; \forall i \notin V_p\}$ is the constraint set determining the set of valid poisons. Intuitively, the perturbations change the parameters θ of the network such that minimizing the training loss on RHS of Eq. (1) also minimizes the adversarial loss on LHS of Eq. (1).

We assume that the network is trained by minimizing the training loss $\mathcal{L}(\theta) = \sum_{i \in V} \mathcal{L}(x_i + \delta_i, y_i, \theta)$ over the entire set of clean and poisoned training examples $i \in V, \delta_i = 0 \; \forall i \notin V_p$. Applying gradient descent with learning rate η to minimize the training loss $\mathcal{L}(\theta)$, iteration τ take the form:

$$\theta_{\tau+1} = \theta_\tau - \eta \nabla \mathcal{L}(\theta_\tau). \tag{2}$$

Attack and defense assumptions. We consider a worst-case scenario, where the attacker has knowledge of the defender's training procedure (e.g. learning rate, optimization algorithm), architecture, and defense strategy, but cannot influence training, initialization, or mini-batch sampling. In transfer learning where the defender uses a pre-trained model and only trains the last layer, we assume the parameters of the pre-trained model is known to the attacker. However, the defender is not aware of the target example or the specific patch chosen by the attacker. We also assume that the defender does not have access to additional clean data points.

3.1. Motivation

For a targeted poisoning attack to be successful, the target needs to be misclassified as the adversarial class y_{adv}. Effectively, the poisons need to pull the representation of the target toward the poison class. To do so, they need to mimic the gradient of the adversarially labeled target. Formally,

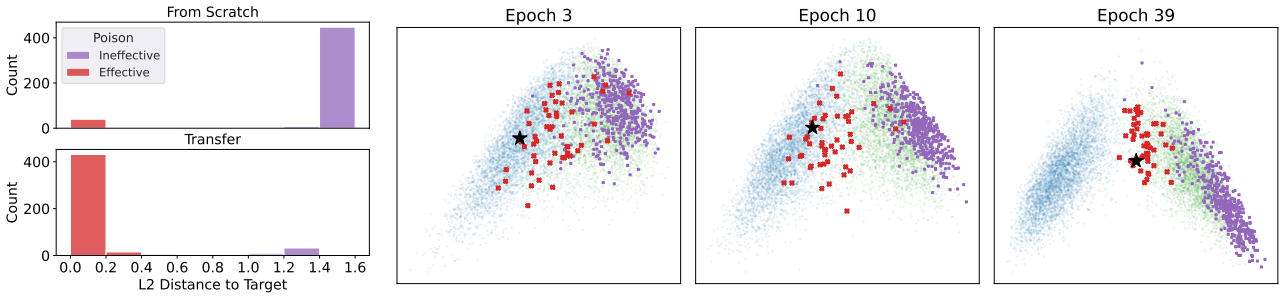
$$\nabla \mathcal{L}(x_t, y_{\text{adv}}, \theta) \approx \frac{1}{|V_p|} \sum_{i \in V_p} \nabla \mathcal{L}(x_i + \delta_i, y_i, \theta) \tag{3}$$

needs to hold for any θ encountered during training.

This is the motivation behind the poison generation in the end to end training scenario. In particular, Gradient Matching (Geiping et al., 2021b) and Sleeper Agent (Souri et al., 2021) explicitly minimize the alignment (cosine similarity) between poison and target gradient as in Eq. (3), using a clean pre-trained model. Although the poisons are generated using a pre-trained clean model, (Geiping et al., 2021a) empirically showed that the alignment between the gradient of adversarial and training loss remains large during the training. MetaPoison (Huang et al., 2020) uses a number of partially-trained models to generate poisons that minimize the adversarial loss at different stages during the training. Bullseye Polytope (Aghakhani et al., 2021) maximizes the similarity between representations of the poisons and target. In doing so, it implicitly minimizes the alignment between poison and target gradients w.r.t. the penultimate layer, which captures most of the gradient norm variation (Katharopoulos & Fleuret, 2018).

In the transfer learning scenario, the poisons are crafted to have a similar representation to that of the target. Here, a linear layer is trained on the poisoned data using the representations obtained from a pre-trained clean model. The gradient of the linear model is proportional to the representations learned by the pre-trained model. Therefore, by maximizing the similarity between the representations of the poisons and the adversarially labeled target, the attack indeed increases the alignment between their gradients.

Crucially, the better the poisons can surround the target in the gradient space, the more effective the attack becomes. This is demonstrated by the superior success rate of Bullseye Polytope (Aghakhani et al., 2021) and Convex Polytope



(a) Poison distribution

(b) Training from scratch for 40 epochs with GM poisons

Figure 1. 500 effective (red) and ineffective (purple) poisons crafted by GM and BP in from-scratch and transfer learning scenarios on CIFAR10. (a) Number of effective vs. ineffective poisons and their distance to the target in the (last layer) gradient space of a clean model; (b) Embeddings of effective (red) and ineffective (purple) poisons, and clean examples of the target (blue) and poison (green) class, projected on the first 2 principal components. Effective poisons are not examples with lowest confidence or highest loss.

(Zhu et al., 2019), compared to that of Feature Collision (Shafahi et al., 2018). While Feature Collision only optimizes the poisons to have a similar representation to that of the adversarially labeled target, Convex Polytope moves poisons until the target is inside their convex hull, and Bullseye Polytope makes further refinements to move the target away from the polytope boundary.

