Adversarial Attacks are a Surprisingly Strong Baseline for Poisoning Few-Shot Meta-Learners

Anonymous Author(s)
Affiliation
Address
email

Abstract

This paper examines the robustness of deployed few-shot meta-learning systems when they are fed an imperceptibly perturbed few-shot dataset. We attack amortized meta-learners, which allows us to craft colluding sets of inputs that are tailored to fool the system's learning algorithm when used as training data. Jointly crafted adversarial inputs might be expected to synergistically manipulate a classifier, allowing for very strong data-poisoning attacks that would be hard to detect. We show that in a white box setting, these attacks are very successful and can cause the target model's predictions to become worse than chance. However, in opposition to the well-known transferability of adversarial examples in general, the colluding sets do not transfer well to different classifiers. We explore two hypotheses to explain this: "overfitting" by the attack, and mismatch between the model on which the attack is generated and that to which the attack is transferred. Regardless of the mitigation strategies suggested by these hypotheses, the colluding inputs transfer no better than adversarial inputs that are generated independently in the usual way.

1 Introduction

Standard deep learning approaches suffer from poor sample efficiency [Krizhevsky et al., 2012] which is problematic in tasks where data collection is difficult or expensive. Few-shot learners have been developed to address this shortcoming by supporting rapid adaptation to a new task using only a few labeled examples [Finn et al., 2017, Snell et al., 2017]. This success has made few-shot learners more attractive for increasingly sensitive applications where the repercussions of confidently-wrong predictions are severe, such as clinical risk assessment [Sheryl Zhang et al., 2019], glaucoma diagnosis [Kim et al., 2017] and diseases identification in skin lesions [Mahajan et al., 2020].

As few-shot learners gain popularity, it is essential to understand how robust they are and how they may be exploited. It is well known that standard classifiers are vulnerable to *adversarial* inputs which have been purposefully and imperceptibly modified to cause incorrect predictions [Biggio and Roli, 2017]. Such examples may be presented to a model either at test time, called *evasion attacks* [Biggio et al., 2017] or *adversarial examples* [Szegedy et al., 2014], or at training time, which is referred to as *poisoning* [Newsome et al., 2006, Rubinstein et al., 2009]. The concepts of evasion and poisoning can be re-stated in terms of few-shot learners. At meta-test time, a few-shot learner is presented with an unseen task containing a few labeled examples, the *support set*, and a number of unlabeled examples to classify, called the *query set*. We can perpetrate a "standard" adversarial attack by perturbing an image in the query set or perform a poisoning attack by perturbing images in the support set.

In this paper, we propose a poisoning attack against amortized meta-learners that *jointly* optimizes attack images *through the meta-learner's adaptation mechanism*. We produce a colluding *set* of adversarial images that is explicitly tailored to fool the learning algorithm. We consider as a baseline the insertion of standard adversarial attack images into the support set. We expect our attack to

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

perform better than this baseline, since standard adversarial attacks are optimised *individually* to cause misclassification by a fixed classifier (rather than being curated as a set to fool the adaptation mechanism itself). We may also expect the new attack to transfer to other learners, especially considering the well-known transferability of adversarial examples in literature Goodfellow et al. [2014], Papernot et al. [2016a], Szegedy et al. [2014]. Our contributions are as follows:

1. We define a poisoning attack on few-shot classifiers, referred to as an **adversarial support attack** or ASP. This applies coordinated adversarial perturbations to produce a colluding set of inputs, generated by jointly backpropagating through the meta-learner's adaptation process to minimize model accuracy over a set of query points. To the best of the authors' knowledge, ours is the first work to perpetrate poisoning attacks on trained few-shot classifiers.
2. In a white box setting ASP is a very strong attack, far more effective than the baseline.
3. Surprisingly, we show that ASP attacks do not transfer significantly better than the **swap attack** baseline, which inserts adversarial examples generated "in the usual way" into the support set.
4. We observe that transferability of support attacks are highly dependent on the feature extractor used by the meta-learner. Although we propose various mitigation strategies to improve the transferability of support set attacks, we do not succeed at significantly improving over the efficacy of a swap attack, which is simpler and cheaper.

The last two findings are unexpected: we expect the ASP attack to be significantly stronger than the swap attack baseline because it is tailored to fool the learner's adaptation process, is optimized over a set of query points and is able to collude within the adversarial set.

2 Background

We focus on image classification, denoting input images by $x \in \mathbb{R}^{ch \times W \times H}$ where W is the image width, H the image height, ch the number of image channels and image labels $y \in \{1, \ldots, C\}$ where C is the number of image classes. We use bold $\boldsymbol{x}$ and $\boldsymbol{y}$ to denote sets of images and labels. We consider the few-shot image classification scenario using a meta-learning approach. Rather than a single, large dataset D, we assume access to a dataset $\mathcal{D} = \{\tau_t\}_{t=1}^{K}$ comprising a large number of training *tasks* τ_t, drawn i.i.d. from a distribution $p(\tau)$. An example task is shown in Fig. 4. The data for a task consists of a *support set* $D_S = \{(x_n, y_n)\}_{n=1}^{N}$ comprising N elements, with the inputs x_n and labels y_n observed, and a *query set* $D_Q = \{(x_m^*, y_m^*)\}_{m=1}^{M}$ with M elements for which we wish to make predictions. We may use the shorthand $D_S = \{\boldsymbol{x}, \boldsymbol{y}\}$ and $D_Q = \{\boldsymbol{x}^*, \boldsymbol{y}^*\}$. The meta-learner g takes as input the support set D_S and produces task-specific classifier parameters $\boldsymbol{\psi} = g(D_S)$ which are used to adapt the classifier f to the current task. The classifier can now make task-specific predictions $f(x^*, \boldsymbol{\psi} = g(D_S))$ for any test input $x^* \in D_Q$. Here the inputs x^* are observed and the labels y^* are only observed during meta-training (i.e. training of the meta-learning algorithm). Note that the query set examples are drawn from the same set of labels as the examples in the support set. The majority of modern meta-learning methods employ *episodic* training [Vinyals et al., 2016], as detailed in A.1. At meta-test time, the classifier f is required to make predictions for query set inputs of unseen tasks, which will often include classes that have not been seen during meta-training, and D_S will contain only a few observations.

