MetaPoison: Learning to Craft Adversarial Poisoning Examples via Meta-Learning

W. Ronny Huang*
University of Maryland
wrhuang@umd.edu

Jonas Geiping*
University of Siegen
jonas.geiping@uni-siegen.de

Liam Fowl
University of Maryland
lfowl@math.umd.edu

Tom Goldstein
University of Maryland
tomg@cs.umd.edu

Abstract

We consider a new, general method for *data poisoning* attacks on neural networks, in which the attacker takes control of a model by making small perturbations to a subset of its training data. We formulate the task of finding poisons as a bi-level optimization problem, which can be solved using methods borrowed from the meta-learning community. Unlike previous poisoning strategies, the meta-poisoning can poison networks that are trained *from scratch* using an initialization unknown to the attacker.

1 Introduction

Deep neural networks are increasingly deployed in high stakes applications like self driving cars, and medical diagnoses (Santana and Hotz [2016], Lee et al. [2017]). However, neural networks are vulnerable to *adversarial attacks* in which inputs to networks are maliciously modified to cause misclassification (Szegedy et al. [2013], Goodfellow et al. [2014]). Most research on ML security to date has focused on evasion attacks, in which the attacker manipulates classifier inputs at test time. In contrast, *data poisoning* is an emerging threat model in which the attacker manipulates training data with the goal of eliciting a chosen outcome from a classifier trained on that data (Shafahi et al. [2018], Zhu et al. [2019]). Unlike evasion attacks, data poisoning poses a threat in realistic situations wherein the attacker may not have access to test-time data, but can leave malicious data online and wait for it to be scraped by a bot or social media site.

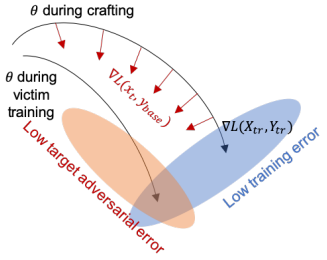


Figure 1: Schematic of the poisoning process in weight space. Poisons are crafted to adjust weight trajectories within the low training loss basin so that the network misclassifies the target image.

Classical poisoning attacks generally work by degrading overall model performance (Steinhardt et al. [2017]). In neural networks, the new class of *targeted clean-label* poisoning attacks are arguably more sinister. These attacks change the class label of a specific target input selected by the attacker, while leaving behavior on other inputs intact. The attacker achieves this by perturbing a small fraction of the training inputs, while leaving their labels untouched. Hence, the victim cannot know that their model is compromised by looking at the inputs (given the perturbation is small) nor by observing test set performance, since only the target image is compromised. While these attacks have potentially deleterious effects, their scope has been limited. Currently targeted attacks only work in the case of

*Equal contribution

Workshop on Meta-learning (spotlight), 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

transfer learning, in which a pre-trained model is fine-tuned on a small dataset (Shafahi et al. [2018], Zhu et al. [2019]).

Crafting clean-label examples has thus far depended on human-ingenuity, with techniques such as *feature collision* (Shafahi et al. [2018]) and *convex polytope* (Zhu et al. [2019]) which are based on heuristics. For example, feature collision is based on the assumption that if the poison is close enough to the target in representation space, then the decision boundary will wrap around the poison, inadvertently including the target onto the poison side of the decision boundary. While such heuristics have been successful, they apply only to certain scenarios (e.g. transfer learning) and leave telltale signs that poisoning has occurred (e.g. poison example has representation vector close to the target class examples).

Heuristics have thus far been necessary because it is difficult to map how perturbing a few inputs in the training set will affect the trained model behavior on new inputs. Inspired by meta-learning techniques, we devise a new technique called meta-poisoning which divests from reliance on heuristic-based poison crafting to directly *learning* to craft poisons. More specifically meta-poisoning directly optimizes the bi-level poisoning problem (3) by unrolling network computation graphs to discover how perturbations to training images impact SGD updates to network parameters. Our results using meta-poisoning achieve near 100% success rate on clean-label attacks using a poison budget of 4% in the challenging context of from-scratch training (not transfer learning) of a victim network, a more realistic scenario which has not been successfully attacked before.