3.2. Not all the poisons are created equal

To successfully prevent poisoning attacks, we make the following key observation: Not all the poisoned examples are responsible for the success of the attack. We define *effective poisons* as examples that make the attack successful. That is, if the model is trained with the effective poisons, the attack will be successful even if all the other poisons are removed. In contrast, if the effective poisons are eliminated, the remaining (ineffective) poisons cannot make the attack successful. Fig. 1a shows 500 effective and ineffective poisons generated by Gradient Matching (GM) and Bulleyes Polytope (BP) in the training from scratch and transfer learning scenarios. We tried different combinations of the poisons and identified the smallest subset of poisons that is responsible for the success of the attack. We observe that indeed not all poisons are effective. While for from-scratch training only 8% of the poisons are effective, for transfer learning around 90% of the poisons are effective.

We explain the above observation as follows: not all the randomly selected examples can be modified by bounded perturbations to have a gradient that closely matches that of the target. When training from scratch, attacks can only craft a handful of effective poisons as the poisons need to match the very high-dimensional gradient of the target with bounded perturbations. On the other hand, during transfer learning, poisons are optimized to match the much lower-dimensional gradient of the target. Hence, attacks can craft a much larger number of effective poisons.

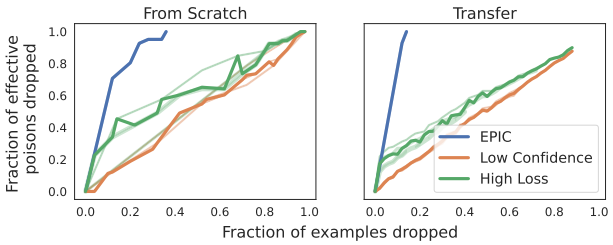


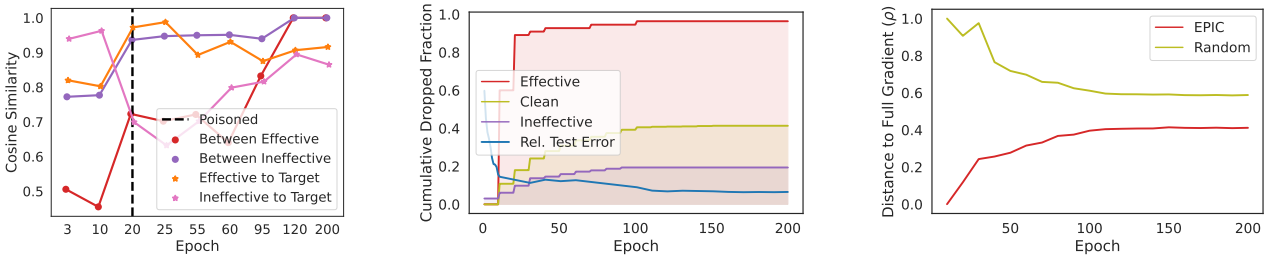
Figure 2. Fraction of effective poisons dropped vs fraction of all examples dropped during training on CIFAR10 poisoned with GM, for our method (EPIC) vs lowest-confidence and highest-loss with thresholds .25,.5/1,2 shown by transparent colors, and their average shown in opaque. Left: from-scratch. Right: transfer learning.

3.3. Effective poisons are not examples with highest loss or lowest confidence

It is important to note that effective poisons are not the data points around the decision boundary for which the model is not confident, or outliers that have a higher loss than other data points in their class. Fig. 1b shows the embedding of clean and poisoned examples of the poison and target class during training from scratch. We see that effective poisons can be within the poison or target class, or at the boundary of the classes, at different training iterations. Fig. 2 shows the fraction of effective poisons eliminated when we drop examples with highest loss or lowest confidence with various thresholds during the training. We see that dropping lowest-confidence or highest-loss examples during the training indeed removes a larger number of *clean* data points, and cannot successfully eliminate the effective poisons.

3.4. Effective poisons become isolated in gradient space

Attacks exploit the non-convex nature of the neural network loss to optimize poisons that match the target gradient. For ineffective poisons, attacks cannot successfully modify the base example with bounded perturbations to match the target gradient. This is the case where loss is relatively smooth



(a) Cosine similarity between effective/ineffective poisons and the target.

(b) Cumulative fraction of (in)effective poisons & cleans dropped by EPIC vs test error.

(c) Gradient difference of examples not dropped with EPIC vs random, with full data.

Figure 3. Training with EPIC on CIFAR10 poisoned with GM. (a) Similarity between effective poisons' gradients to each other becomes small (they get isolated) after the warmup period, (b) EPIC effectively eliminates effective poisons while dropping a smaller fraction of clean examples, (c) EPIC preserves main gradient components, hence remaining examples have a closer gradient to that of the full data, compared to random subsets of the same size. Thus, EPIC preserves the training dynamics.

in a ball of radius ϵ around the base. Thus, for ineffective poisons attacks can only increase the alignment between the gradients of ineffective poisons with that of the target to some extent. In doing so, the similarity between ineffective poisons' gradients becomes larger. Hence, they form larger gradient clusters in the poison class, as shown by Fig. 1b.

