CAN WE ACHIEVE ROBUSTNESS FROM DATA ALONE?

Anonymous authors
Paper under double-blind review

ABSTRACT

In robust machine learning, there is a widespread belief that samples can be decomposed into robust features (parts of the data that withstand small perturbations) and non-robust ones, and it is the role of the robust algorithm (i.e. adversarial training) to amplify the former and erase the latter. In this work, we challenge this view and try to position adversarial robustness as a more model-dependent property: many approaches that assume this simplistic distinction in the features, optimizing the data directly, only give rise to *superficial* adversarial robustness. We revisit prior approaches in the literature that were believed to be robust, and proceed to devise a principled meta-learning algorithm, that optimizes the dataset for robustness. Our method can be thought as a non-parametric version of adversarial training, and it is of independent interest and potentially wider applicability. Specifically, we cast the bi-level optimization as a min-max procedure on kernel regression, with a class of kernels that describe infinitely wide neural nets (Neural Tangent Kernels). Through extensive experiments we analyse the properties of the models trained on the optimized datasets and identify their shortcomings - all of them come in a similar flavor.

1 INTRODUCTION

Since the discovery of adversarial examples (Szegedy et al., 2014; Goodfellow et al., 2015; Papernot et al., 2017; Carlini & Wagner, 2017), the pursuit for adversarially robust machine learning models has been very fruitful in terms of new ideas and concepts that are of interest for the whole machine learning community. Such ideas include for example a new training paradigm, *adversarial training* (Kurakin et al., 2017; Madry et al., 2018), which adapts the learning framework for defending against malicious perturbations. In principle, if $\mathcal{P}$ denotes a data distribution and Δ is a set of allowed perturbations of the input space, we would like to solve the following problem (Madry et al., 2018)

$$\inf_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{P}} \sup_{\delta\in\Delta} \mathcal{L}(f(x+\delta;\theta,\mathcal{D}_{\text{train}}),y), \tag{1}$$

where f is a model parameterized by θ (e.g. a neural network), $\mathcal{D}_{\text{train}}$ denotes a finite dataset used for training, and $\mathcal{L}$ is a loss function used for classification.

Another particularly intriguing concept that has been proposed for understanding the success of adversarial attacks is the presence of *non-robust* features in the data, patterns that seem incomprehensible to humans, yet are useful for classification (Schmidt et al., 2018; Tsipras et al., 2019). Ilyas et al. (2019) captured this notion in a theoretical framework and showed both theoretically and empirically that such patterns can produce vulnerable classifiers. In a dual sense, that work argued that robust features alone are *sufficient* for robust classification; evidence was provided by training a neural network with gradient descent on a modified dataset that presumably contained only such robust features (based on the activations of a previously adversarially trained net), and then observing that this model had non-trivial robustness against gradient based attacks (on the original, unmodified dataset). This seminal work influenced many derivative works that either tried to devise new algorithms or tried to understand the brittleness of models (see e.g. Allen-Zhu & Li (2022); Tsilivis & Kempe (2022)). However, to the best of our knowledge, this data based approach to adversarial robustness has been relatively unexplored thus far.

In this work we challenge the current viewpoint on the role of robust features in the data itself through several angles. First, we scrutinize the “robust” dataset of Ilyas et al. (2019) to show that its robustness is fallacious: it does not withstand adaptive adversarial attacks that are not gradient based.

Even though the dataset was produced from an adversarially trained model that is truly robust (Croce & Hein, 2020), this robustness collapses when the training algorithm does not follow the paradigm of Eq. (1). We also analyze the dataset that is produced on-the-fly during adversarial training to show that it does not, by itself, exhibit robust features of any generality. That is, once we train a model (with different initial seed) with gradient descent on the worst case perturbations that are generated through adversarial training, we fail to achieve adversarial robustness of any form.

To provide a more principled foundation, we develop a non-parametric approach to produce robust datasets by directly optimizing for the robustness objective. This method uses kernel ridge regression with the Neural Tangent Kernel (NTK) as a surrogate to solve an otherwise intractable optimization. It is inspired by recent advances in dataset distillation using NTKs, the kernel-inducing points (KIP) algorithm (Nguyen et al. (2021a;b)). In principle, this approach also makes the underlying assumption that the dataset itself contains all the information needed to yield robust classifiers. Our new method, *advKIP*, produces datasets that on common architectures seem to be even more robust than the previous two approaches that we described. Yet, we find that they also fail to defend against adaptive attacks, further challenging the idea of robust features in common computer vision tasks. Our algorithm is of independent interest, since (i) it shows that certain robust properties transfer from kernel regression with kernels that correspond to infinite neural networks to actual, finite width neural nets, and, (ii) it can serve as a blueprint for any bi-level maximization problem that would be intractable for neural nets and might hence be of interest to other such problems in meta-learning and beyond.

