Poisons that are learned faster are more effective

Pedro Sandoval-Segura[1] Vasu Singla[1] Liam Fowl[1] Jonas Geiping[1] Micah Goldblum[2]
David Jacobs[1] Tom Goldstein[1]
[1]University of Maryland [2]New York University
{psando, vsingla, lfowl, jgeiping, dwj, tomg}@umd.edu goldblum@nyu.edu

Abstract

Imperceptible poisoning attacks on entire datasets have recently been touted as methods for protecting data privacy. However, among a number of defenses preventing the practical use of these techniques, early-stopping stands out as a simple, yet effective defense. To gauge poisons' vulnerability to early-stopping, we benchmark error-minimizing, error-maximizing, and synthetic poisons in terms of peak test accuracy over 100 epochs and make a number of surprising observations. First, we find that poisons that reach a low training loss faster have lower peak test accuracy. Second, we find that a current state-of-the-art error-maximizing poison is 7× less effective when poison training is stopped at epoch 8. Third, we find that stronger, more transferable adversarial attacks do not make stronger poisons. We advocate for evaluating poisons in terms of peak test accuracy.

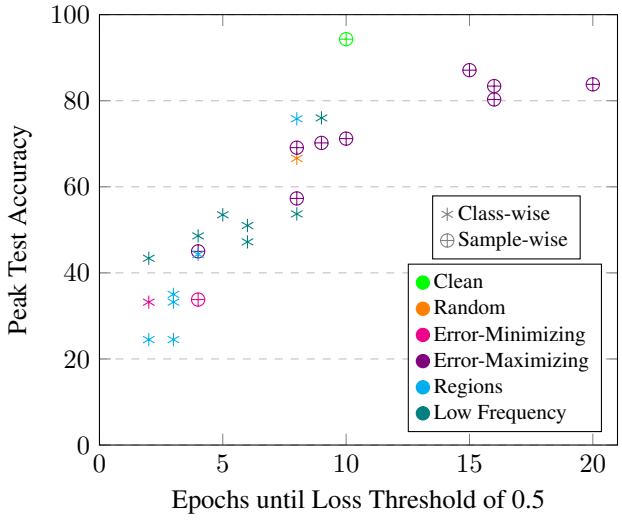

Figure 1. Poisons which are learned faster have lower peak test accuracy on CIFAR-10. For every poison, we log both the epoch after which the training loss dips below 0.5 and the peak test accuracy over 100 epochs.

1. Introduction

The threat of maliciously perturbed data being unexpectedly included in a dataset is high due to automated web scraping. Web scraping is increasingly used to construct large datasets, necessary for training groundbreaking deep learning models [4, 23]. The modified data, called a *poison*, can induce malicious behavior in a deep neural network trained on this data by an unwitting practitioner [11]. Data poisoning attacks are tailored for different kinds of erroneous model behavior: backdoor attacks ensure prespecified input features cause inaccurate output [6, 13], feature-collision attacks cause a particular target example to be misclassified as a base class [24, 33], and availability attacks aim to degrade overall test performance [2, 3, 8, 17]. In this work, we focus on *availability attacks* - a flavor of data poisoning attack where the poisoner tries to induce poor performance for the victim network on the clean distribution. Often, for availability attacks, the poisoner is allowed to perturb the entire dataset, or a large portion of it. Interestingly, this form of data poisoning has recently been pitched as a means of protecting data privacy. For example, a number of practical experiments have demonstrated that adversarial poisons can be used to reduce the accuracy of protected classes within a facial recognition dataset [8, 9, 17].

But no approach or application of data poisoning is useful if it can be circumvented easily. That is, if the victim which trains on a poison can still achieve good performance on the clean distribution, then the poison is ineffective. Many defenses have been shown to reduce a poison's effectiveness: adversarial training [10, 17, 29, 31], early-stopping [17, 30], and diluting the poison with clean data [8, 17]. In this work, we focus on the defense of early-stopping. Using early-stopping, a practitioner chooses among the best models in terms of validation accuracy on a holdout set, over all models seen during training. In comparison to other defenses, this defense is essentially *passive*. Even if the practitioner training the model is not aware of the attack (and