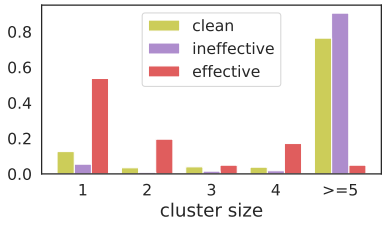
On the other hand, effective poisons can be modified under bounded perturbations to match the target gradient. This is the case where there are sharp regions in a ball of radius ϵ around a base example. Here, the base can be perturbed and taken to such sharp regions, and its gradient can be further optimized there to match the target gradient. During the training on the poisoned data, the gradients of effective poisons move far away from their class and get close to the target. But, they each have a *different trajectory* (starting from difference base examples) for interpolating between their base and the target gradients. These trajectories are neither similar to each other (as they start from different bases), nor similar to other examples in the base class (as they end up matching target's gradient in another class). Fig. 3a shows that while the similarity between gradients of effective poisons and target increases during the training, the gradient of effective poisons is very different from each other after a few epochs of training, and before the model gets poisoned. Hence, effective poisons' gradients become isolated in the gradient space, early in training. Such isolated points in low-density gradient regions can be best identified by proximity-based methods, such as k-medoids, as we discuss next.

3.5. Eliminating the effective poisons

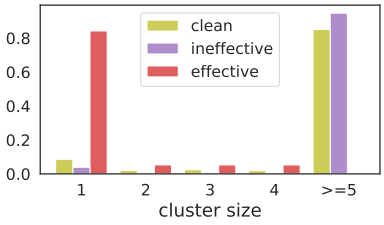
To prevent data poisoning while maintaining the generalization performance of the network, we aim at identifying and removing the effective poisons. To do so, our key idea is to drop data points that have a different gradient compared to other examples in their class, i.e., are isolated in the gradient space during training. As we discuss next, dropping such points effectively eliminates the majority of poisoning attacks with only a slight impact on the gradient of the full training loss. By preserving the important gradient components, we guarantee similar training dynamics and convergence to a close neighborhood of the solution obtained by training on full data. To find the effective poisons that do not have a similar gradient to the other data points in their class, we train the model for a few epochs (warmup). Then, we iteratively find and drop the isolated points in low-density gradient regions. To do so, we find the

Algorithm 1 Effective Poison Identification (EPIC)

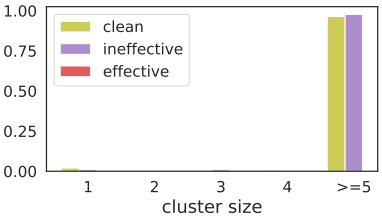
Input: Training data indexed by V, submodular facility location function F, loss function $\mathcal{L}(\cdot)$, warmup iterations K, poison drop interval T, number of medoids k.
Output: Subset $S \subseteq V$
Train the network for K epochs on V.
for every T epochs **do**
  for examples V_c in class $c \in [C]$ **do**
    Initialize $S \leftarrow \emptyset, Z \leftarrow \emptyset$
    while $|S| < k/C$ **do**
      $j \in \arg\max_{e \in V_c \setminus S} F(e|S)$
      $S = S \cup \{j\}$
    end while
    for $j = 1$ **to** $|S|$ **do**
      $\gamma_j = \sum_{i \in V_c} \mathbb{I}[j = \text{argmin}_{s \in S} ||\nabla^L_f \mathcal{L}_i(\theta) - \nabla^L_f \mathcal{L}_s(\theta)||]$
      if $\gamma_i == 1$ **then**
        $Z = Z \cup \{j\}$
      end if
    end for
    $V = V \setminus Z$
  end for
  Train on V for T epochs.
end for



(a) Epoch 10. Most of the effective poisons are isolated as clusters of size 1.

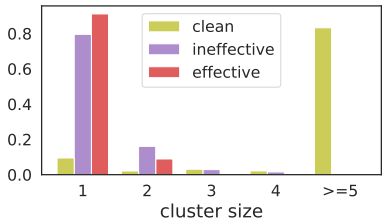


(b) Epoch 20. Most of remaining effective poisons become isolated.

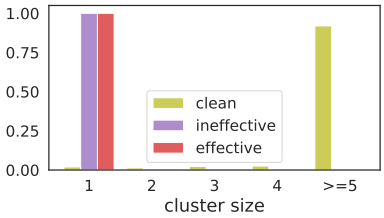


(c) Epoch 80. Clean examples form larger gradient clusters during the training.

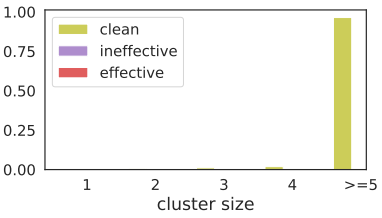
Figure 4. Fraction of clean vs Gradient Matching poisons in gradient clusters of different sizes, during from-scratch learning with EPIC for 200 epochs. Effective poisons become isolated during training and can be iteratively eliminated by EPIC.



(a) Epoch 1. Most of the effective poisons are isolated as clusters of size 1.



(b) Epoch 2. The remaining effective poisons become isolated as clusters of size 1.



(c) Epoch 20. Clean examples form larger gradient clusters during the training.

Figure 5. Fraction of clean vs Bullseye Polytope poisons in gradient clusters of different sizes, during transfer learning with EPIC for 40 epochs. Effective poisons become isolated during training and can be iteratively eliminated by EPIC.

medoids—the most centrally located data points—of each class, in the gradient space. Then, we assign every data point to its closest medoid, and drop the medoids to which no other data point is assigned. In our experiments, we show that selecting small subsets (10%-30% of data) of medoids at every iteration can successfully prevent various types of data poisoning attacks. Fig. 4, 5 show that during training from scratch or transfer learning, effective poisons are isolated medoids of the gradient space. Hence, our strategy successfully identifies the majority of the effective poisons in both from scratch and transfer learning settings, while only dropping a small number of clean examples (Fig. 2,3b).