Lastly, we zoom in on all these methods and analyze the properties of the neural nets that were trained with these optimized datasets. We find, perhaps surprisingly, common “signatures of failure” for all of them: shattered gradients give a false sense of robustness, evidencing the well-documented phenomenon of “obfuscated” gradients. What is surprising is the fact that this “obfuscation” does not come from non-differentiable parts of the architecture or addition of stochasticity in the evaluation pipeline, rather it is a property of the data *alone*. We complement our analysis by showing the overconfidence and overcalibration of the models, and contrast them with truly robust networks. We believe that these findings will help designing and faster debugging of data-based approaches in the future, and allow to better understand properties of truly robust models. To summarize:

1. We systematically explore the idea that robust features in the data underlie robust models. We revisit the work of Ilyas et al. (2019) and show that an optimized dataset which presumably contains only robust features yields models that are only superficially robust (fail to withstand adaptive attacks (Croce & Hein, 2020)). We also challenge the data-based viewpoint of robustness by collecting data generated during adversarial training and showing that it fails to yield robust models when deployed on an independent model.
2. We devise a principled meta-learning algorithm, *adv-KIP*, to optimize datasets for the robust loss. Our approach is rooted in kernel regression with a particular class of kernels called Neural Tangent Kernels (NTKs) (Jacot et al., 2018), which are known to describe infinitely wide neural networks. We discover a range of surprising transfer properties from kernels to common networks and point out how the underlying algorithmic framework, which is of independent interest, can be adapted to a range of two-loop (min-max) opimization tasks (Section 4). We find that kernels and neural networks share some common robustness properties, but demonstrate that adv-KIP also fails to produce truly robust models.
3. Finally, we analyze common architectures that are trained with all these datasets and find surprising similarities and common risks. In particular, we show that the models suffer from the “obfuscated gradient” phenomenon (Athalye et al., 2018), and discuss several interesting properties of the model, in terms of overconfidence and poor calibration.

2 Preliminaries

Adversarial Training. Eq. 1 establishes the min-max underpinning for the construction of adversarially robust classifiers (Madry et al., 2018). The most common way to approximate the solution of this optimization problem for a neural network f on a data point $(\mathbf{x}, y)$ is to first generate adversarial examples by running multiple steps of projected gradient descent (PGD) (Kurakin et al., 2017; Madry et al., 2018). When the set of allowed perturbations Δ is $\mathcal{B}_{\mathbf{x}}^{\epsilon}$ - the ℓ_∞ ball of radius ϵ and center $\mathbf{x}$ -

the iterative N-step approximation is given by

$$\mathbf{x}^{k+1} = \Pi_{\mathcal{B}^{\epsilon}_{\mathbf{x}^0}} \left(\mathbf{x}^k + \alpha \cdot \mathrm{sign}(\nabla_{\mathbf{x}^k} \mathcal{L}(f(\mathbf{x}^k), y))\right), \tag{2}$$

where $\mathbf{x}^0 = \mathbf{x}$ is the original example, α is a learning step, $\tilde{\mathbf{x}} = \mathbf{x}^N$ is the final adversarial example, and Π is the projection on the valid constraint set of the data. During adversarial training we alternate steps of generating adversarial examples (using f from the current network) and training on this data instead of the original one. Several variations of this approach have been proposed in the literature (e.g. Zhang et al. (2019); Shafahi et al. (2019); Wong et al. (2020)), modifying either the attack used for data generation (inner loop in Eq. (1)) or the loss in the outer loop.

Kernel Regression and NTK. Kernel regression is a fundamental non-linear regression method. Given a dataset $(\mathcal{X}, \mathcal{Y})$, where $\mathcal{X} \in \mathbb{R}^{n \times d}$ and $\mathcal{Y} \in \mathbb{R}^{n \times k}$ (e.g., a set of one-hot vectors), kernel regression computes an estimate

$$\hat{f}(\mathbf{x}) = K(\mathbf{x}, \mathcal{X})^\top K(\mathcal{X}, \mathcal{X})^{-1} \mathcal{Y}, \tag{3}$$

where $K(x, \mathcal{X}) = [k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_n)]^\top \in \mathbb{R}^n$, $K(\mathcal{X}, \mathcal{X})_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ and k is a kernel function that measures similarity between points in $\mathbb{R}^d$.