The canonical example for modern gradient-based few-shot learning systems is MAML [Finn et al., 2017]. Another widely used class of meta-learners are *amortized-inference* or *black box* based approaches e.g, VERSA [Gordon et al., 2019] and CNAPS [Requeima et al., 2019a]. In these methods, the task-specific parameters $\boldsymbol{\psi}$ are generated by one or more *hyper-networks*, g [Ha et al., 2016]. Prototypical Networks (ProtoNets) [Snell et al., 2017] is a special case of this which is based on *metric* learning and employs a nearest neighbor classifier. We focuses on Simple CNAPS Bateni et al. [2020] – which improves on the performance of CNAPS – and uses a ProtoNets classifier head with an adaptive feature extractor like that of CNAPS; but we have also considered attacks against MAML, ProtoNets, and CNAPS with similar outcomes.

3 Attacking Few-Shot Learners

We summarize the threat model in terms of the adversary's goal, capabilities and knowledge. In this work, we develop poisoning attacks that degrade the model *availability* (i.e. affect prediction results

indiscriminately such that it is useless) [Jagielski et al., 2018]. We allow the attacker to manipulate some fraction of the support set and further constrain pattern modifications to be imperceptible (i.e. within some ϵ of the original image, measured using the ℓ_∞ norm). When considering transfer attacks, we call the model used generate the attack the *surrogate* and the model to which we transfer the attack the *target*. We presume access to the surrogate model's gradients and internal state. We don't assume any access to the target model's internal state, though some of the experiments do access a limited number of the target's predictions. We also assume access to enough data to form a query set.

Consider a target model that has been trained and deployed as a service. As described in Section 1, a malicious party could perpetrate two kinds of attacks: a "standard" adversarial attack (hereafter referred to as a *query* attack) by perturbing elements in the query set of a task; or perform a poisoning attack by perturbing elements in the support set. We expand on these below:

Query Attack Given a meta-learner that has already adapted to a specific task, the attacker perturbs a *single* test input x^* in such a way that *it* will be misclassified. This corresponds to solving $\arg\max_\delta \mathcal{L}(f(x^* + \delta, g(\boldsymbol{x}, \boldsymbol{y})), y^*)$. Refer to Appendix A.6 for details. These attacks are essentially evasion attacks as considered in Biggio et al. [2017] and many algorithms can be used to generate adversarial examples [Madry et al., 2017, Carlini and Wagner, 2017, Chen et al., 2017]. Query attacks have been perpetrated successfully against few-shot learners [Goldblum et al., 2019, Yin et al., 2018].

Adversarial Support Attack (ASP) The attacker perturbs the *dataset* that the target meta-learner will learn from so as to influence *all future test predictions*. The attacker thus computes a perturbed support set $\tilde{D}_S = \{\tilde{\boldsymbol{x}}, \boldsymbol{y}\}$ whose inputs are jointly optimized to fool the system on a specific query set, which we call the *seed query set*, with the goal of generalizing to unseen query sets. This corresponds to solving $\arg\max_{\boldsymbol{\delta}} \mathcal{L}(f(\boldsymbol{x}^*, g(\boldsymbol{x} + \boldsymbol{\delta}, \boldsymbol{y})), \boldsymbol{y}^*)$ such that $\|\boldsymbol{\delta}\|_\infty < \epsilon$, where $D_Q = \{\boldsymbol{x}^*, \boldsymbol{y}^*\}$ denotes the seed query set and ϵ is the maximum size of the perturbation. Refer to Algorithm 1 for details. Our attack is a poisoning attack, since the attacker is manipulating data that the model will use to do inference. However, it is important to note the the attack is perpetrated at meta-test time, after the meta-learner has already been meta-trained. Unlike a query attack, which is generated by backpropagating through the adapted learner with respect to a single point, our attack backpropagates jointly through the adaptation process with respect to the loss on the entire query set and should thus be a much stronger poisoning attack than say, inserting a query point into the support set. Without loss of generality, we use Projected Gradient Descent (PGD) [Madry et al., 2017] to generate perturbed support sets because it is effective, simple to implement, and easily extensible to sets of inputs.

Algorithm 1 PGD for ASP

Require:
I_{min}, I_{max}: Min/Max image intensity, γ: Step size, L: Number of iterations,
$D_S \equiv \{\boldsymbol{x}, \boldsymbol{y}\}, D_Q \equiv \{\boldsymbol{x}^*, \boldsymbol{y}^*\}, \mathcal{L} \equiv$ cross-entropy, ϵ: Perturbation amount

procedure PGDS(D_S, D_Q, f, g)
  $\boldsymbol{\delta} \sim U(-\epsilon, \epsilon)$
  $\tilde{\boldsymbol{x}} \leftarrow \text{clip}(\boldsymbol{x} + \boldsymbol{\delta}, I_{min}, I_{max})$
  for $i \in 1, ..., L$ **do**
    $\boldsymbol{\delta} \leftarrow \text{sgn}(\nabla_{\tilde{\boldsymbol{x}}} \mathcal{L}(f(\boldsymbol{x}^*, g(\tilde{\boldsymbol{x}}, \boldsymbol{y})), \boldsymbol{y}^*))$
    $\tilde{\boldsymbol{x}} \leftarrow \text{clip}(\tilde{\boldsymbol{x}} + \gamma\boldsymbol{\delta}, I_{min}, I_{max})$
    $\tilde{\boldsymbol{x}} \leftarrow \boldsymbol{x} + \text{clip}(\tilde{\boldsymbol{x}} - \boldsymbol{x}, -\epsilon, \epsilon)$
    return $\tilde{\boldsymbol{x}}$