The set of medoids of a class minimizes the average gradient dissimilarity to all the other data points in the class. For a specific value of k, the set of k-medoids can be found as:

$$S_\tau^* \in \arg\min_{\substack{S\subseteq V \\ |S|\leq k}} \sum_{i\in V} \min_{j\in S} \|\nabla\mathcal{L}_i(\theta_\tau) - \nabla\mathcal{L}_j(\theta_\tau)\|_2, \quad (4)$$

where $\mathcal{L}_i(\theta) = \mathcal{L}(x_i, y_i, \theta)$ is the loss associated with (potentially poisoned) training example $i \in V$. The minimization problem (4) is NP-hard. However, it can be turned into maximizing a submodular[2] facility location objective:

$$S_\tau^* \in \arg\min_{S\subseteq V} |S|, \quad s.t. \quad (5)$$
$$F(S) = \sum_{i\in V} \max_{j\in S} c_0 - \|\nabla\mathcal{L}_i(\theta_\tau) - \nabla\mathcal{L}_j(\theta_\tau)\|_2,$$

[2]A set function $F: 2^V \to \mathbb{R}^+$ is submodular if $F(S\cup\{e\}) - F(S) \geq F(T\cup\{e\}) - F(T)$, for any $S \subseteq T \subseteq V$ and $e \in V\setminus T$. F is *monotone* if $F(e|S) \geq 0$ for any $e \in V\setminus S$ and $S \subseteq V$.

where c_0 is a constant satisfying $c_0 \geq \|\nabla\mathcal{L}_i(\theta_\tau) - \nabla\mathcal{L}_j(\theta_\tau)\|_2$, for all $i, j \in V$. For maximizing a monotone submodular function, the greedy algorithm provides a $(1-1/e)$ approximation guarantee (Wolsey, 1982). The greedy algorithm starts with the empty set $S_0 = \emptyset$, and at each iteration t, it chooses an element $e \in V$ that maximizes the marginal utility $F(e|S_t) = F(S_t \cup \{e\}) - F(S_t)$. Formally, $S_t = S_{t-1} \cup \{\arg\max_{e\in V} F(e|S_{t-1})\}$. The computational complexity of the greedy algorithm is $\mathcal{O}(nk)$. However, its complexity can be reduced to $\mathcal{O}(|V|)$ using stochastic methods (Mirzasoleiman et al., 2015), and can be further improved using lazy evaluation (Minoux, 1978) and distributed implementations (Mirzasoleiman et al., 2013).

During the training, the gradients of data points change at every iteration. To identify the effective poisons, we need to update the gradient medoids iteratively. The gradient vectors can be very high-dimensional, in particular when training from scratch. To efficiently solve Eq. (5), we rely on a recent results showing that the variation of the gradient norms is mostly captured by the gradient of the loss w.r.t. the input to the last layer (Katharopoulos & Fleuret, 2018). Hence, upper-bound on the normed difference between pairwise gradient dissimilarities can be efficiently calculated:

$$\|\nabla\mathcal{L}_i(\theta_\tau) - \nabla\mathcal{L}_j(\theta_\tau)\|_2 \leq \mathcal{O}\big(\|\nabla_f^L\mathcal{L}_i(\theta_\tau) - \nabla_f^L\mathcal{L}_j(\theta_\tau)\|_2\big)$$

where $\nabla_f^L\mathcal{L}_i$ is gradient of the loss function $\mathcal{L}$ w.r.t. the input to the last layer L for data point i. The above upper-bound is marginally more expensive to calculate than loss. Hence, upper-bounds on the gradient dissimilarities can be

efficiently calculated. Alg. 1 illustrates the pseudocode.

Iteratively eliminating the isolated medoids during the training allows to successfully prevent various types of attacks. At the same time, as EPIC drops scattered gradient outliers and doesn't skew larger (main) gradient clusters, it only introduces a small limited error (ρ) on the full training gradient. Fig. 3c shows that the gradient of the remaining training examples during training with EPIC is much closer to the full training gradient, compared to that of random subsets of the same size. Theorem 3.1 leverage this idea to upper-bound the difference between loss of model trained with EPIC and the model trained on the full data, at every step of training. This ensures similar training (loss) dynamics to that of training on the full data, and allows the network to obtain a similar generalization performance.

Theorem 3.1. *Assume that the loss function $\mathcal{L}(\theta)$ is μ-PL* on a set Θ, i.e., $\frac{1}{2}\|\nabla\mathcal{L}(\theta)\|^2 \geq \mu\mathcal{L}(\theta), \forall\theta \in \Theta$. Assume ρ is the maximum change in the gradient norm due to dropping points. Then, applying gradient descent with a constant learning rate η has similar training dynamics to that of training on the full data. I.e.,*

$$\mathcal{L}(\theta_t) \leq (1-\eta\mu)^t\mathcal{L}(\theta_0) - \frac{1}{2\mu}(\rho^2 - 2\rho\nabla_{\max}). \quad (6)$$

The proof can be found in the Appendix.

Compared to existing defense strategies, our method do not require a pre-trained clean model, is not attack specific, can be applied very efficiently during the training, and provides quality guarantee for the performance of the trained model.