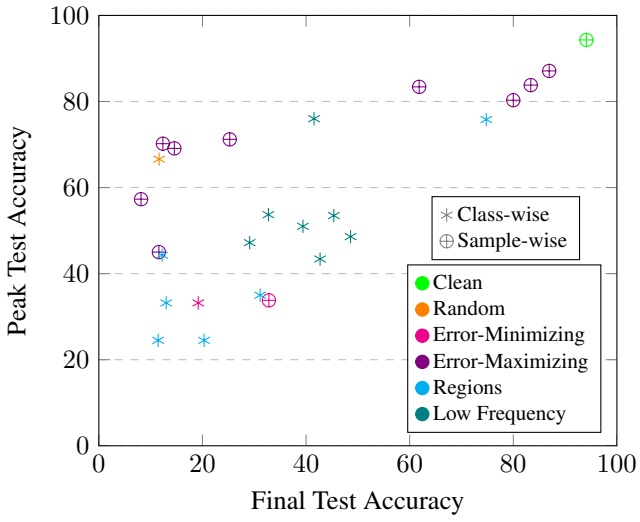

Figure 2. There are a number of effective poisons for which final test accuracy doesn't correlate well with peak test accuracy. Effective poisons have both low peak test accuracy and low final test accuracy.

hence would not consider actively using defenses), they would still select the best model by peak validation accuracy. The defense does not require writing new training logic, designing a new architecture, or collecting new data.

Overall, our contributions can be summarized as follows:

- We find that the early epochs of poison training are a good indicator of whether a poison will reach a high test accuracy. There exists a correlation between peak test accuracy and the number of epochs before a threshold loss is reached.
- We demonstrate that final test accuracy does not correlate well with peak test accuracy. We advocate for evaluating poisons based on *peak test accuracy* over final test accuracy.
- We find that adversarial attacks which are stronger or more transferable do not always lead to more effective error-maximizing poisons.

2. Related Work

To conduct availability attacks on neural networks, recent works have modified data to explicitly cause gradient vanishing [25] or have minimized the loss with respect to the input image [17]. More recently, strong adversarial attacks, which perturb clean data by maximizing the loss with respect to the input image, have been shown to be the most successful approach thus far [8]. But success has largely been defined by the final test set accuracy of the poisoned network. In this work, we show that final test set accuracy does not tell the whole story by investigating early-stopping as it is the simplest and most practical mitigation. A thorough overview of data poisoning methods can be found in [11].

2.1. Error-Minimizing Noise

Dubbed *unlearnable examples*, images perturbed with error-minimizing noises are a surprisingly good data poisoning attack. A ResNet-18 network trained on a CIFAR-10 [18] sample-wise error-minimizing poison achieves 19.9% final test accuracy, while the class-wise variant achieves 16.4% final test accuracy after 60 epochs of training [17]. The discovery of unlearnable examples also included analysis of error-maximizing noise, but results demonstrated that error-maximizing noise was not as effective in degrading test set performance of a classifier.

2.2. Error-Maximizing Noise

A number of works have shown that adversarial examples can fool DNNs at test time [5, 12, 19, 21, 28] through the use of error-maximizing perturbations. Adversarial examples are crafted by optimizing an objective which seeks to maximize the network's prediction error. While many adversarial attacks operate under an additive threat model, where the adversary is allowed to add an imperceptible perturbation vector to a clean image, spatial [32] and functional [20] threat models have also been explored. Functional adversarial attacks introduced a novel class of threat models which, at the time, produced the highest attack success rates even after adversarial training. The ReColorAdv [20] attack uses a parameterized function to map each color in an image to a new color in the adversarial example. Laidlaw *et al*. also explore combining this functional threat model with spatial and additive threat models, resulting in stronger, more transferable attacks. In Sec. 4.2, we explore whether poisons created using these attacks are effective.

The use of error-maximizing noises for data poisoning is highly effective as an availability attack. The most effective error-maximizing poison can poison a network to achieve 6.25% test accuracy on CIFAR-10 [8]. In Sec. 4.1, we find that if the victim network were to train a network for fewer epochs, this poison is less effective.