2 Method

2.1 Clean-label poisoning threat model

We consider an attacker who wishes to force a target image x_t of her choice to be assigned label y_{source} by the victim model. She has the ability to perturb the training set by adding $\Delta \in [-\epsilon, \epsilon]^{N \times M}$ where ϵ is a small value, N is the total number of examples, and M is the dimensionality of the examples. We limit the attacker to being able to perturb only a small number of examples n, where n/N is typically a few percent. Outside of these n examples, the corresponding rows in Δ are zero, meaning no perturbation is made. The optimal perturbation Δ^* is then

$$\Delta^* = \underset{\|\Delta\|_\infty < \epsilon}{\operatorname{argmin}} \; L_{\theta^*(\Delta)}(x_t, y_{\text{source}}), \tag{1}$$

Algorithm 1 *Crafting* poisoned images via Meta-Learning

Input N The training set of images and labels $(\mathbf{X_{tr}}, \mathbf{Y_{tr}})$, target image x_t and intended target label y_source, ϵ threshold, $n < N$ subset of images to be poisoned. T budget of training epochs. M randomly initialized models.
Stagger the M given models, training the t-th model up to $\lfloor tM/T \rfloor$ epochs.
Select n images from the training set to be poisoned, denoted by $\mathbf{X_p}$.
Begin
  For $i = 1, \ldots, k$ unrolling steps and for M models do:
    $\theta_M^{i+1} = \theta_M^i - \tau \nabla L_{\theta_M^k}(\mathbf{X_{tr}} \cup \mathbf{X_p}, \mathbf{Y_{tr}})$
  Average target losses for all models: $L_{\text{total}} = \sum_{j=1}^{M} L_{\theta^k}(x_t, y_t)$
  Find $\nabla_{\mathbf{X_p}} L_{\text{total}}$ by backpropagation
  Update $\mathbf{X_p}$ by first-order Opt. (e.g. Adam or signGD) and project onto Δ_ϵ
  Update all models by taking a first-order step minimizing $L_{\theta_M}(\mathbf{X_{tr}}, \mathbf{Y_{tr}})$.
  Reset models that have reached T epochs to initialization
  STOP if maximal number of crafting steps reached or $\mathbf{X_p}$ is converged.
Return $\mathbf{X_p}$

where $L_\theta(x, y)$ is a loss function used to measure how well a model with parameters θ assigns label y to image x, and $\theta^*(\Delta)$ are the network parameters achieved by training on the perturbed training data $X + \Delta$. Note that (1) is a bi-level optimization problem (Bard [2013]) – the minimization for Δ

involves the parameters $\theta(\Delta)$, which are themselves the minimizer of the training problem

$$\theta^*(\Delta) = \underset{\theta}{\operatorname{argmin}}\ L_\theta(X_{tr} + \Delta, Y_{tr}). \tag{2}$$

The formulation (1) considers the training of a single network with a single initialization. To produce a *transferable* poison attack that is robust to changes in the initialization and training process, we average over M surrogate loss functions (2), randomly initialized and independently trained, leading to the final optimization objective of

$$\Delta^* = \underset{\Delta<\epsilon}{\operatorname{argmin}}\ \frac{1}{M}\sum_{i=0}^{M} L_{\theta_i^*(\Delta)}(x_t, y_{\text{source}}), \tag{3}$$

Finding the global minimum of this objective over many model instances is intractable using conventional bi-level methods. For example, each iteration of the direct gradient descent strategies as discussed in (Colson et al. [2007]) for poisoning a simple SVM models as in (Biggio et al. [2012]) would require minimizing the full training objective (2) for all surrogate models as well as computing the inverse of the parameter Hessian matrix.