3.6. Adaptive attacks

Adaptive attacks can generate more powerful poisons by taking into account the knowledge of the particular defense mechanisms in place. For example, Gradient Matching (Geiping et al., 2021b) and Sleeper Agent (Souri et al., 2021) include augmented data points that are transformed with e.g. crop and flip in addition to the original ones during poison crafting in Eq. (3). In doing so, the attack can successfully poison the model even when data augmentations like crops and flips are applied to the learning pipeline. For adaptive attacks to be successful in presence of EPIC, they need to generate clusters of effective poisons. To do so, the attacker may craft poisons with similar gradient trajectories during the training, or optimize the choice of base examples that result in clustered poison gradients. However, crafting poisons with similar gradient trajectories during the training makes the poison optimization prohibitive and may result in less effective attacks. While selecting similar base images does not lead to clustered effective poisons due to non-convex nature of loss, optimizing the choice of base examples worth further investigation in future work.

Next, we show that our method achieves a superior performance compared to existing defense techniques.

4. Experiments

4.1. Against Data Poisoning Attacks

We evaluate the effectiveness of defense methods against data poisoning attacks, during *from-scratch training*, *transfer learning* and *finetuning*. For our evaluation, we use the standardized data poisoning benchmark (Schwarzschild et al., 2020), with 200 training epochs, starting learning rate of 0.1 and decaying factor of 10 at epochs 100, 150. As several defense methods are prohibitive to be applied to standard learning pipeline with 200 epochs, we also consider a *proxy* setup used by (Geiping et al., 2021a) which trains for only 40 epochs, with a starting learning rate of 0.1 and decaying factor of 10 at epochs 25, 35.

4.1.1. From-Scratch Training

We model the from-scratch training experiments based on the benchmark setting (Schwarzschild et al., 2020). For our attack model, we select 1% of the training examples as poisons, which are perturbed within the l_∞ ball of radius $\epsilon = 8/255$. The defender initializes a model based on a random seed and trains on the poisoned dataset using SGD. To maximize reproducibility, we only use publicly available poisoned datasets generated by authors of the attacks.

Unless otherwise specified, we augment training images with random horizontal flip followed by random cropping, and per-channel normalization. For our proposed defense, we run EPIC with $T=2$ in a 40-epoch training pipeline, or $T=10$ in a 200-epoch training pipeline.

Warmup The more medoids we select for each class, the longer warmup period (K) we need for EPIC. In the experiments, we set $K = 10$ for EPIC-0.1, $K = 20$ for EPIC-0.2 and $K = 30$ for EPIC-0.3.

Gradient Matching (GM) GM is currently the state-of-the-art among data poisoning attacks for from-scratch training (Schwarzschild et al., 2020). (Geiping et al., 2021b) shows it significantly outperform the other effective attack, MetaPoison (Huang et al., 2020). We test 100 datasets provided by the authors . The datasets were generated based on the 100 preset benchmark settings, each with 500 specific base and a target image. We follow the training hyperparameters specified by the benchmark to train ResNet-18 from scratch with 128 examples per mini-batch. Table 1 shows that the average attack success rate of GM on these 100 datasets is 45% and the average test accuracy is 94.95%. We see that our proposed defense, EPIC, is able to successfully drop the the average attack success rate to only 1% while keeping the test accuracy above 90%.

Bullseye Polytope (BP) Schwarzschild et al. (2020) shows the superiority of BP attack in traning VGG models (Simonyan & Zisserman, 2014) from scratch on Tiny-

Table 1. Average attack success rate and validation accuracy for EPIC against various data poisoning attacks (200-epoch pipeline).

ATTACK	SENARIO	UNDEFENDED		DEFENDED	
		ATT SUCC.↑	TEST ACC.↑	ATT SUCC.↓	TEST ACC.↑
GRADIENT MATCHING	FROM-SCRATCH	45%	94.95%	1%	90.26%
SLEEPER AGENT (BACKDOOR)	FROM-SCRATCH	78.54%	94.42%	11.55%	88.28%
BULLSEYE POLYTOPE	TRANSFER	86%	94.69%	1%	94.80%
FEATURE COLLISION	TRANSFER	40%	94.68%	0%	94.81%
BULLSEYE POLYTOPE	FINETUNE	80%	92.24%	0%	92.38%

Table 2. Avg. Poison Success versus validation accuracy for various defenses against the gradient matching attack of (Geiping et al., 2021b) in the from-scratch setting. The proposed Robust Training Against Data Poisoning is listed as EPIC.

