ADVERSARIAL EXAMPLES MAKE STRONGER POISONS

Liam Fowl*
Department of Mathematics
University of Maryland
lfowl@umd.edu

Micah Goldblum *
Department of Computer Science
University of Maryland
goldblum@umd.edu

Ping-yeh Chiang
Department of Computer Science
University of Maryland
pchiang@cs.umd.edu

Jonas Geiping
Dep. of Electr. Eng. and Computer Science
University of Siegen
jonas.geiping@uni-siegen.de

Wojciech Czaja
Department of Mathematics
University of Maryland
wojtek@math.umd.edu

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

ABSTRACT

Adversarial attacks craft perturbations during inference to fool already trained models. In contrast, data poisoning attacks manipulate training data. Early poisoning works focused on inducing a large performance drop in linear models which train on the modified data, but these methods are not effective against deep neural networks. In this work, we discover that adversarial examples, originally intended for attacking pre-trained models, are even more effective for data poisoning than existing data poisoning methods. Models trained on adversarially perturbed data cannot even recognize the original unperturbed training images. Our findings indicate that adversarial examples look fundamentally different from clean examples to neural networks, and learning on one is useless for performing inference on the other. Furthermore, adversarial examples with labels re-assigned by the crafting network are ineffective poisons, suggesting that adversarial examples contain useful semantic content, just from the incorrect classes.

1 INTRODUCTION

The threat of malicious dataset manipulations is ever increasing as the training data curation process for machine learning systems is increasingly void of human supervision. Automated scraping has become necessary to satisfy the exploding demands of cutting-edge deep models (Bonawitz et al., 2019; Brown et al., 2020), but the same automation that enables massive performance boosts exposes these models to security vulnerabilities (Bagdasaryan et al., 2020; Chen et al., 2020). *Data poisoning* attacks manipulate training data in order to cause the resulting models to misclassify samples during inference (Koh & Liang, 2017), while *backdoor attacks* embed exploits which can be triggered by pre-specified input features (Chen et al., 2017). In this work, we focus on a flavor of data poisoning known as *availability attacks*, which aim to degrade overall testing performance **?**Biggio et al. (2012).

Adversarial attacks instead focus on manipulating samples at test-time, rather during training (Szegedy et al., 2013). In this work, we connect adversarial and poisoning attacks by showing that adversarial examples form stronger availability attacks than any existing poisoning method, even though the latter were designed specifically for manipulating training data while adversarial examples were not. We compare our method, *adversarial poisoning*, to all existing availability attacks for neural networks, and we exhibit consistent performance boosts. In fact, models trained on adversarial examples may exhibit test-time performance below that of random guessing.

*Authors contributed equally.

Intuitively, adversarial examples look dramatically different from the originals in the eye of neural networks despite the two looking similar to humans. Thus, models trained only on such perturbed examples are completely unprepared for inference on the clean data. In support of this intuition, we observe that models trained on adversarially perturbed training data cannot even classify the original clean training samples.

But does this phenomenon occur simply because adversarial examples are off the "natural image manifold" or because they actually contain informative features from other classes? Popular belief assumes that adversarial examples live off the natural image manifold causing a catastrophic mismatch when digested by models only trained on clean data (Khoury & Hadfield-Menell, 2018; Stutz et al., 2019; Zhao et al., 2017). However, models trained on data with random additive noise (rather than adversarial noise) perform well on noiseless data, suggesting that the affects of adversarial examples do not simply arise from moving off the manifold (see Table 1). We instead find that since adversarial attacks inject features that a model associates with incorrect labels, training on these examples is similar to training on mislabeled training data. After re-labeling adversarial examples with the "wrong" prediction of the network with which they were crafted, models trained on such label-corrected dataset perform substantially better than models trained on uncorrected adversarial examples and almost as well as models trained on clean images.