3. Adversarial and Synthetic Poisons

3.1. Problem Statement

We formulate the problem of creating a clean-label poison in the context of image classification with DNNs, following [17]. For a K-class classification task, we denote the clean training and test datasets as $\mathcal{D}_c$ and $\mathcal{D}_t$, respectively. We assume $\mathcal{D}_c, \mathcal{D}_t \sim \mathcal{D}$. We let f_θ represent a classification DNN with parameters θ. The goal is to perturb $\mathcal{D}_c$ into

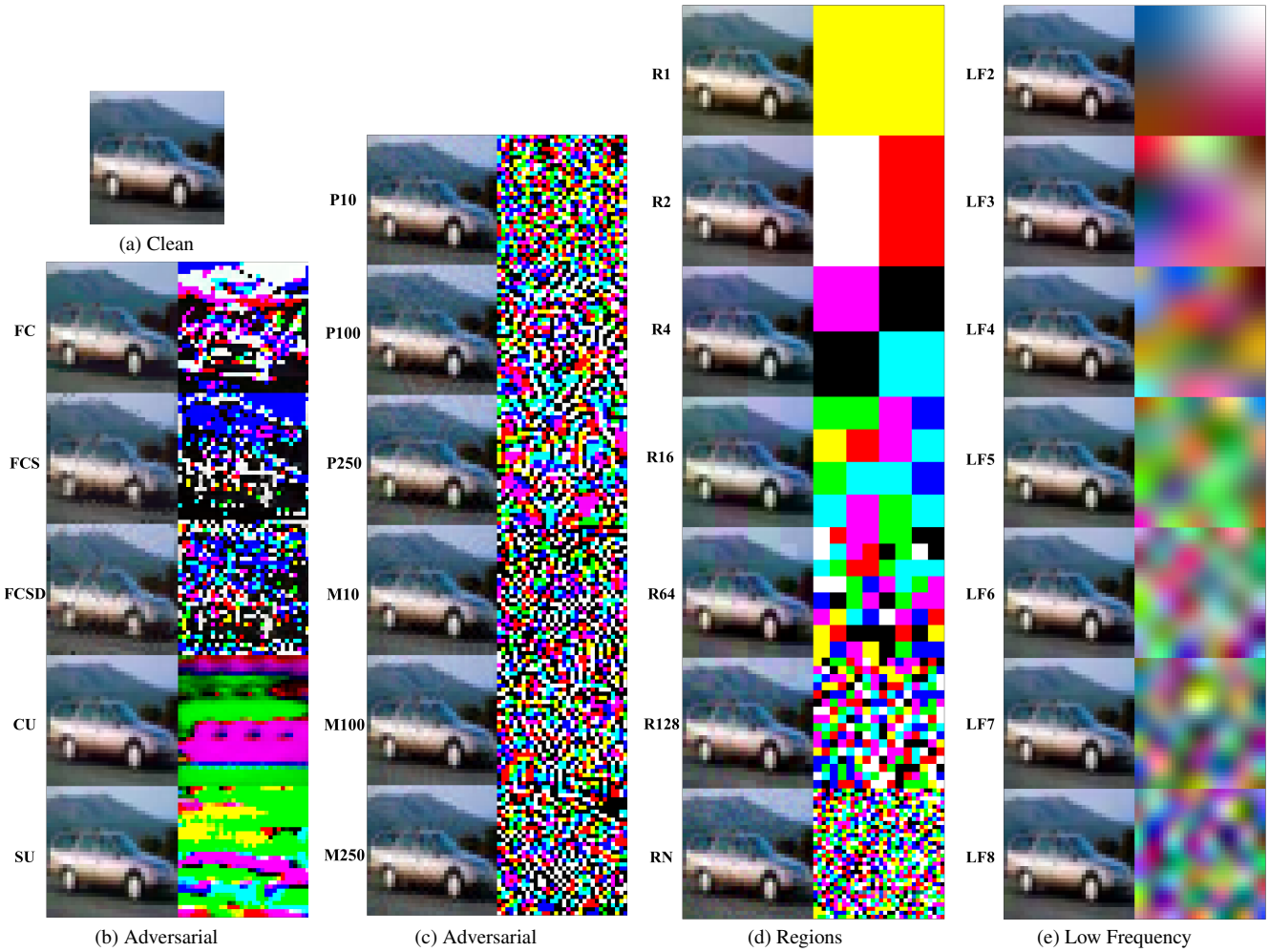

Figure 3. A CIFAR-10 image and its corresponding normalized perturbation from different poisons labeled by IDs from Tab. 1. The original clean sample is shown in (a). Error-minimizing and error-maximizing poisons are shown in columns (b) and (c). Synthetic random poisons are shown in columns (d) and (e).

a poison $\mathcal{D}_p$ such that when DNNs are trained on $\mathcal{D}_p$, they perform poorly on test set $\mathcal{D}_t$.