EPOCH	DEFENSE	ATTACK SUCC.↓	TEST ACC.↑	TIME(HR:MIN)
40	NONE	25%	92.48%	00:15
40	DEEPKNN (PERI ET AL., 2020)	21%	91.86%	02:25
40	SPECTRAL SIGNATURES (TRAN ET AL., 2018)	17%	90.13%	00:40
40	ACTIVATION CLUSTERING (CHEN ET AL., 2019)	9%	84.20%	00:31
40	DIFF. PRIV. SGD (HONG ET AL., 2020)	2%	70.34%	00:16
40	ADV. POISONING-0.25 (GEIPING ET AL., 2021A)	4%	91.48%	01:53
40	ADV. POISONING-0.5 (GEIPING ET AL., 2021A)	1%	90.67%	02:02
40	ADV. POISONING-0.75 (GEIPING ET AL., 2021A)	0%	87.97%	02:26
40	EPIC-0.1 (PROPOSED)	2.7%±0.6%	90.92%±0.26%	00:22
40	EPIC-0.2 (PROPOSED)	1.3%±0.6%	88.95%±0.08%	00:19
40	EPIC-0.3 (PROPOSED)	1.0%±0.0%	87.03%±0.11%	00:17
200	NONE	45%	94.95%	01:18
200	SPECTRAL SIGNATURES (TRAN ET AL., 2018)	10%	92.99%	03:22
200	ACTIVATION CLUSTERING (CHEN ET AL., 2019)	11%	90.88%	02:33
200	DIFF. PRIV. SGD (HONG ET AL., 2020)	2%	80.71%	01:23
200	EPIC-0.1 (PROPOSED)	2.3%±0.6%	92.50%±0.03%	01:50
200	EPIC-0.2 (PROPOSED)	1.0%±1.0%	89.71%±0.06%	01:35
200	EPIC-0.3 (PROPOSED)	0.7%±0.6%	87.05%±0.05%	01:28

Table 3. Defending against the BP attack on TinyImageNet while training from scratch. Our method (EPIC) can train more accurate models than the SOTA defense (AP) without increasing the success rate of poisoning attacks, and is more scalable.

DEFENSE	ATTACK SUCC.↓	TEST ACC.↑	TIME↓
NONE	40%	61.80%	1HR
AP-0.5	0%	53.54%	7HRS
EPIC-0.2	0%	57.50%	1HR

ImageNet, a subset of the ILSVRC2012 classification dataset (Deng et al., 2009). We tested the first 10 example datasets provided by the benchmark, and observed attack success rate of 40%. Training with EPIC drops the attack success rate significantly to 0%, as shown in Table 3.

Sleeper Agent (SA) SA is the only backdoor attack that can achieve higher than single-digit success rate on CIFAR-10 in the from-scratch setting. We generated 20 poisoned datasets with SA ($\epsilon = 16$) using the source-target class pairs in the first 20 CIFAR-10 benchmark settings. For backdoor attacks, we evaluate the attack success rate over 1000 patched test images. The average attack success rate is 78.54% without defenses and 11.55% with EPIC.

4.1.2. TRANSFER LEARNING

Here, we use the 40-epoch pipeline of (Geiping et al., 2021a) to evaluate defense methods. The same pretrained model is used for generating the attack and for transfer learning onto the defender. Similar to the from-scratch setting, the attacker can modify 1% of 50000 training examples in CIFAR10 with $\epsilon = 8$. The linear layer (classifier) of the pretrained model is then re-initialized and trained with the poisoned dataset with all the other layers (feature extractor) fixed during the training. This white-box setup allows attacks to produce stronger poisons. Here, we apply EPIC with $T = 1$.

Bullseye Polytope (BP) According to (Schwarzschild et al., 2020), Bullseye Polytope attack (Aghakhani et al., 2021) has the highest average attack success rate in the white-box setting. When evaluated on all 100 benchmark setups, BP succeeded in 86 of them. Table 1 shows that EPIC could successfully drop the attack success rate to only 1% while even increasing the test accuracy of the model.

Feature Collision (FC) As imposing the l_∞ constraint $\epsilon = 8$ will greatly reduce the power of Feature Collison attack (Schwarzschild et al., 2020), we keep the l_2 regu-

larization term in their original optimization objective to impose a soft rather than hard constraint on the l_∞ perturbation. We generate 20 poisoned datasets using the first 20 benchmark setups (indexed from 0 to 19). With the default seed used by the benchmark, the attack success rate of these 20 datasets generated by FC is 40% before and 0% after we apply our EPIC defense, as shown in Table 1.

4.1.3. FINETUNING

We also consider the finetuning scenario in which the classifier is re-initialized and the feature extractor is not fixed during the training. We follow the same setup in (Geiping et al., 2021a), test 20 datasets poisoned with BP, and report the result in Table 1. Again, EPIC successfully prevents all attacks in this scenario without decreasing the test accuracy.

4.2. Comparison to SOTA Defenses against GM

Table 2 compares the effectiveness of our model with existing defense methods against the state-of-the-art GM attack, in both 200 epoch and 40 epoch training scenarios. We see that EPIC can successfully drop the success rate of GM while allowing the model to achieve a superior performance. We note that unlike existing defense methods, our method is easily scalable to standard deep learning pipelines.

Scalability As many defense methods (Geiping et al., 2021a; Peri et al., 2020) are prohibitive when applied to the standard 200-epoch pipeline under time or space constraints, they are evaluated using a 40-epoch pipeline. However, as Table 2 shows, training a model on the same poisoned datasets for more epochs increases attack success rate. Therefore, defenses that are successful within 40 epochs are not guaranteed to have the same effectiveness when models are trained for longer. On the contrary, our proposed defense requires nearly no extra time compared to normal training. Time spent on running EPIC every few epochs is usually well compensated by training time saved every epoch on the examples we drop. Table 2 includes the time for each defense on CIFAR10 poisoned with GM, and Table 3 compares EPIC with AP on TinyImagenet poisoned with BP. We report the wall-clock time of training a model with each defense on single NVIDIA A40 GPU with 4 workers. We see that EPIC effectively reduces various attacks' success rate while having substantially faster run time.