4 Experiments

The experiments presented in the main body of the paper are carried out on Simple CNAPS [Bateni et al., 2020] using the challenging META-DATASET benchmark (refer to Appendix A.2 and for details on the meta-data training protocols and META-DATASET). The Simple CNAPS model uses a ProtoNets head in conjunction with an adaptive feature extractor. The ProtoNets head uses Euclidean distance and the feature extractor is a Resnet18 endowed with Feature-wise Linear Modulation (FiLM) layers [Perez et al., 2018], which scales and shifts a feature map $\mathbf{x}$ by $\zeta\mathbf{x} + \beta$. In the meta-learning setting, FiLM parameters are generated by an adaptation network as part of $\boldsymbol{\psi}$, the

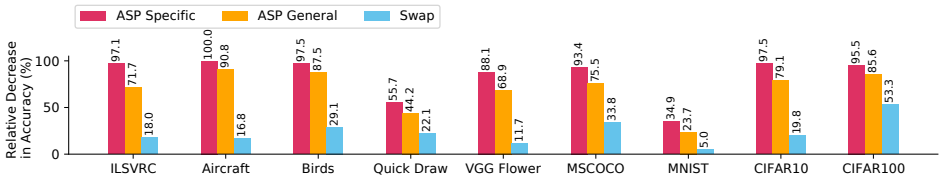

Figure 1: Relative drop in model accuracy attacking Simple CNAPS on META-DATASET with $\epsilon = 0.05$, $\gamma = 0.0015$, $L = 100$, with all classes, but only 20% of the shots poisoned.

learner's adaptation to a new task, and allows the feature extractor to adapt flexibly to each new task with relatively few additional parameters [Requeima et al., 2019a]. In META-DATASET, task support sets may be large — up to 500 images across all classes. In a realistic scenario, an attacker would not likely be able to perturb all the images in such a large support set, so we only perturb a specified fraction of the support set in each experiment. Each task is composed of a support set, a seed query set and up to 50 unseen query sets used for attack evaluation (some META-DATASET benchmarks do not have sufficiently many patterns available to form 50 query sets and thus were excluded from the experiments). The unseen query sets are all disjoint from the seed query set to avoid information leakage. For each task, we generate an adversarial support set using the original support set and corresponding seed query set. The adversarial support set is then evaluated on the task's unseen query sets. We refer to the average classification accuracy on the seed query sets as the ASP *Specific* attack accuracy, and when evaluating the attack on unseen query sets we refer to it as the ASP *General* attack accuracy. An important baseline is the **swap attack**, which is generated by using the task's support set and query set to produce a query attack, then "swap" the role of the adversarial query set by using it as a support set. Like our ASP attack, the swap attack is evaluated on unseen query sets. Note that the swap attack is far less expensive to compute than the ASP attack, since it only requires backpropagation through the adapted learner and not through the entire meta-learning process.

4.1 White Box Attack

We first consider the simplest scenario, a white box attack in which the surrogate and target model are the same. Fig. 1 shows the relative decrease in accuracy of Simple CNAPS on META-DATASET. In general, the clean accuracy of the surrogate and target models may be quite different and so we normalise the effect an attack has by computing the percentage relative decrease in classification accuracy as follows: $100\% \times (a_{clean} - a_{attack})/a_{clean}$ where a_{clean} is the *clean* classification accuracy before the attack, and a_{attack} is the classification accuracy after the attack. Unnormalised results are in A.5. Although it is expected for ASP *Specific* to be very effective, the ASP attack remains highly effective even on unseen query sets (see ASP *General*), easily out-performing the *Swap* baseline, in spite of the fact that only 20% of the support set shots are poisoned. At least in the white box setting, ASP is very strong attack.

4.2 Attack Transfer

We now turn our attention to the more difficult scenario, in which the target and surrogate models are different. We fix our surrogate model as Simple CNAPS with a particular Resnet18 feature extractor (hereafter referred to Resnet18 A). We consider a Simple CNAPS model with a slightly different feature extractor: Resnet18 B, which was trained on the same data with similar training conditions, but with no FiLM parameters. We also consider turning the FiLM parameters for Resnet18 A on and off. Although this seems a rather trivial extension, the results shown in Fig. 2 indicate that the ASP attack no longer outperforms the *Swap* baseline by a statistically significant margin. Fig. 2 considers three scenarios in increasing order of difficulty. For Scenario A, the perturbation size is very large ($\epsilon = 1$) and the entire support set has been adversarially perturbed. At this scale, tampering could be clearly inferred by inspection and it is unlikely that an adversary would have control over the entire support set, so this is the weakest scenario. In this setting, we see good transfer between the surrogate model, Resnet18 A with FiLM, and the target models (Resnet18 A without FiLM and Resnet18 B without FiLM). Scenario B reduces the perturbation size to be mostly imperceptible ($\epsilon = 0.3$), but still perturbs the entire support set. Here, we start to see a reduction in attack effectiveness when transferring to Resnset18 B. In Scenario C, we use $\epsilon = 0.3$ and also reduce the proportion of poisoned