2 Related Work

Data poisoning. Typical approaches to availability attacks involve solving a bilevel optimization problem which minimizes loss with respect to parameters in the inner problem while maximizing loss with respect to inputs in the outer problem (Biggio et al., 2012; Huang et al., 2020). On simple models, the inner problem can be solved exactly, and many works leverage this (Biggio et al., 2012; Mei & Zhu, 2015; Xiao et al., 2015), but on neural networks, obtaining exact solutions is intractable. To remedy this problem, (Muñoz-González et al., 2017) approximates a solution to the inner problem using a small number of descent steps, but the authors note that this method is ineffective against deep neural networks. Other works adopt similar approximations but for integrity attacks on deep networks, where the attacker only tries to degrade performance on one particular test sample (Geiping et al., 2020; Huang et al., 2020). The bilevel approximation of Geiping et al. (2020) can also adapted to availability attacks for neural networks by substituting the original targeted adversarial objective with an indiscriminate adversarial objective. Another related approach harnesses *influence functions* which estimate the impact of each training sample on a resulting model (Fang et al., 2020; Koh & Liang, 2017; Koh et al., 2018). However, influence functions are brittle on deep networks whose loss surfaces are highly irregular (Basu et al., 2020). In order to conduct availability attacks on neural networks, recent works have instead modified data to cause gradient vanishing either explicitly or by minimizing loss with respect to the image, thus removing the influence of data on training (Huang et al., 2021; **?**). We compare to these methods and find that adversarial poisoning is significantly more effective.

Adversarial examples. Adversarial attacks probe the blindspots of trained models where they catastrophically misclassify inputs that have undergone small perturbations (Szegedy et al., 2013). Prototypical algorithms for adversarial attacks simply maximize loss with respect to the input while constraining perturbations. The resulting adversarial examples exploit the fact that as inputs are even slightly perturbed in just the right direction, their corresponding deep features and logits change dramatically, and gradient-based optimizers can efficiently find these directions. The literature contains a wide array of proposed loss functions and optimizers for improving the effectiveness of attacks (Carlini & Wagner, 2017; Gowal et al., 2019). A number of works suggest that adversarial examples are off the image manifold, and others propose methods for producing on-manifold attacks (Khoury & Hadfield-Menell, 2018; Stutz et al., 2019; Zhao et al., 2017).

Adversarial training. The most popular method for producing neural networks which are robust to attacks involves crafting adversarial versions of each mini-batch and training on these versions (Madry et al., 2017). On the surface, it might sound as if adversarial training is very similar to training on poisons crafted via adversarial attacks. After all, they both involve training on adversarial examples. However, adversarial training ensures that the robust model classifies inputs correctly within a ball surrounding each training sample. This is accomplished by updating perturbations to inputs *throughout* training. This process desensitizes the adversarially trained model to small perturbations

to its inputs. In contrast, a model trained on adversarially poisoned data is only encouraged to fit the exact, fixed perturbed data.

3 ADVERSARIAL EXAMPLES AS POISONS

Formally stated, availability poisoning attacks aim to solve the following bi-level objective in terms of perturbations $\delta = \{\delta_i\}$ to elements x_i of a dataset $\mathcal{T}$

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

$$\text{s.t. } \theta(\delta) \in \arg\min_{\theta} \sum_{(x_i,y_i)\in\mathcal{T}} \mathcal{L}(F(x_i + \delta_i; \theta), y_i), \tag{2}$$

where $\mathcal{S}$ denotes the constraint set of the perturbations. As is common in both the adversarial attack and poisoning literature, we employ an ε- ℓ_∞ bound on each δ_i (Madry et al., 2017; Zhu et al., 2019; Geiping et al., 2020). Unless otherwise stated, our attacks are bounded by $\epsilon = 8/255$ as is common on CIFAR-10 data in both adversarial and poisoning literature (Madry et al., 2017; Geiping et al., 2020). Simply put, the attacker wishes to cause a network, F, trained on the poisons to generalize poorly to distribution $\mathcal{D}$ from which $\mathcal{T}$ was sampled.