Strength of Defense Due to computational constraints and the scalability problem mentioned above, we only scale the two general adversarial training methods, Adversarial Training (Madry et al., 2018) and DP-SGD (Hong et al., 2020) to the standard 200-epoch training pipeline. According to Table 2, 3 and Fig. 6, our method provides the best tradeoffs between the defended attack success rate and the overall test accuracy. Adversarial Poisoning (Geiping et al., 2021a) can give equally good tradeoffs but requires 6x training time. Other defenses either cannot guarantee a low attack success rate or have a high computation cost.

Table 4. Comparison of avg. poison accuracy, validation accuracy and time against the strongest attack GM (Geiping et al., 2021b) with $\epsilon = 16$ in the from-scratch setting for 40 epochs. Our proposed defense is listed as EPIC.

DEFENSE	ATTACK SUCC.↓	TEST ACC.↑
NONE	90%	92.01%
AP-0.25	35%	91.21%
AP-0.5	10%	90.58%
AP-0.75	0%	87.97%
EPIC-0.1	10%	91.15%
EPIC-0.2	0%	89.07%

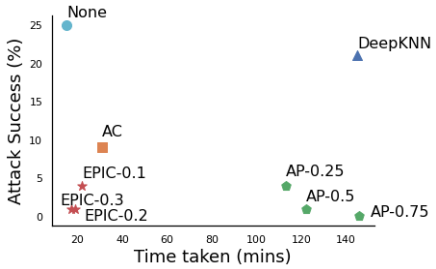


Figure 6. Attack success rate vs. running time of different defenses, for GM attack on CIFAR-10 with the 40 Epochs pipeline.

4.3. Comparison under Larger Perturbations

Attacks usually have higher success rates when allowed to perturb the base images within a larger ϵ constraint (Schwarzschild et al., 2020). We generate 20 poisoned datasets with a larger $\epsilon = 16$ with GM. We use the default 20 seeds used in (Geiping et al., 2021b) to sample 500 base and 1 target images from CIFAR10. Table 4 shows that for larger ϵ, EPIC achieves a superior performance compared to the strongest baseline, adversarial poisoning.

5. Conclusion

We proposed an efficient defense mechanism against various data poisoning attacks. We showed that under bounded perturbations, only a small number of poisons can be optimized to have a gradient that is close enough to that of the target and make the attack successful. Such examples move away from their original class and get isolated in the gradient space. Consequently, we showed that training on large gradient clusters of each class successfully eliminates the effective poisons, and guarantees similar training dynamics to that of training on the full data. Our experiments showed that our method significantly decreases the success rate of the state-of-the-art targeted attacks, including Gradient Matching, Bullseye Polytope. We note that our method is the only effective defense against strong poisoning attacks, which easily scales to standard deep learning pipelines.

Acknowledgements

This research was supported in part by Cisco Systems and UCLA-Amazon Science Hub for Humanity and AI.

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, pp. 308–318, 2016.

Aghakhani, H., Meng, D., Wang, Y.-X., Kruegel, C., and Vigna, G. Bullseye polytope: A scalable clean-label poisoning attack with improved transferability. In 2021 IEEE European Symposium on Security and Privacy (EuroS&P), pp. 159–178. IEEE, 2021.

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. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3855–3859. IEEE, 2021.

Boyd, S., Boyd, S. P., and Vandenberghe, L. Convex optimization. Cambridge university press, 2004.

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, 2019.

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

Cretu, G. F., Stavrou, A., Locasto, M. E., Stolfo, S. J., and Keromytis, A. D. Casting out demons: Sanitizing training data for anomaly sensors. In 2008 IEEE Symposium on Security and Privacy (sp 2008), pp. 81–95. IEEE, 2008.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pp. 248–255. Ieee, 2009.

Fort, S., Dziugaite, G. K., Paul, M., Kharaghani, S., Roy, D. M., and Ganguli, S. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. Advances in Neural Information Processing Systems, 33: 5850–5861, 2020.

Geiping, J., Fowl, L., Somepalli, G., Goldblum, M., Moeller, M., and Goldstein, T. What doesn't kill you makes you robust (er): Adversarial training against poisons and backdoors. 2021a.

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, 2021b. URL https://openreview.net/forum?id=01olnfLIbD.

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

Hong, S., Chandrasekaran, V., Kaya, Y., Dumitraş, T., and Papernot, N. On the effectiveness of mitigating data poisoning attacks with gradient shaping. arXiv preprint arXiv:2002.11497, 2020.

Huang, W. R., Geiping, J., Fowl, L., Taylor, G., and Goldstein, T. Metapoison: Practical general-purpose clean-label data poisoning. Advances in Neural Information Processing Systems, 33, 2020.

Jayaraman, B. and Evans, D. Evaluating differentially private machine learning in practice. In 28th {USENIX} Security Symposium ({USENIX} Security 19), pp. 1895–1912, 2019.

Katharopoulos, A. and Fleuret, F. Not all samples are created equal: Deep learning with importance sampling. In International Conference on Machine Learning, pp. 2525–2534, 2018.

Koh, P. W., Steinhardt, J., and Liang, P. Stronger data poisoning attacks break data sanitization defenses. arXiv preprint arXiv:1811.00741, 2018.

Levine, A. and Feizi, S. Deep partition aggregation: Provable defenses against general poisoning attacks. In International Conference on Learning Representations, 2020.

Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B., and Ma, X. Anti-backdoor learning: Training clean models on poisoned data. Advances in Neural Information Processing Systems, 34, 2021.

Liu, C., Zhu, L., and Belkin, M. Toward a theory of optimization for over-parameterized systems of non-linear equations: the lessons of deep learning. arXiv preprint arXiv:2003.00307, 2020.