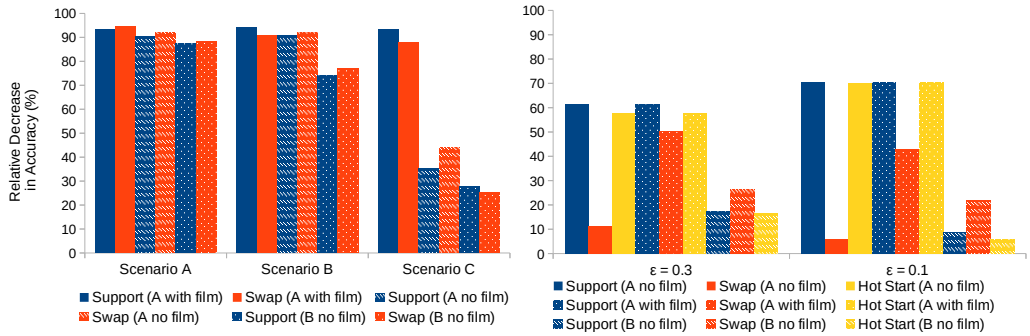

Figure 2: Transfer to Simple CNAPS with different backbones. Attacks on the surrogate model are solid fill; transfer attacks have diagonal hatching for Resnset18 A with FiLM and dots for Resnet18 B no FiLM. **(Left)** Scenario A: $\epsilon = 1.0$ with entire support set adversarial; Scenario B: $\epsilon = 0.3$ with entire support set adversarial; Scenario C: $\epsilon = 0.3$ with 50% of support set adversarial. Dataset is CIFAR-100; $L = 100$. **(Right)** ASP, hot-started ASP and swap attacks on ILSVRC 2012; $L = 50$.

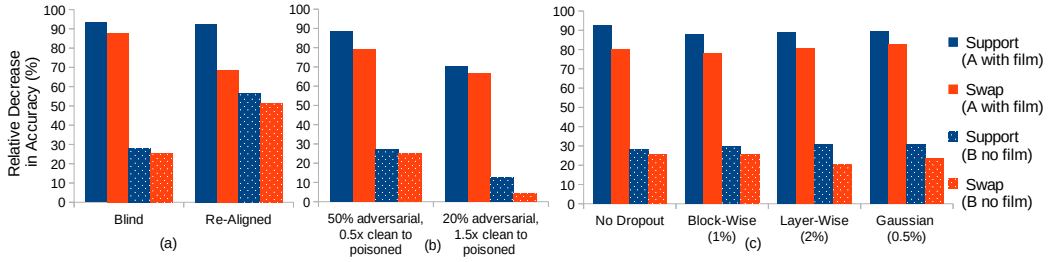

Figure 3: Mitigation strategies for transfer attacks on CIFAR-100. Solid fill indicates attacks on the surrogate (Resnset18 A with FiLM); dots indicate transfer attacks to Resnet18 B no FiLM. Support set is 50% adversarial, $\epsilon = 0.3$, $\gamma = 0.0167$, $L = 100$. (a) In *Blind*, the surrogate has no knowledge of the target's decision boundaries; in *Re-Aligned* the support set presented to the surrogate has been relabeled to match the target's predictions. (b) Clean support images are varied during attack generation. (c) Effect of different dropout strategies.

samples in the support set to 50%. Here we see significant reduction in transfer, even to Resnet18 A without FiLM. We also note that unlike the white box attack, we do not perform significantly better than the *Swap* attack, and possibly perform a little worse. We know from literature that adversarial attacks transfer well and should thus be fairly robust to the kind of feature extractor being used. Since we're only changing the feature extractor and keeping the type of learner the same, we expect the support attacks, which are tailored to fool the learning mechanism, to have a benefit over the swap attack. However, this is not borne out in experiments. We consider two avenues of explanation that account for the poor transfer between feature extractors: mismatch between the target and surrogate models, and "overfitting" of the attack to the support set and/or surrogate model. We consider a number of mitigation strategies intended to address these two problems below:

Decision Boundary Alignment Since the target and surrogate models have different feature extractors, it may be that their embedding spaces and consequently also their decision boundaries are very different. We can try to "re-align" the surrogate model's decision boundaries by training it to produce the same predictions as the target model, an approach which has been used in literature when perpetrating transfer attacks [Papernot et al., 2016b]. In a meta-learning setting, this corresponds to relabeling the support set presented to the surrogate model based on the target model's predictions. The labels of the support set when presented to the surrogate model will thus not be the true labels, but rather the labels assigned to the support images by the target model. The relabeled support set is also used to generate the swap attack so that the comparison is fair. As shown in Fig. 3(a), relabeling the support set does significantly improve transfer compared to the "blind" scenario in which the

surrogate model has no knowledge about the target model's decision boundaries. However, ASP is still not significantly better than the swap attack.

Hot Start The ASP optimization procedure may be finding a different minimum, which is less robust to different feature extractors than that found by the swap attack. We propose to *hot-start* the ASP attack, where given a support set D_S, we spend half the optimization budget performing a query attack to produce D'_S, which is then used as input to an adversarial support set attack to produce $\tilde{D}_S$ with the other half of the budget. In this way, we start the ASP attack off in the region of the local minimum found by the query attack. However, as shown in Fig. 1, the hot start does not improve on the normally initialized ASP attack and may be worse in some scenarios.