Directly solving this optimization problem is intractible for neural networks as it requires unrolling the entire training procedure and backpropagating through it to perform a single step of gradient descent on the outer objective. Thus, the attacker must approximate the bilevel objective. Approximations to this objective often involve heuristics. For example, TensorClog aims to cause gradient vanishing in order to disrupt training, while more recent work aims to align poison gradients with an adversarial objective (**?**Geiping et al., 2020). We opt for an entirely different strategy and instead replace the bi-level problem with an empirical loss maximization problem, thus turning the poison generation problem into an adversarial example problem. Specifically, optimize the following objective:

$$\max_{\delta \in \mathcal{S}} \Big[\sum_{(x_i,y_i)\in\mathcal{T}} \mathcal{L}\left(F(x_i + \delta_i; \theta^*), y_i\right) \Big] \tag{3}$$

where θ^* denotes the parameters of a pre-trained crafting model. Fittingly, we call our method *adversarial poisoning*.

Projected Gradient Descent (PGD) has become the standard method for generating adversarial examples for deep networks (Madry et al., 2017). Accordingly, we craft our poisons with 250 steps of PGD on this loss-maximization objective. Borrowing from recent targeted data poisoning works, we also employ differentiable data augmentation in the crafting stage (Geiping et al., 2020). We study the effect of optimizer, data augmentation, steps, and crafting network on poison generation in Appendix Tables 6, 7, and 8. These tables demonstrate that a wide variety of adversarial attacks with various crafting networks and hyperparameters yield effective poisons.

We compare our method to existing availability attacks including TensorClog (**?**), Loss Minimization (Huang et al., 2021), and an availability adaptation of the alignment poisoning method found in Geiping et al. (2020). We find that poisons generated via the proposed method transfer to a variety of common architectures in the black-box setting. These results can be found in Table 1 and appendix Table 5. Compared with the previous best method (loss minimization), we degrade the validation accuracy of a victim network by a factor of more than three. We also compare different methods of adversarial example generation in appendix Table 4. Specifically, we comparere to a Carlini-Wagner (CW) attack wherein the attacker aims to maximize the second highest logit in order to cause misclassification (Carlini & Wagner, 2017), a vanilla Fast Gradient Sign Method (FGSM) attack (Goodfellow et al., 2014), and a feature explosion attack where the attacker maximizes the ℓ_2 norm of feature vectors. We find that while other adversaries do produce effective poisons, a PGD based attack is the most effective in generating poisons.

Note that we test our method in a completely black-box setting wherein the attacker has no knowledge of the victim network's initialization, architecture, learning rate scheduler, optimizer, etc. We find our adversarial poisons transfer across these settings and reliably degrade the validation accuracy of all the models tested.

Table 1: **Validation accuracies of models trained on data from different availability attacks.** Tested on randomly initialized ResNet-18 models on CIFAR-10. All crafted with $\varepsilon = 8/255$.

METHOD	VALIDATION ACCURACY (%, ↓)
NONE (CLEAN)	94.56
TENSORCLOG	84.24
ALIGNMENT	36.83
RANDOM NOISE	90.52
LOSS MINIMIZATION	19.93
ADVERSARIAL POISONING (OURS)	**6.25**

4 ANALYSIS

Why do adversarial examples make such potent poisons? In Section 1, we motivate the effectiveness of adversarial poisons with the explanation that the perturbed data contains semantically useful information, but for the wrong class. For instance, an adversarially perturbed "dog" might be labeled as a "cat" by the crafting network because the perturbations contain discriminatory features useful for the "cat" class. In Ilyas et al. (2019), the authors discover that there exist image features which are both brittle under adversarial perturbations and useful for classification. Thus, adversarial examples might poison networks so effectively because, for example, they teach the network to associate "cat" features found in perturbed data with the label "dog" of the original, unperturbed sample. Then, when the network tries to classify clean test-time data, it leverages the "mislabeled" features found in the perturbed data and displays low test-time accuracy. We confirm this behavior by evaluating *how* data is misclassified at test-time. We find that the distribution of predictions on *clean* data closely mimics the distributions of labels assigned by the network used for crafting after adversarial attacks (c.f. Appendix Figure 2).