Suppose there are n samples in the clean training set, i.e. $\mathcal{D}_c = \{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^d$ are the inputs and $y_i \in \{1, ..., K\}$ are the labels. We denote the poisoned dataset as $\mathcal{D}_p = \{(x_i', y_i)\}_{i=1}^n$ where $x_i' = x_i + \delta$ is the poisoned version of the example $x_i \in \mathcal{D}_c$ and where $\delta \in \Delta \subset \mathbb{R}^d$ is the perturbation[1]. Perturbations δ are bounded by $\|\delta\|_p < \epsilon$ where $\|\cdot\|_p$ is the ℓ_p norm and ϵ is set to be small enough that it does not affect the utility of the example. Perturbations are typically restricted to be sampled from a set of allowable perturbations Δ which, in effect, defines the threat model.

Poisons are created by applying a perturbation to a clean image in either a class-wise or sample-wise manner. When a perturbation is applied class-wise, every sample of a given class is perturbed in the same way. That is, $x_i' = x_i + \delta_{y_i}$ and $\delta_{y_i} \in \Delta_C = \{\delta_1, ..., \delta_K\}$. Due to the explicit correlation between the perturbation and the true label, it should not be surprising that class-wise poisons appear to trick the model to learn the perturbation over the image content, subsequently reducing generalization to the clean test set. When a poison is applied sample-wise, every sample of the training set is perturbed independently. That is, $x_i' = x_i + \delta_i$ and $\delta_i \in \Delta_S = \{\delta_1, ..., \delta_n\}$.

Error-maximizing availability poisoning aims to solve the following objective in terms of perturbations $\delta_i \in \Delta$ to samples $x_i \in \mathcal{D}_c$:

$$\max_{\delta \in \Delta} \mathbb{E}_{(x,y)\sim\mathcal{D}_t} \left[\mathcal{L}(f(x), y; \theta(\delta))\right] \tag{1}$$

[1]Functional poisons are not contained within this additive threat model. Whereas an additive attack perturbs each pixel separately, functional attacks apply a function $f(\cdot)$ to every pixel, and the adversarial example is written $x_i' = f(x_i)$.

whereas error-minimizing poisons solve the following objective:

$$\min_{\delta \in \Delta} \mathbb{E}_{(x,y)\sim\mathcal{D}_t} \left[\mathcal{L}(f(x), y; \theta(\delta))\right]. \quad (2)$$

Both objectives are bi-level optimization problems, as θ is implicitly defined via:

$$\theta(\delta) = \arg\min_{\theta} \sum_{(x_i,y_i)\in\mathcal{D}_c} \mathcal{L}(f(x_i+\delta_i), y_i; \theta) \quad (3)$$

In the case of an error-maximizing poison, the adversary intends for a network, f, trained on the poison in the manner of Eq. (3), to perform poorly on the test distribution D_t, as in Eq. (1), from which D_c was also sampled. We use SGD to optimize Eq. (3) and projected gradient descent (PGD) to optimize Eq. (1) or Eq. (2), following [17] and [8].

With the exception of the functional poisons, all poisons we consider contain perturbations δ which satisfy $\|\delta\|_\infty \leq \epsilon = \frac{8}{255}$, as previous studies have determined that these perturbations are imperceptible to human observers [17]. All poisons are outlined in Tab. 1. For the remainder of our work, we refer to a poison by their ID. Only **P10** and **P100** are optimized using a ResNet-50. All other poisons are crafted using a ResNet-18 [15], trained from scratch on clean CIFAR-10 for 40 epochs. We plot a sample image and its corresponding normalized perturbation for a variety of poisons in Fig. 3.

3.2. Generating Adversarial Poisons

We craft a number of error-maximizing poisons using the open-source code of Fowl *et al*. [8]. In particular, we employ targeted 10, 100, and 250-step PGD [21] and MIFGSM [7] attacks. The poison crafted using a targeted 250-step PGD attack is the main poison published by Fowl *et al*. [8], where a ResNet-18 trained on the poison will achieve a final test set accuracy of 6.25%. Our version[2] of this poison is slightly less effective, achieving final 8.2% test accuracy. Our functional poisons make use of a functional attack, ReColorAdv (C) [20], which can be combined with a spatial attack, StAdv (S) [32] and an additive delta (D) attack like PGD. Following the naming style from Laidlaw *et al*., the **FCSD** poison perturbs images using ReColor, StAdv, and PGD. The functional poisons are the only poisons where perturbations are not constrained to an ℓ_∞ ball around the original image. Functional attacks use different distance measures to quantify perceptual similarity between the clean and the adversarial example.