Support Set Shuffling Simple CNAPS makes use of a set encoder to produce a representation of the entire support set and so the attack must effectively manipulate the poisoned examples so that the embedded representation of the poisoned support set is sufficiently different to affect predictions. This may explain why we observe that the poisoning attack requires a high proportion of poisoned examples in the support set in order to be effective. The potency of the poisoned examples thus depend on the rest of the (unperturbed) support set. When transferred, the support set may embed very differently, especially if a large proportion of it is not adversarial, negating the effects of the poisoned points. In order to generate solutions that are less dependent on the rest of the support set we propose a procedure whereby the clean elements in the support set are not kept fixed for the duration of attack generation, but instead shuffle different clean images in and out of the support set. Fig. 3(b) considers two configurations: each varying the percentage of the support set that is poisoned (50% and 20%, respectively) and the proportion of clean images in the support set during attack generation (the clean : poisoned ratio is $0.5 : 1$ and $1.5 : 1$, respectively). Note that we are careful not to assume access to more data than any of the other scenarios. The first setting performs no better than an attack generated without shuffling. The second setting performs worse – although the attack has access to a more varied support set, this comes at the price of poisoning a smaller proportion of the support set since we cannot access additional data.

Dropout If the attack optimization procedure is overfitting to the surrogate model, then regularization by dropout is an established remedy. By applying dropout to the feature extractor, we may hope for the resulting attack to generalize better to new feature extractors. We drop out the FiLM parameters of the surrogate's feature extractor using one of three different strategies: (1) Block-Wise - the FiLM parameters for an entire Resnet block is either on/off, i.e. either computed as usual or we set $\zeta = 1.0$, $\beta = 0$; (2) Layer-Wise - each FiLM layer within a Resnet block may be switched on or off individually, i.e. compute ζ and β as usual, but apply a dropout mask; (3) Gaussian Dropout - the FiLM parameters are smoothed by a Gaussian with standard deviation given by $\sqrt{d/(1-d)}$ where d is the dropout probability. Although Fig. 3(c) shows slight improvement in transferability, the difference between the ASP and swap attacks remain statistically insignificant.

5 Conclusion

In this paper, we propose ASP, an attack against amortized meta-learners that jointly crafts sets of colluding points which can manipulate a meta-learner when used as training data. We expect ASP attacks to be strong because the attack images are generated by backpropagating through the learner's adaptation process with respect to the loss over an entire set of test inputs and are thus able to collude to fool the classifier. We may also expect ASP enjoy the same robustness to different feature extractors as a standard evasion attack. We show that although ASP is very effective in a white box setting, it does not transfer well, even to the same learner with a different feature extractor.

We explore a number of strategies to improve the transferability of ASP, but in all cases ASP was not able to transfer significantly better than the swap attack baseline (which performs a query or evasion attack in the usual way and simply swaps it into the meta-learner's support set). Even though the swap attack should be much weaker, since it is generated to fool the learner only for the image in question, it proves a very effective baseline for poisoning few-shot learners. Since an attacker will rarely have access to the internal workings of the target model, the transferability of an attack is an important consideration, which prevents the deployment of ASP in real-life. But future work may glean insight into making models more robust.

References

Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *CoRR*, abs/1712.03141, 2017. URL `http://arxiv.org/abs/1712.03141`.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Srndic, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion Attacks against Machine Learning at Test Time. *arXiv e-prints*, art. arXiv:1708.06131, August 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, May 2017. doi: 10.1109/SP.2017.49.

Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. EAD: Elastic-Net Attacks to Deep Neural Networks via Adversarial Examples. *arXiv e-prints*, art. arXiv:1709.04114, Sep 2017.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL `http://proceedings.mlr.press/v70/finn17a.html`.

Micah Goldblum, Liam Fowl, and Tom Goldstein. Adversarially robust few-shot learning: A meta-learning approach. *arXiv*, pages arXiv–1910, 2019.

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

Jonathan Gordon, John Bronskill, Matthias Bauer, Sebastian Nowozin, and Richard Turner. Meta-learning probabilistic inference for prediction. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HkxStoC5F7`.

David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *International Conference on Learning Representations*, 2016. URL `https://openreview.net/forum?id=rkpACe1lx`.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35, 2018.

Mijung Kim, Jasper Zuallaert, and Wesley De Neve. Few-shot learning using a small-sized dataset of high-resolution fundus images for glaucoma diagnosis. In *Proceedings of the 2nd International Workshop on Multimedia for Personal Health and Health Care*, MMHealth '17, page 89–92, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450355049. doi: 10.1145/3132635.3132650. URL `https://doi.org/10.1145/3132635.3132650`.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2:18, 2010.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv e-prints*, art. arXiv:1706.06083, June 2017.

Kushagra Mahajan, Monika Sharma, and Lovekesh Vig. Meta-dermdiagnosis: Few-shot skin disease identification using meta-learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

James Newsome, Brad Karp, and Dawn Song. Paragraph: Thwarting signature learning by training maliciously. In Diego Zamboni and Christopher Kruegel, editors, *Recent Advances in Intrusion Detection*, pages 81–105, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-39725-0.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. *arXiv e-prints*, art. arXiv:1605.07277, May 2016a.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical Black-Box Attacks against Machine Learning. *arXiv e-prints*, art. arXiv:1602.02697, February 2016b.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *Proceedings of the International Conference on Learning Representations*, 2017. URL `https://openreview.net/pdf?id=rJY0-Kcll`.

James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 7957–7968. Curran Associates, Inc., 2019a.

James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Code for "Fast and flexible multi-task classification using conditional neural adaptive processes". https://github.com/cambridge-mlg/cnaps, 2019b.