To tease apart these effects, we conduct several experiments. First, we verify that the victim network does indeed train - i.e. reach a region of low loss - on the adversarial examples. This is in contrast to the motivation of **?**Huang et al. (2021) which try to *prevent* the network from training on poisoned data. We find that the victim network is able to almost perfectly fit the adversarial data. However, the accuracy of the victim network on the original, unperturbed training data is just as low as accuracy on the clean validation data, revealing an interesting duality - clean training data are adversarial examples for networks trained on their perturbed counterparts (see Table 2). But are adversarial examples simply so different from clean examples that learning on one is useless for performing inference on the other? Or do adversarial examples contain useful features but for the *wrong* class?

To tease out these hypotheses, we train models on the adversarially poisoned data but with "corrected" labels - labels assigned by the crafting network to the poisons, rather than the ground-truth label of their clean counterparts. For example, if a "dog" is perturbed into the "cat" class by an adversarial attack, we train on the perturbed "dog" image, but assign it the "cat" label. Intriguingly, we find that this simple re-labeling trick boosts validation accuracy significantly. For example from 6.25% to 75.69% on a victim ResNet-18 model (see Table 3). This confirms the finding of Ilyas et al. (2019) concerning non-robust features. One further intricacy remains; even if adversarial examples contained no useful features, they may still encode decision boundary information that accounts for the increased validation accuracy - behaviour that has previously been demonstrated with out-of-distribution data in Nayak et al. (2019). However, we find that training on data from a drastically different distribution (SVHN), labeled with the CIFAR-10 crafting network's predictions, fails to achieve comparable CIFAR-10 validation accuracy. This confirms that the adversarial CIFAR-10 images contain useful features for the CIFAR-10 distribution but are simply mislabeled.

Table 2: **Testing the victim on clean vs. adversarial training images.** Poisons are crafted on a CIFAR-10 trained ResNet-18 with 250 steps of PGD and differentiable data augmentation.

MEASUREMENT \ VICTIM	VGG19	RESNET-18	GOOGLENET	DENSENET121	MOBILENETV2
TRAINING ACC. ON POISONS	99.95 ± 0.00	99.99 ± 0.00	99.99 ± 0.00	99.99 ± 0.00	99.94 ± 0.00
ACC. ON CLEAN TRAIN DATA	10.98 ± 0.32	6.16 ± 0.16	6.80 ± 0.08	7.06 ± 0.13	6.15 ± 0.17

Table 3: **Training the victim with labels corrected to the "adversarial labels".** Poisons are crafted on a CIFAR-10 trained ResNet-18 with 250 steps of PGD and differentiable data augmentation.

DATA \VICTIM	VGG19	RESNET-18	GOOGLENET	DENSENET121	MOBILENETV2
UNCORRECTED CIFAR-10	10.98 ± 0.27	6.25 ± 0.17	7.03 ± 0.12	7.16 ± 0.16	6.11 ± 0.17
CORRECTED ONE-HOT CIFAR-10	74.95 ± 0.31	78.98 ± 0.25	77.72 ± 0.37	78.55 ± 0.36	74.33 ± 0.24
CORRECTED SOFTMAX CIFAR-10	75.88 ± 0.25	75.69 ± 0.25	73.57 ± 0.47	70.26 ± 0.32	69.46 ± 0.32
CORRECTED ONE-HOT SVHN	31.13 ± 0.49	30.19 ± 0.16	40.71 ± 0.12	40.31 ± 0.43	34.18 ± 0.29

REFERENCES

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 2938–2948. PMLR, 2020.

Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile. *arXiv preprint arXiv:2006.14651*, 2020.

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

Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, H Brendan McMahan, et al. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*, 2019.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. IEEE, 2017.

Xiaoyi Chen, Ahmed Salem, Michael Backes, Shiqing Ma, and Yang Zhang. Badnl: Backdoor attacks against nlp models. *arXiv preprint arXiv:2006.01043*, 2020.

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.

Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. Influence function based data poisoning attacks to top-n recommender systems. In *Proceedings of The Web Conference 2020*, pp. 3019–3025, 2020.

Jonas Geiping, Liam Fowl, W Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. Witches' brew: Industrial scale data poisoning via gradient matching. *arXiv preprint arXiv:2009.02276*, 2020.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Sven Gowal, Jonathan Uesato, Chongli Qin, Po-Sen Huang, Timothy Mann, and Pushmeet Kohli. An alternative surrogate loss for pgd-based adversarial testing. *arXiv preprint arXiv:1910.09338*, 2019.

Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. *arXiv preprint arXiv:2101.04898*, 2021.

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

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *arXiv preprint arXiv:1905.02175*, 2019.

Marc Khoury and Dylan Hadfield-Menell. On the geometry of adversarial examples. *arXiv preprint arXiv:1811.00525*, 2018.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pp. 1885–1894. PMLR, 2017.

Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger data poisoning attacks break data sanitization defenses. *arXiv preprint arXiv:1811.00741*, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

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

Gaurav Kumar Nayak, Konda Reddy Mopuri, Vaisakh Shaj, Venkatesh Babu Radhakrishnan, and Anirban Chakraborty. Zero-shot knowledge distillation in deep networks. In *International Conference on Machine Learning*, pp. 4743–4751. PMLR, 2019.

David Stutz, Matthias Hein, and Bernt Schiele. Disentangling adversarial robustness and generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6976–6987, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

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

Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*, 2017.

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*, pp. 7614–7623. PMLR, 2019.

A Appendix

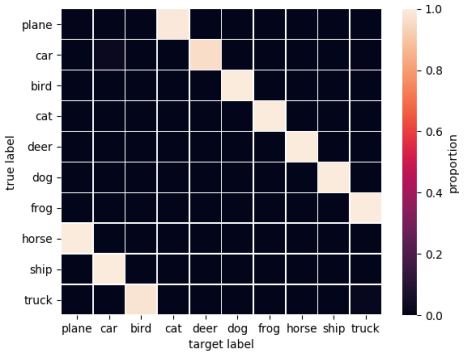
A.1 Visualization

In Figure 1, we visualize randomly selected perturbed images at different ε levels. As with adversarial attacks, and other poisoning attacks, there is a trade-off between visual similarity and potency of the perturbations.

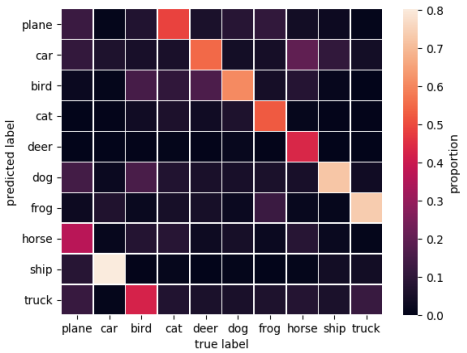
Figure 1: Randomly selected example perturbations to CIFAR-10 datapoint (class "frog"). **Left**: unaltered base image. **Middle**: $\varepsilon = 4/255$ perturbation. **Right**: $\varepsilon = 8/255$ perturbation. Networks trained on perturbations including the one on the right achieve below random accuracy.

A.2 Classification patterns

Here we experiment whether a class-wise targeted adversarial attack where every image from a given class is perturbed into one different class. For example, all "dog" images might be perturbed to look like "cats". We then see whether the test-time behavior of the network mimics the attack pattern. Does a network trained on adversarial poisons learn to associate "clean" cat features with dogs? We find that the pattern of clean, test-time classification does indeed match that of the class-wise adversarial attacks - see Figure 2.

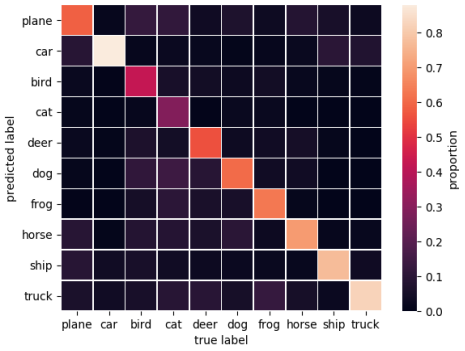


(a) Adversarial attack predictions of network used to craft adversarial poisons.