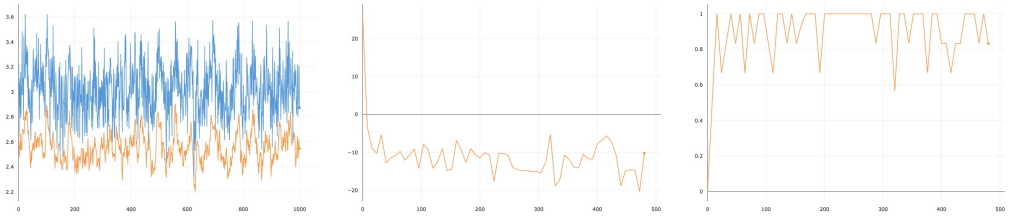
Rather than directly solving the bi-level problem, we take a page from the meta-learning playbook (Finn et al. [2017]). In meta-learning, one tries to find model parameters that yield the lowest possible loss *after* being updated by a step of SGD on a randomly chosen task. For meta-poisoning, we are interested in choosing a set of parameters (here, Δ) that will minimize the poisoning loss *after* applying an SGD update with a randomly chosen model. In both cases, this can be done by unrolling and back-propagating through a complete step of the training process.

2.2 MetaPoison: learning to *craft*

Intuitively, MetaPoison can be described by Figure 1. Instead of minimizing the full bi-level objective, we settle for a "first-order" approximation that only looks ahead one step in the training process. We run natural training, following the parameters θ toward low training error. After each SGD update, we ask *how can we update the poison perturbation Δ so that, when added to the training images for a single SGD step, it causes the most severe impact?*

We answer this question by (i) applying the poison perturbation to the training images, (ii) computing a parameter update to θ with respect to the training loss, (iii) passing the target image x_t through the resulting model, and (iv) evaluating the target adversarial loss $L(x_t, y_{\text{source}})$ on the target image x_t to measure the discrepancy between the network output and adversarially chosen label. Finally, we back-prop through this entire process to find the gradient of $L(x_t, y_{\text{source}})$ with respect to the data perturbation Δ. This gradient is used to refine Δ . This process is depicted in Figure 1.

In practice, we insert the poisons randomly into the training set, and form a random mini-batch of data on each SGD update. An image perturbation is updated only when the corresponding image is selected to be part of the minibatch. The SGD trajectories during the poison crafting process are computing using only clean images. At test time, the poisoned images are inserted into an unknown

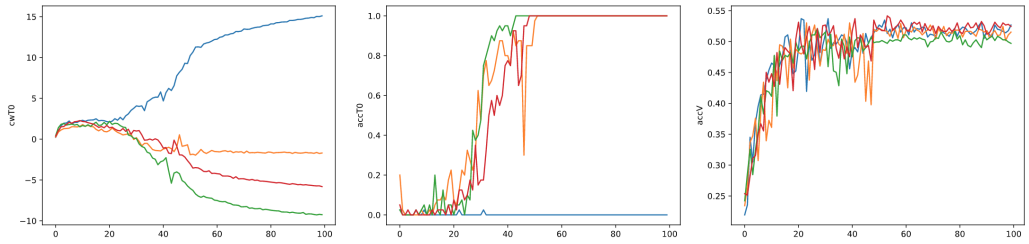


(a) Carlini-Wagner loss on target example during crafting process. Red: average loss over all unroll steps (crafting objective). Blue: Loss of the target example at the current network weights.

(b) Carlini-Wagner loss on target example at the end of 100 steps of victim training, using the poisons at the current craft step (x-axis).

(c) Poisoning success rate at the end of 100 steps of victim training using the poisons at the current craft step (x-axis).

Figure 2: Training from scratch with 1/10 of the CIFAR10 training set.



(a) Carlini-Wagner loss on target example during victim network training. Different colors represent different victim networks with random initialization.

(b) Poisoning success rate of the individual victim networks as a function of epoch.

(c) Validation accuracy during victim training. Different colors represent different victim networks with random initialization.

Figure 3: Training curves of the individual victim networks.

"victim" model, and the training now follows a new trajectory with a new random initialization. If the poisons succeed, this trajectory would lead to a minimum with both low training error and low target adversarial error.

This approach is supplemented by repeating the gradient computation over N_{models} surrogate models. Before we begin the poisoning process we pre-train these (independently initialized) surrogate models to varying epochs, so that every poison update sees models from numerous stages of training. Once a model reaches a sentinel number of epochs, it is reset to epoch 0 with a new random initialization. Akin to meta-learning, which samples new tasks with each meta-step, meta-poison "samples" new model parameters along training trajectories for each craft step. Algorithm 1 summarizes the details.