We also craft two error-minimizing poisons, **SU** and **CU**, using open-source code from Huang *et al*. [17], one of which is sample-wise and the other class-wise.

[2]Fowl *et al*. only release the untargeted version of their poison, so we crafted the targeted version using their open-source code.

ID	Detail	Type	Appl.	Peak Acc (↓)	Final Acc (↓)
P10	10-step PGD	Max	S	83.43	61.84
P100	100-step PGD	Max	S	45.02	11.59
P250	250-step PGD [8]	Max	S	57.30	**8.15**
M10	10-step MIFGSM	Max	S	71.18	25.27
M100	100-step MIFGSM	Max	S	69.05	14.58
M250	250-step MIFGSM	Max	S	70.20	12.36
FC	100-step Functional C	Max	S	87.09	86.93
FCS	100-step Functional C+S	Max	S	83.82	83.37
FCSD	100-step Functional C+S+D	Max	S	80.31	80.00
SU	SW-Unlearnable [17]	Min	S	**33.77**	32.83
CU	CW-Unlearnable [17]	Min	C	33.22	19.19
R1	1 region	R	C	75.83	74.79
R2	2 regions	R	C	34.96	31.14
R4	4 regions	R	C	**24.51**	**11.45**
R16	16 regions	R	C	24.54	20.32
R64	64 regions	R	C	33.21	13.00
R128	128 regions	R	C	44.26	12.22
RN	1024 regions	R	C	66.58	11.64
LF2	2x2 DCT	R	C	53.51	45.36
LF3	3x3 DCT	R	C	43.36	42.71
LF4	4x4 DCT	R	C	48.59	48.59
LF5	5x5 DCT	R	C	47.16	29.12
LF6	6x6 DCT	R	C	51.04	39.40
LF7	7x7 DCT	R	C	53.72	32.74
LF8	8x8 DCT	R	C	75.96	41.54

Table 1. A summary of all CIFAR-10 poisons considered in this work. Poisons are given an ID for easy reference. Poison perturbations are either Error-Minimizing (Min), Error-Maximizing (Max), or Random (R), and are applied to clean images in a Class-wise (C) or Sample-wise (S) manner. The best class-wise and sample-wise poisons are in bold.

3.3. Generating Synthetic Poisons

A quick glance at the normalized error-minimizing and error-maximizing perturbations in Fig. 3 illustrates the difficulty of understanding why they work. A much simpler noise type to understand is that of synthetic, random perturbations. Our synthetic poisons are randomly generated and are designed to take advantage of two kinds of features that convolutional networks seem to be biased towards: spatially-local [1] and low frequency [14, 22]. Both these synthetic class-wise noises represent a first step in designing poisons which are learned quickly.

Regions noise. To generate a noise with n patch regions, we sample n vectors of size 3 from a Bernoulli distribution. Each vector is then scaled to lie in the range $[-\frac{8}{255}, \frac{8}{255}]$. Finally, the n vectors are repeated along two dimensions to achieve a shape of $32\times32\times3$. With the exception of the **R2** poison, a Regions noise contains patches of size $\frac{32}{\sqrt{n}} \times \frac{32}{\sqrt{n}}$.

Low frequency noise. To generate low frequency perturbations, we first sample an $n \times n \times 3$ matrix of Gaussian noise, representing a DCT block. We then append zeros to the matrix along the first two dimensions to achieve a shape of $32 \times 32 \times 3$ and subsequently perform the inverse DCT transform. When n is small, the resulting image contains low frequency patterns.