Benjamin I.P. Rubinstein, Blaine Nelson, Ling Huang, Anthony D. Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J. D. Tygar. Antidote: Understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, IMC '09, page 1–14, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605587714. doi: 10.1145/1644893.1644895. URL `https://doi.org/10.1145/1644893.1644895`.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Xi Sheryl Zhang, Fengyi Tang, Hiroko Dodge, Jiayu Zhou, and Fei Wang. MetaPred: Meta-Learning for Clinical Risk Prediction with Limited Patient Electronic Health Records. *arXiv e-prints*, art. arXiv:1905.03218, May 2019.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4077–4087. Curran Associates, Inc., 2017.

Christian Szegedy, Google Inc, Wojciech Zaremba, Ilya Sutskever, Google Inc, Joan Bruna, Dumitru Erhan, Google Inc, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *In ICLR*, 2014.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rkgAGAVKPr`.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3630–3638. Curran Associates, Inc., 2016.

Chengxiang Yin, Jian Tang, Zhiyuan Xu, and Yanzhi Wang. Adversarial meta-learning. *arXiv preprint arXiv:1806.03316*, 2018.

A Appendix

A.1 Episodic Training

There has been an explosion of meta-learning based few-shot learning algorithms proposed in recent years. For an in-depth review see Hospedales et al. [2020]. The majority of modern meta-learning methods employ *episodic* training [Vinyals et al., 2016]. During meta-training, a task τ is drawn from $p(\tau)$ and randomly split into a support set D_S and query set D_Q. Fig. 4 depicts an example few-shot classification task. The meta-learner g takes as input the support set D_S and produces task-specific classifier parameters $\boldsymbol{\psi} = g(D_S)$ which are used to adapt the classifier f to the current task. The classifier can now make task-specific predictions $f(x^*, \boldsymbol{\psi} = g(D_S))$ for any test input $x^* \in D_Q$. Refer to *Clean* in Fig. 5. A loss function $\mathcal{L}(f(x^*, \boldsymbol{\psi}), y^*)$ then computes the loss between the predictions for the label $f(x^*, \boldsymbol{\psi})$ and the true label y^*. Assuming that $\mathcal{L}$, f, and g are differentiable, the meta-learning algorithm can then be trained with stochastic gradient descent by back-propagating the loss and updating the parameters of f and g.

A.2 Few-shot Learner Meta-training Protocols

In the following, the meta-training protocols for the few-shot learners used in the experiments.

A.2.1 Datasets

***mini*ImageNet** *mini*ImageNet is a subset of the larger Imagenet dataset [Russakovsky et al., 2015] created by Vinyals et al. [2016]. It consists of 60,000 color images that is sub-divided into 100 classes, each with 600 instances. The images have dimensions of 84×84 pixels. Ravi and Larochelle [2017] standardized the 64 training, 16 validation, and 20 test class splits. *mini*ImageNet has become a defacto standard dataset for benchmarking few-shot image classification methods with the following classification task configurations: (i) 5-way, 1-shot; (ii) 5-way, 5-shot.

META-DATASET META-DATASET [Triantafillou et al., 2020] is composed of ten (eight train, two test) image classification datasets. We augment Meta-Dataset with three additional held-out datasets: MNIST [LeCun et al., 2010], CIFAR10 [Krizhevsky and Hinton, 2009], and CIFAR100 [Krizhevsky and Hinton, 2009]. The challenge constructs few-shot learning tasks by drawing from the following distribution. First, one of the datasets is sampled uniformly; second, the "way" and "shot" are sampled randomly according to a fixed procedure; third, the classes and support / query instances are sampled. Where a hierarchical structure exists in the data (ImageNet or Omniglot), task-sampling respects the hierarchy. In the meta-test phase, the identity of the original dataset is not revealed and the tasks must be treated independently (i.e. no information can be transferred between them). Notably, the meta-training set comprises a disjoint and dissimilar set of classes from

Figure 4: Example task with $C = 4$ classes with $N = M = 4$.

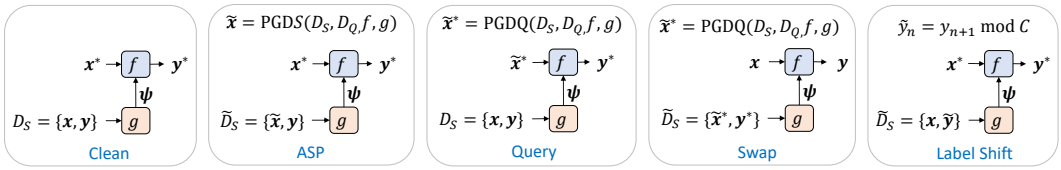

Figure 5: Attacks on meta-learning based few-shot image classifiers. f and g denote the classifier and trained meta-learner, respectively. Each diagram depicts how an attack is applied and includes an expression for the attack's computation using Algorithms 1 and 2.

those used for meta-test. META-DATASET is presently, the "gold standard" for evaluating few-shot classification methods. Full details are available in Triantafillou et al. [2020].

In our experiments, we excluded the Omniglot, Textures, Fungi, and Traffic Signs datasets from evaluation because their test splits are too small to allow for a fair assessment of the attack's generalization, even though the attacks reduced the classification accuracy on those datasets to approximately zero in the ASP *Specific* case.

For META-DATASET, we meta-trained Simple CNAPS using the code from Requeima et al. [2019b] with FiLM feature adaptation. We made modifications to the code to enable various adversarial attacks. The meta-trained model attained the following results:

ilsvrc 2012: 55.1 ± 1.1, omniglot: 90.8 ± 0.6, aircraft: 82.3 ± 0.6, cu birds: 74.0 ± 0.9, dtd: 63.4 ± 0.7, quickdraw: 75.3 ± 0.8, fungi: 44.6 ± 1.0, vgg flower: 90.3 ± 0.5, traffic sign: 67.9 ± 0.8, mscoco: 40.8 ± 1.0, mnist: 91.4 ± 0.5, cifar10: 72.5 ± 0.8, cifar100: 58.4 ± 1.0.