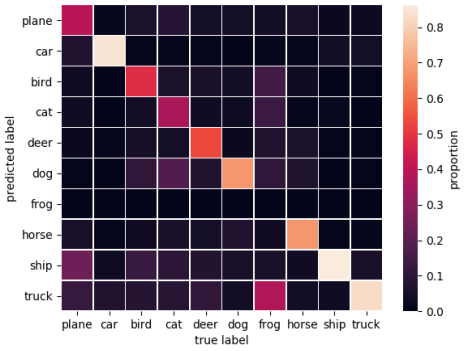


(b) Test-time predictions of network trained on adversarial poisons.

Figure 2: Classification heatmaps. **Left** - heatmap of predictions after the adversarial attack on the network used for crafting. **Right** - heatmap of clean test predictions after training a new network on adversarial poisons.



(a) Test-time predictions of network trained on label-corrected, class-targeted poisons.



(b) Test-time predictions of network trained on label-corrected, class-targeted poisons without any images of "cats".

Figure 3: Classification heatmaps. **Left** - heatmap of clean test predictions from network trained on label-corrected, class based targeted poisons. **Right** - heatmap of clean test predictions from network trained on same poisons, without any "cat" images.

A.3 Ablating data

We have seen that it is not necessary to have "correctly" labeled data (labeled with ground truth labels) in order to get good performance at test-time. We can extend this to the question: does one need ground truth data to learn how to classify? For example, can a network learn how to classify cats without ever seeing an image of a cat, but instead only seeing images of dogs perturbed to look like cats? We find the answer is yes. Specifically, we conduct an experiment where we craft targeted, class-based poisons (i.e. all images of one class are perturbed via targeted attacks into another class). We then train on the label-corrected full data, and also train on a label-corrected pruned training set without any images from the cat class (label 3, according to the ground-truth labels). Interestingly, we find that the network trained on the ablated set is able to classify clean, test-time images of cats having never seen an example during training! Moreover, the network fails to classify clean images from the class into which cats were perturbed (class 6, "frog") under the targeted attack. Instead, clean test-time frogs are more likely to be classified as class 9 ("truck") - the class where frog images were perturbed into under the crafting attack. This means the network not only learns to associate the perturbed features of frogs with the label "truck", but also features of clean frog images even though the network never encountered "clean" frog features during training because the cat class was ablated. These results can be found in Figure 3.

A.4 ADVERSARY COMPARISON

We find that a PGD based attack supersedes other common adversarial attacks in poison efficiency - a behavior that has been observed in classical adversarial attacks as well. These results can be found in Table 4.

Table 4: **Validation accuracies of victim models trained on data generated by different adversarial attacks.** Tested in the black-box setting on randomly initialized models on CIFAR-10.

METHOD \VICTIM	VGG19	RESNET-18	GOOGLENET	DENSENET121	MOBILENETV2
PGD W/ AUG	10.98 ± 0.27	6.25 ± 0.17	7.03 ± 0.12	7.16 ± 0.16	6.11 ± 0.17
CW	59.82 ± 1.36	48.40 ± 3.24	19.25 ± 1.15	43.40 ± 2.76	45.62 ± 4.87
FGSM	65.32 ± 0.58	76.35 ± 0.08	47.50 ± 1.06	59.44 ± 1.28	74.77 ± 0.43
FEATURE EXPLOSION	82.41 ± 0.38	83.26 ± 0.94	78.53 ± 0.49	81.83 ± 0.46	82.30 ± 0.88

A.5 EPSILON COMPARISON

We compare different values of the ℓ_∞ bound for the poisons and find that even with a very small perturbation, $\varepsilon = 4/255$, adversarial poisons are able to significantly degrade the validation accuracies of randomly initialized networks (see Table 5).