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

Ma, Y., Zhu, X. Z., and Hsu, J. Data poisoning against differentially-private learners: Attacks and defenses. In International Joint Conference on Artificial Intelligence, 2019.

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, 2018.

Minoux, M. Accelerated greedy algorithms for maximizing submodular set functions. In Optimization techniques, pp. 234–243. Springer, 1978.

Mirzasoleiman, B., Karbasi, A., Sarkar, R., and Krause, A. Distributed submodular maximization: Identifying representative elements in massive data. In Advances in Neural Information Processing Systems, pp. 2049–2057, 2013.

Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrák, J., and Krause, A. Lazier than lazy greedy. In Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015.

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 European Conference on Computer Vision, pp. 55–70. Springer, 2020.

Saha, A., Subramanya, A., and Pirsiavash, H. Hidden trigger backdoor attacks, 2019.

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 preprint arXiv:2006.12557, 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, 2018.

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

Souri, H., Goldblum, M., Fowl, L., Chellappa, R., and Goldstein, T. Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. arXiv preprint arXiv:2106.08970, 2021.

Steinhardt, J., Koh, P. W., and Liang, P. Certified defenses for data poisoning attacks, 2017.

Tao, L., Feng, L., Yi, J., Huang, S.-J., and Chen, S. Better safe than sorry: Preventing delusive adversaries with adversarial training. Advances in Neural Information Processing Systems, 34, 2021.

Tran, B., Li, J., and Madry, A. Spectral signatures in backdoor attacks. In Advances in Neural Information Processing Systems, pp. 8000–8010, 2018.

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

Veldanda, A. and Garg, S. On evaluating neural network backdoor defenses. arXiv preprint arXiv:2010.12186, 2020.

Weber, M., Xu, X., Karlaš, B., Zhang, C., and Li, B. Rab: Provable robustness against backdoor attacks. arXiv preprint arXiv:2003.08904, 2020.

Wolsey, L. A. An analysis of the greedy algorithm for the submodular set covering problem. Combinatorica, 2(4): 385–393, 1982.

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, 2019.

A. Proof of Theorem 3.1

A loss function $\mathcal{L}(w)$ is considered μ -PL on a set $\mathcal{S}$, if the following holds:

$$\frac{1}{2}\|\mathbf{g}\|^2 \geq \mu\left(\mathcal{L}(w) - \mathcal{L}\left(w_*\right)\right), \forall w \in \mathcal{S} \tag{7}$$

where w_* is a global minimizer. When additionally $\mathcal{L}\left(w_*\right) = 0$, the μ-PL condition is equivalent to the μ-PL* condition

$$\frac{1}{2}\|\mathbf{g}\|^2 \geq \mu\mathcal{L}(w), \forall w \in \mathcal{S}. \tag{8}$$

For Lipschitz continuous $\mathbf{g}$ and μ-PL condition, gradient descent on the entire dataset yields

$$\mathcal{L}(w_{t+1}) - \mathcal{L}(w_t) \leq -\frac{\eta}{2}\|\mathbf{g}_t\|^2 \leq -\eta\mu\mathcal{L}(w_t), \tag{9}$$

and,

$$\mathcal{L}(w_t) \leq (1-\eta\mu)^t\mathcal{L}(w_0), \tag{10}$$

which was shown in (Liu et al., 2020). We build upon this result.

For the subset we have

$$\mathcal{L}(w_{t+1}) - \mathcal{L}(w_t) \leq -\frac{\eta}{2}\|\mathbf{g}_t^S\|^2 \tag{11}$$

By substituting Eq. (**??**) we have.

$$\leq -\frac{\eta}{2}(\|\mathbf{g}_t\| - \rho)^2 \tag{12}$$

$$= -\frac{\eta}{2}(\|\mathbf{g}_t\|^2 + \rho^2 - 2\rho\|\mathbf{g}_t\|) \tag{13}$$

$$\leq -\frac{\eta}{2}(\|\mathbf{g}_t\|^2 + \rho^2 - 2\rho\nabla_{\max}) \tag{14}$$

$$\leq -\frac{\eta}{2}(2\mu\mathcal{L}(w_t) + \rho^2 - 2\rho\nabla_{\max}) \tag{15}$$

where we can upper bound the norm of $\mathbf{g}_t$ in Eq. (13) by a constant $\nabla_{\max}$. And Eq. (15) follows from the μ-PL condition from Eq. (7). While loss is very non-convex during the first part of training, it becomes nearly-convex afterwards (Fort et al., 2020). EPIc starts dropping points after a few epochs of training, where Lipschitzness, μ-PL condition, and norm-bounded gradients are likely to hold. In Eq. (10), LHS directly results from Lipschitzness (Boyd et al., 2004).

Hence,

$$\mathcal{L}(w_{t+1}) \leq (1-\eta\mu)\mathcal{L}(w_t) - \frac{\eta}{2}(\rho^2 - 2\rho\nabla_{\max}) \tag{16}$$

Since, $\sum_{j=0}^{k}(1-\eta\mu)^j \leq \frac{1}{\eta\mu}$, for a constant learning rate η we get

$$\mathcal{L}(w_{t+1}) \leq (1-\eta\mu)^{t+1}\mathcal{L}(w_0) - \frac{1}{2\mu}(\rho^2 - 2\rho\nabla_{\max}) \tag{17}$$