A.3 Attack Summary

We summarize the types of attacks we perform in Fig. 5. The first scenario, *Clean*, illustrates how the meta-learner g performs a test-time task, taking the support set D_S as input to produce parameters $\boldsymbol{\psi} = g(D_S)$ which are used to adapt the classifier f to the task. The classifier makes task-specific predictions $f(x^*, \boldsymbol{\psi} = g(D_S))$ for any test input $x^* \in D_Q$. *ASP* and *Query* illustrate the ASP and Query attacks as discussed in Section 3. *Swap* illustrates a swap attack, which is used as a baseline comparison for ASP, where a set of images are perturbed with a query attack and then inserted into the support set. Query attacks are typically cheaper to compute, since they do not require back-propagation through the meta-learner, so it is an important baseline to consider.

A.4 Additional Experimental Details

In our experiments, all the input images were re-scaled to have pixel values between -1 and 1. We considered perturbations using the ℓ_∞ norm, on a scale of $[-1, 1]$, so that $\epsilon = 0.1$ corresponds to allowing $\pm 10\%$ or an absolute change of ± 0.2 to the intensity of each pixel in an image.

We calculated the perturbation step size γ to depend on ϵ and the maximum number of iterations, so that $\gamma = r\frac{\epsilon}{L}$, where r is a scaling coefficient.

A.5 Unnormalised Large-Scale Attack Results

This section of the appendix presents the unnormalized results for all figures, along with 95% confidence intervals.

In Table 1 we present the unnormalized numbers for Fig. 1, which shows the efficacy of ASP in a white box setting. Table 2 considers transfer between a surrogate model and target models with different feature extractors as presented in Fig. 2*(Left)*. Tables 3 and 4 consider the effect of the hot-start mitigation strategy for different perturbation sizes as presented in Fig. 2*(Right)*. Tables 5, 6 and 7 correspond to Fig. 3 and show the effect of other mitigation strategies, specifically decision boundary re-alignment, shuffling different clean inputs into the support set, and dropout. All numbers are percentages and the $\pm$ sign indicates the 95% confidence interval.

Table 1: Accuracy of Simple CNAPS on the META-DATASET benchmark in the *Clean*, *Specific* and *General* scenarios when attacking with an adversarial support set, with $\epsilon = 0.05$, $\gamma = 0.0015$, $L = 100$, with all classes, but only 20% of the shots poisoned. All figures are percentages and the $\pm$ sign indicates the 95% confidence interval over 500 tasks.

	Clean	Specific	General	Swap
ilsvrc_2012	52.2±0.2	1.5±0.1	14.8±0.1	42.8±0.2
aircraft	78.5±0.5	0.0±0.0	7.2±0.2	65.3±1.0
cu_birds	71.4±1.1	1.8±0.2	8.9±0.4	50.6±2.0
quickdraw	74.1±0.1	32.8±1.2	41.3±0.2	57.7±0.3
vgg_flower	89.1±0.6	10.6±0.8	27.7±1.0	78.7±1.8
traffic_sign	35.2±0.4	4.1±0.3	12.3±0.3	24.8±0.5
mscoco	44.1±0.3	2.9±0.1	10.8±0.1	29.2±0.3
mnist	90.3±0.1	58.8±1.4	68.9±0.2	85.8±0.2
cifar10	64.6±0.1	1.6±0.1	13.5±0.1	51.8±0.2
cifar100	53.5±0.6	2.4±0.1	7.7±0.2	25.0±0.8

Table 2: Transfer from the surrogate model (Simple CNAPS using Resnet A with FiLM) to Simple CNAPS using alternative feature extractors on CIFAR-100. Scenario A: $\epsilon = 1.0$ with entire support set adversarial; Scenario B: $\epsilon = 0.3$ with entire support set adversarial; Scenario C: $\epsilon = 0.3$ with 50% of support set adversarial. Attacks are generated using PGD with $L = 100$. These results correspond to Fig. 2 (*Left*).

		Clean	ASP	Swap
Surrogate (Resnet A FiLM)	Scenario A	52.4±2.1	3.6±0.4	2.9±0.4
	Scenario B	54.2±1.2	3.2±0.2	4.9±0.3
	Scenario C	54.5±1.8	3.6±0.3	6.6±0.6
Transfer to Resnet A (no FiLM)	Scenario A	48.7±2.1	4.7±0.5	3.9±0.5
	Scenario B	50.2±1.2	4.6±0.3	4.0±0.3
	Scenario C	50.5±3.4	32.7±2.0	28.2±1.3
Transfer to Resnet B (no FiLM)	Scenario A	38.6±2.0	4.9±0.6	4.6±0.6
	Scenario B	39.1±1.2	10.2±0.6	9.0±0.5
	Scenario C	40.1±1.9	28.9±2.0	29.9±1.6

Table 3: Effect of hot-starting the ASP attack on transferability from the surrogate model (Simple CNAPS using Resnet A without FiLM) to Simple CNAPS using an alternative feature extractor on ILSVRC 2012. Attacks were generated using PGD with $\epsilon = 0.3$ and $L = 100$. These results correspond to Fig. 2 (*Right*) for $\epsilon = 0.3$.