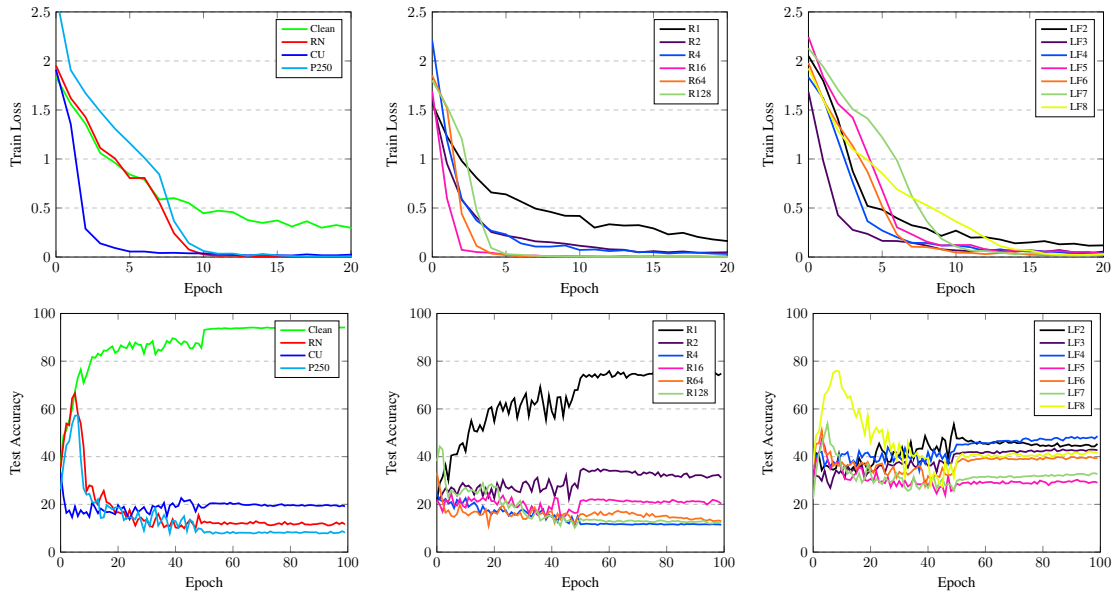

Figure 4. Train Loss and Test Accuracy curves when training a ResNet-18 on a variety of poisons. The first column displays curves for a clean training run plus three selected poisons: class-wise random noise (RN), class-wise unlearnable (CU), and 250-step PGD (P250). The second column displays curves for regions poisons. The third column displays curves for low-frequency poisons.

4. Experiments

Employing a variety of poisons, our experiments were designed to answer two key questions: Are poisons that are learned faster a better defense against early stopping? and do stronger, more transferable adversarial attacks make more effective poisons? In Sec. 4.1, we train a number of ResNet-18 models on different poisons with cross-entropy loss for 100 epochs using a batch size of 128. For our optimizer, we use SGD with momentum of 0.9 and weight decay of 5×10^{-4}. We use an initial learning rate of 0.1, which decays by a factor of 10 on epoch 50.

4.1. Poisons learned faster are more effective

While poison effectiveness is generally measured by the final test set accuracy of a poisoned model, an arbitrarily low final test set accuracy is not effective if the victim could simply stop training early and obtain better test set performance. Thus, we define peak test accuracy as the highest test set performance over 100 epochs, and evaluate our poisons from Tab. 1 on this metric. As we show in Fig. 2, there are a number of poisons which reach low final test accuracy, but high peak test accuracy.

To define the speed at which a poison is learned, we arbitrarily choose a loss threshold of 0.5 and take note of the first epoch when the training loss is less than 0.5. This is a reasonable loss threshold given that nearly 85% of poisons we tested reach a cross-entropy loss of 0.5 by epoch 10. By taking note of the epoch after which a model reaches this threshold, we obtain a proxy for the ease with which a poison is learned.

For every poison, we hence obtain both an epoch until the loss threshold is reached and the overall peak test accuracy during training. We plot our results in Fig. 1 and observe that poisons which are learned faster reach lower peak test accuracy, and are thus better defended against early-stopping. We provide training loss and validation accuracy curves in Fig. 4. Our version of the sample-wise adversarial poison from Fowl *et al*. [8], which we label **P250**, reaches a final test set accuracy of 8.15%, but reaches a peak test accuracy of 57.3% as shown in Fig. 4 — the poison is 7× less effective when we stop training at epoch 8! This serves as an example that poisons should be evaluated by their worst-case performance. The correlation between how quickly a poison is learned and its peak test accuracy suggests we should strive to minimize the loss early in training.