Table 5: **Comparison of different ε-bounds for our adversarial poisoning method.** All poisons generated by ResNet-18 crafted with 250 steps of PGD, with differentiable data augmentation.

BOUND \VICTIM	VGG19	RESNET-18	GOOGLENET	DENSENET121	MOBILENETV2
CLEAN	92.24 ± 0.09	94.56 ± 0.06	94.68 ± 0.02	94.90 ± 0.03	92.31 ± 0.06
$\varepsilon = 4/255$	52.97 ± 1.23	40.14 ± 0.50	41.06 ± 0.22	42.91 ± 0.51	33.99 ± 0.45
$\varepsilon = 8/255$	10.98 ± 0.27	6.25 ± 0.17	7.03 ± 0.12	7.16 ± 0.16	6.11 ± 0.17

A.6 CRAFTING ABLATIONS

In Table 6, we find that the more steps we perform in the PGD optimization of adversarial poisons, the more effective they become. However, the poisons still degrade validation accuracy at lower numbers of steps.

Table 6: **Comparison of different number of crafting steps for our adversarial poisoning method.** All poisons generated by ResNet-18 with steps of PGD, with differentiable data augmentation.

STEPS \VICTIM	VGG19	RESNET-18	GOOGLENET	DENSENET121	MOBILENETV2
50 STEPS	25.10 ± 0.60	16.53 ± 0.26	18.51 ± 0.34	18.91 ± 0.34	15.29 ± 0.40
100 STEPS	16.06 ± 0.41	9.87 ± 0.26	11.66 ± 0.33	13.42 ± 0.28	10.38 ± 0.22
250 STEPS	10.98 ± 0.27	6.25 ± 0.17	7.03 ± 0.12	7.16 ± 0.16	6.11 ± 0.17

Table 7: **A comparison of different optimizers, crafting objectives, and use of differentiable data augmentation in crafting.** All poisons crafted with bound $\varepsilon = 8/255$. If not otherwise stated, poisons are crafted with PGD, differentiable data augmentation.

METHOD \VICTIM	VGG19	RESNET-18	GOOGLENET	DENSENET121	MOBILENETV2
PGD W/ AUG	10.98 ± 0.27	6.25 ± 0.17	7.03 ± 0.12	7.16 ± 0.16	6.11 ± 0.17
SIGNADAM W/ AUG	8.92 ± 0.27	6.17 ± 0.27	6.75 ± 0.18	6.42 ± 0.23	7.15 ± 0.22
SIGNADAM W/O AUG	9.96 ± 0.22	6.96 ± 0.15	7.41 ± 0.15	7.84 ± 0.29	8.22 ± 0.23
CW LOSS	59.82 ± 1.36	48.40 ± 3.24	19.25 ± 1.15	43.40 ± 2.76	45.62 ± 4.87
FEATURE EXPLOSION LOSS	82.41 ± 0.38	83.26 ± 0.94	78.53 ± 0.49	81.83 ± 0.46	82.30 ± 0.88

A.7 NETWORK TRANSFERABILITY

Table 8: **Results varying the *crafting* network for the poisons.** All poisons crafted with bound $\varepsilon = 8/255$, PGD, differentiable data augmentation.

CRAFTING \TESTING	VGG19	RESNET-18	GOOGLENET	DENSENET121	MOBILENETV2
RESNET-18	10.98 ± 0.27	6.25 ± 0.17	7.03 ± 0.12	7.16 ± 0.16	6.11 ± 0.17
RESNET-50	17.86 ± 0.57	9.71 ± 0.12	11.44 ± 0.21	10.64 ± 0.46	6.82 ± 0.18
VGG19	20.88 ± 0.84	18.53 ± 1.15	21.48 ± 0.67	23.77 ± 0.74	17.59 ± 0.56
MOBILENETV2	21.66 ± 0.26	12.42 ± 0.17	14.84 ± 0.29	15.95 ± 0.18	9.60 ± 0.24
CONVNET	15.43 ± 0.34	10.05 ± 0.20	11.88 ± 0.17	10.30 ± 0.11	7.95 ± 0.26