3 Experiments

We perform our training and poison crafting on a subset of CIFAR-10 which contains the first 500 examples from each class. In the appendix we show analogous results for the case of the entire CIFAR-10. We choose the first image in the test set, a "cat," to be the target image, and we choose the first class in the training set, the "airplane" class, to be the adversarial label. We allow for a poison budget of $n/N = 4\%$, which corresponds to $n = 200$ poisons, which are taken to be the first 200 images in the source class. The model architecture is a simple 6-layer convolutional network, akin to the one used in MAML (Finn et al. [2017]).

We use the Carlini-Wagner loss (Carlini and Wagner [2017]) as out objective function during poison crafting and make poison updates using the Adam optimizer. We project onto the constraint $\|\Delta\|_\infty < \epsilon$ after each gradient update, with $\epsilon = 16/255$. Figures 2 show the results of crafting poisons based on an ensemble of $N_{models} = 20$ surrogate models.

The resulting poisoned images can be seen in Figure 4. Figure 3 shows that inserting the poison into training successfully changes the label of the targeted image for nearly 100% of initializations, without affecting the validation accuracy of the other training images, making the attack imperceptible without knowledge of the target image or detection of the adversarial pattern.

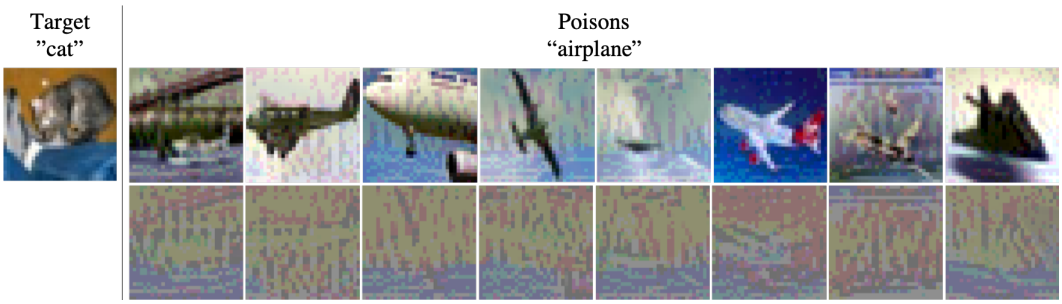


Figure 4: Visuals of the target image (left) and a random sampling of the poisons (top) along with their perturbations (bottom).

4 Conclusion

We have extended learning-to-learn techniques to adversarial poison example generation, or *learning-to-craft*. We devised a novel fast algorithm by which to solve the bi-level optimization inherent to poisoning, where the inner training of the network on the perturbed dataset must be performed for every crafting step. Our results, showing the first clean-label poisoning attack that works on networks trained from scratch, demonstrates the effectiveness of this method. We hope that our work establishes a strong attack baseline for future work on data poisoning.

References

Eder Santana and George Hotz. Learning a driving simulator. *arXiv preprint arXiv:1608.01230*, 2016.

June-Goo Lee, Sanghoon Jun, Young-Won Cho, Hyunna Lee, Guk Bae Kim, Joon Beom Seo, and Namkug Kim. Deep learning in medical imaging: general overview. *Korean journal of radiology*, 18(4):570–584, 2017.

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.

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

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. In *Advances in Neural Information Processing Systems*, pages 6103–6113, 2018.

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, 2019.

Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. In *Advances in neural information processing systems*, 2017.

Jonathan F. Bard. *Practical Bilevel Optimization: Algorithms and Applications*. Springer Science & Business Media, March 2013. ISBN 978-1-4757-2836-1.

Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153(1):235–256, June 2007. ISSN 0254-5330, 1572-9338. doi: 10.1007/s10479-007-0176-2.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning Attacks against Support Vector Machines. *arXiv:1206.6389 [cs, stat]*, June 2012. URL `http://arxiv.org/abs/1206.6389`.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.