Our synthetic regions poisons and low-frequency poisons are a first step at designing perturbations which are learned quickly by convolutional DNNs like ResNet-18. Attempting to take advantage of a CNN's bias for spatially-local image features, our regions poisons have patches of the same color. Taking advantage of a DNNs bias for low-frequency patterns, our low frequency poisons span a spectrum of complexity. Training curves from Fig. 4 demonstrate that as the number of regions increases, poisons are generally learned more quickly, but there is a sweet spot for the number of regions (**R4** and **R16** have particularly low peak accuracy and final test accuracy). Compared to regions poisons, low frequency poisons perform more poorly overall. Interestingly, as the number of frequencies increases

Poison	Victim Networks					Poison Detail	
	ResNet-18	VGG-19	MobileNet	GoogLeNet	Avg (↑)	Peak Acc (↓)	Final Acc (↓)
RN	8.35	6.28	23.50	10.43	12.14	66.58	11.64
P10	14.24	7.55	34.55	13.00	17.34	83.43	61.84
P100	83.90	65.55	90.03	81.57	80.26	**45.02**	11.59
P250	95.55	84.58	95.73	90.93	**91.70**	57.30	**8.15**
M10	38.40	20.21	46.56	29.35	33.63	71.18	25.27
M100	53.18	22.79	46.75	43.12	41.46	69.05	14.58
M250	48.70	20.65	46.71	36.42	38.12	70.20	12.36
FC	94.95	38.88	53.88	50.85	59.64	87.09	86.93
FCS	99.97	73.18	76.23	88.69	84.52	83.82	83.37
FCSD	100.00	83.53	86.24	94.74	**91.13**	80.31	80.00

Table 2. Attack transferability of different error-maximizing CIFAR-10 poisons against victim models of varying architecture. Peak and Final accuracy (%) are for a ResNet-18 trained on the poison. Values within 1% of the best are in bold.

from **LF2** to **LF8**, the poisons appear to be learned more slowly, and the corresponding test accuracy curves show higher peak accuracies. For example, **LF7** reaches a peak accuracy of 53.72% while **LF8** reaches 75.96%. Further work is needed to develop a poison which can induce lower than **R4**'s 24.51% peak test accuracy on CIFAR-10.

4.2. Stronger attacks do not make stronger poisons

Both error-maximizing noise and error-minimizing noises were evaluated in Huang *et al*. [17], but their proposed error-minimizing sample-wise noise produced lower clean test set accuracy across the entire training process compared to other all other poisons. To optimize their error-maximizing noise, they used a 20-step PGD attack.

Despite the apparent success of error-minimizing noise over error-maximizing noise, Fowl *et al*. found a way to bring error-maximizing noise back into the spotlight: using a stronger 250-step PGD attack [8]. Why did a 250-step attack behave so differently as a poison, when compared to the 20-step PGD attack? The findings of Fowl *et al*. [8] could be used as an indication that stronger adversarial attacks are the key to improving error-maximizing poisoning performance. Rather than continue to increase the number of attack steps, we opt to investigate whether adversarial attack transferability is correlated with poisoning performance.

To evaluate this claim, we conduct an analysis of adversarial attack transferability across a range of error-maximizing attacks from Tab. 1. We consider four DNN architectures as victim networks: a ResNet-18 [15], VGG-19 [26], MobileNet [16], and GoogLeNet [27], which we train on CIFAR-10 for 200 epochs. Recall that sample-wise error-maximizing poisons are constructed by performing an adversarial attack on each clean image from the training set. Thus, attack transferability is measured by computing the fraction of examples in the poison which are misclassified by the victim network. We present attack transferability results alongside the corresponding poison's peak and final test accuracy in Tab. 2.

An adversarial attack can be considered stronger than another if it uses more attack steps or if it is more transferable to other DNNs of differing architectures. In terms of peak test accuracy, increasing the number of attack steps for a PGD or MI-FGSM attack does not improve peak test accuracy. For example, **P100** achieves a peak test accuracy of 45.02%, while the more transferable **P250** poison achieves 57.30%. Stronger, more transferable adversarial attacks do not necessarily degrade test accuracy either. The case of **FCSD** is a striking example: the adversarial attack is one of the most transferable across the four victim networks, yet performs very poorly as a poison.

5. Conclusion

In this paper, we made a number of observations which should be taken into account when designing poisons for the purpose of data privacy protection. Motivated by defending against the mitigating effect of early-stopping, we found that poison training exhibits a correlation between how quickly a model reaches a low loss threshold and the peak test accuracy overall. This leads us to propose that poisons should be learned quickly to ensure peak test accuracy remains low throughout training and user privacy is meaningfully protected.

To further understand this phenomenon, we craft two kinds of synthetic poisons aimed at taking advantage of CNN biases. But while random, spatially-local regions poisons are learned quickly, they suffer from being slightly perceptible. Low frequency perturbations, on the other hand, do not achieve low test accuracy.

Finally, our experiments suggest that adversarial attacks which are more transferable or stronger in terms of steps do not lead to effective error-maximizing poisons. Due to high peak test accuracy, we caution against error-maximizing noises for data poisoning.