	Clean	ASP	Swap	Hot Start
Surrogate (Resnet18 A no FiLM)	52.1±0.7	20.1±0.4	46.3±0.9	22.0±0.4
Transfer to Resnet18 A (with FiLM)	55.8±0.7	31.5±0.7	28.7±0.5	35.2±0.7
Transfer to Resnet18 B (no FiLM)	43.6±0.7	36.1±0.7	32.2±0.6	36.4±0.7

Table 4: Effect of hot-starting the ASP attack on transferability from the surrogate model (Simple CNAPS using Resnet A without FiLM) to Simple CNAPS using an alternative feature extractor on ILSVRC 2012. Attacks were generated using PGD with $\epsilon = 0.1$ and $L = 100$. These results correspond to Fig. 2 (*Right*) for $\epsilon = 0.1$.

	Clean	ASP	Swap	Hot Start
Surrogate (Resnet18 A no FiLM)	54.1±0.8	16.0±0.4	50.9±0.8	16.1±0.4
Transfer to Resnet18 A (with FiLM)	58.1±0.8	46.2±0.8	41.1±0.7	48.5±0.8
Transfer to Resnet18 B (no FiLM)	45.7±0.7	41.8±0.7	35.7±0.6	43.1±0.7

Table 5: Effect of decision boundary realignment when transferring attacks from the surrogate (Simple CNAPS using Resnset18 A with FiLM) to the target (Simple CNAPS using Resnet18 B no FiLM). The support set is 50% adversarial. Attacks are generated using PGD with $\epsilon = 0.3$, $\gamma = 0.0167$, $L = 100$. In *Blind*, the surrogate has no knowledge of the target's decision boundaries; in *Re-Aligned* the support set presented to the surrogate has been relabeled to match the target's predictions. This corresponds to Fig. 3(a).

		Clean	ASP	Swap
Surrogate (Resnet18 A with FiLM)	Re-Aligned	52.6±2.0	4.0±0.4	16.5±1.9
	Blind	54.5±1.8	3.6±0.3	6.6±0.6
Transfer to Resnet18 B (no FiLM)	Re-Aligned	43.9±2.1	19.2±1.2	21.4±0.9
	Blind	40.1±1.9	28.9±2.0	29.9±1.6

Table 6: Effect of varying the clean images in the support set when transferring attacks from the surrogate (Simple CNAPS using Resnset18 A with FiLM) to the target (Simple CNAPS using Resnet18 B no FiLM). Attacks are generated using PGD with $\epsilon = 0.3$, $\gamma = 0.0167$, $L = 100$. For Setting A, 50% of the support set is adversarial and the ratio of clean to poisoned patterns is 0.5:1. For Setting B, 20% of the support set is poisoned and the ratio of clean to poisoned patterns is 1.5:1. This table corresponds to Fig. 3(b).

		Clean	ASP	Swap
Surrogate (Resnet18 A with FiLM)	Setting A	55.7±2.2	6.3±0.6	11.5±1.3
	Setting B	68.8±0.4	32.4±0.4	44.6±0.7
Transfer to Resnet18 B (no FiLM)	Setting A	41.3±2.1	30.0±2.1	31.0±1.5
	Setting B	55.1±0.4	51.7±0.5	48.4±0.3

Table 7: Effect of various dropout strategies when transferring attacks from the surrogate (Simple CNAPS using Resnset18 A with FiLM) to the target (Simple CNAPS using Resnet18 B no FiLM). The support set is 50% adversarial. Attacks are generated using PGD with $\epsilon = 0.3$, $\gamma = 0.0167$, $L = 100$. This table corresponds to Fig. 3(c).

	Surrogate (Resnet18 A with FiLM)			Transfer to Resnet18 B (no FiLM)		
	Clean	ASP	Swap	Clean	ASP	Swap
No dropout	55.5±1.4	4.2±0.3	10.9±1.1	40.1±1.9	28.9±2.0	29.9±1.6
Block-Wise (1%)	44.6±1.5	5.4±0.3	9.8±0.7	42.3±1.8	29.7±1.8	31.5±1.3
Layer-Wise (2%)	44.5±2.1	5.0±0.5	8.6±1.0	38.3±1.9	26.5±1.8	30.5±1.5
Gaussian (0.5%)	47.4±1.6	4.9±0.4	8.2±1.0	39.0±1.5	27.1±1.4	29.9±1.2

A.6 Query Attacks

We present our algorithm for performing query attacks with PGD in Algorithm 2.

Algorithm 2 PGD for Query Attack

Require:
I_{min}: Minimum image intensity
I_{max}: Maximum image intensity
L: Number of iterations
ϵ: Perturbation amount
γ: Step size
$D_S \equiv \{\boldsymbol{x}, \boldsymbol{y}\}$
$D_Q \equiv \{\boldsymbol{x}^*, \boldsymbol{y}^*\}$

procedure PGDQ(D_S, D_Q, f, g)
  $\boldsymbol{\delta} \sim U(-\epsilon, \epsilon)$
  $\tilde{\boldsymbol{x}}^* \leftarrow \text{clip}(\boldsymbol{x}^* + \boldsymbol{\delta}, I_{min}, I_{max})$
  for $n \in 1, ..., L$ **do**
    $\boldsymbol{\delta} \leftarrow \text{sgn}(\nabla_{\tilde{\boldsymbol{x}}^*} \mathcal{L}(f(\tilde{x}^*, g(x, y)), y^*)$
    $\tilde{\boldsymbol{x}}^* \leftarrow \text{clip}(\tilde{\boldsymbol{x}}^* + \gamma\boldsymbol{\delta}, I_{min}, I_{max})$
    $\tilde{\boldsymbol{x}}^* \leftarrow x + \text{clip}(\tilde{\boldsymbol{x}}^* - \boldsymbol{x}^*, -\epsilon, \epsilon)$
  end for
  return $\tilde{\boldsymbol{x}}^*$
end procedure=0