Acknowledgements. This work is supported in part by the Guaranteeing AI Robustness Against Deception (GARD) program from DARPA. Pedro is supported by an Amazon Lab126 Diversity in Robotics and AI Fellowship.

References

[1] Nicholas Baker, Hongjing Lu, Gennady Erlikhman, and Philip J Kellman. Local features and global shape information in object classification by deep convolutional neural networks. *Vision research*, 172:46–61, 2020. 4

[2] Marco Barreno, Blaine Nelson, Anthony D Joseph, and J Doug Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010. 1

[3] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012. 1

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1

[5] Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, May 2017. ISSN: 2375-1207. 2

[6] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017. 1

[7] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 4

[8] Liam Fowl, Micah Goldblum, Ping-yeh Chiang, Jonas Geiping, Wojtek Czaja, and Tom Goldstein. Adversarial Examples Make Strong Poisons. *arXiv:2106.10807 [cs]*, June 2021. arXiv: 2106.10807. 1, 2, 4, 5, 6

[9] Liam H Fowl, Ping yeh Chiang, Micah Goldblum, Jonas Geiping, Arpit Amit Bansal, Wojciech Czaja, and Tom Goldstein. Protecting proprietary data: Poisoning for secure dataset release, 2022. 1

[10] Jonas Geiping, Liam Fowl, Gowthami Somepalli, Micah Goldblum, Michael Moeller, and Tom Goldstein. What Doesn't Kill You Makes You Robust(er): Adversarial Training against Poisons and Backdoors. *arXiv:2102.13624 [cs]*, Feb. 2021. arXiv: 2102.13624. 1

[11] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein. Dataset Security for Machine Learning: Data Poisoning, Backdoor Attacks, and Defenses. *arXiv:2012.10544 [cs]*, Mar. 2021. arXiv: 2012.10544. 1, 2

[12] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. *arXiv:1412.6572 [cs, stat]*, Mar. 2015. arXiv: 1412.6572. 2

[13] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017. 1

[14] Chuan Guo, Jared S Frank, and Kilian Q Weinberger. Low frequency adversarial perturbation. *arXiv preprint arXiv:1809.08758*, 2018. 4

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4, 6

[16] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv*, abs/1704.04861, 2017. 6

[17] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. In *ICLR*, 2021. 1, 2, 4, 6

[18] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2

[19] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial Machine Learning at Scale. *arXiv:1611.01236 [cs, stat]*, Feb. 2017. arXiv: 1611.01236. 2

[20] Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. *Advances in neural information processing systems*, 32, 2019. 2, 4

[21] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv:1706.06083 [cs, stat]*, Sept. 2019. arXiv: 1706.06083. 2, 4

[22] Guillermo Ortiz-Jimenez, Apostolos Modas, Seyed-Mohsen Moosavi, and Pascal Frossard. Hold me tight! influence of discriminative features on deep network boundaries. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 2935–2946. Curran Associates, Inc., 2020. 4

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 1

[24] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *Advances in Neural Information Processing Systems*, 31, 2018. 1

[25] Juncheng Shen, Xiaolei Zhu, and De Ma. Tensorclog: An imperceptible poisoning attack on deep neural network applications. *IEEE Access*, 7:41498–41506, 2019. 2

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015. 6

[27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, D. Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015. 6

[28] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv:1312.6199 [cs]*, Feb. 2014. arXiv: 1312.6199. 2

[29] Lue Tao, Lei Feng, Jinfeng Yi, Sheng-Jun Huang, and Songcan Chen. Provable defense against delusive poisoning. *arXiv preprint arXiv:2102.04716*, 2021. 1

[30] Eric Wallace, Tony Z Zhao, Shi Feng, and Sameer Singh. Concealed data poisoning attacks on nlp models. *arXiv preprint arXiv:2010.12563*, 2020. 1

[31] Zhirui Wang, Yifei Wang, and Yisen Wang. Fooling Adversarial Training with Inducing Noise. *arXiv:2111.10130 [cs, stat]*, Nov. 2021. arXiv: 2111.10130. 1

[32] Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially Transformed Adversarial Examples. *arXiv:1801.02612 [cs, stat]*, Jan. 2018. arXiv: 1801.02612. 2, 4

[33] Chen Zhu, W Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *International Conference on Machine Learning*, pages 7614–7623. PMLR, 2019. 1