Recent work in deep learning theory has established a profound connection between kernel regression and the infinite width, low learning rate limit of deep neural networks (Jacot et al., 2018; Lee et al., 2019; Arora et al., 2019). In particular, it can be shown that the evolution of such suitably initialized infinitely wide neural networks admits a closed form solution as in Eq. (3), with a network-dependent kernel function k. Focusing on a scalar neural net for ease of notation, it is given by:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \nabla_\theta f(\mathbf{x}_i; \theta)^\top \nabla_\theta f(\mathbf{x}_j; \theta), \tag{4}$$

where θ are the parameters of the network. This expression becomes constant (in time) in the infinite width limit.

As outlined in the introduction there are many fruitful applications of the NTK framework, some of which have benefited from transfer properties to common neural nets. Our work builds on a recent data distillation algorithm called *Kernel Inducing Points* (KIP) (Nguyen et al., 2021a;b). These works introduce a meta-learning algorithm for data distillation from an original training set $\mathcal{D}$, to an optimized *source* set $(\mathcal{X}_S, \mathcal{Y}_S)$ of *reduced* size but similar output on a test set. The closed form of Eq. (3) allows to express this objective via a loss function on a *target* data set $(\mathcal{X}_T, \mathcal{Y}_T)$ as:

$$\mathcal{L}_{\mathrm{KIP}}(\mathcal{X}_S, \mathcal{Y}_S) = \|\mathcal{Y}_T - K(\mathcal{X}_T, \mathcal{X}_S)^\top K(\mathcal{X}_S, \mathcal{X}_S)^{-1} \mathcal{Y}_S\|_2. \tag{5}$$

The error of Eq. (5) can be minimized via gradient descent on $\mathcal{X}_S$ (and optionally $\mathcal{Y}_S$). Starting with a smaller subset of $\mathcal{D}$, sampling a target dataset from $\mathcal{D}$ to simulate test points, and backpropagating the gradients of the error with respect to the data allows to progressively find better and better synthetic data. Importantly, leveraging the NTK for kernel regression renders the datasets suitable for deployment on actual neural nets as well.

Prior work on dataset optimization. To the best of our knowledge, the idea of trying to obtain robust classifiers through data or representation optimization is rather unexplored. Garg et al. (2018) design a spectral method to extract robust embeddings from a dataset. (Awasthi et al., 2021) formulate an adversarially robust formulation of PCA, to extract provably robust representations. (Ilyas et al., 2019) constructs a robust dataset by traversing the representation layer of a previously trained robust classifier and serves as an inspiration for this work. Yet, all of these methods achieve substantially lower robust accuracy compared to adversarial training.

3 Previous Data Based Approaches to Robustness

Next, we describe prior (this section) and new (Sec. 4) methods that we use for synthesizing datasets that are presumably free from non-robust parts, and hence should be able to yield robust machine models from standard training alone.

3.1 Removal of non-robust features based on adversarial training

To illustrate the theory of robust and non-robust features in the data, Ilyas et al. (2019) have introduced a “robustified” data set (“Robust Feature Dataset” RFD), the only dataset we are aware of that is believed to provide models with some notable robustness via standard training. RFD is generated by traversing the representation layer of an adversarially trained neural network, and is thus believed to provide a general sense of robustness (Ilyas et al., 2019). More specifically, given a mapping $x \mapsto g(x)$ of an input x to the penultimate ("representation") layer of an adversarially trained neural net, a "robustified" input is obtained by optimizing $\min_{x_r} \|g(x) - g(x_r)\|_2$, starting from a random data point using gradient descent, thus enforcing that the robust representations of x and x_r are similar while x_r does not contain non-robust features given a starting point that is uncorrelated with the label of x.

3.2 Worst-case augmentation from adversarial training

Perhaps the most obvious dataset that could potentially have the property to induce robustness in models via standard training alone is the "worst-case" augmented dataset that is produced on the fly during adversarial training. Let f_t denote a neural network after t steps of parameter update during adversarial training, and $x_i^{(t)} = \arg\max_{\delta \in \Delta} \mathcal{L}(f_t(x_i + \delta), y)$ be the worst case version of each of the training samples x_i for f_t. Let $X_t = \bigcup_{i \in [n]} x_i^{(t)}$ denote the collection of all adversarial examples at time t (where n is the number of the training points). Then, we collect $\hat{\mathcal{X}} := \bigcup_{t \in T} X_t$, where T is the number of epochs needed for convergence of the algorithm. We can analyse robustness properties of $\hat{X}$ by giving it as an input to a model (possibly of the same architecture, but with a different seed) to be trained with standard gradient descent.

4 Our Adv-KIP Algorithm

A centerpiece of our work is a new framework to approach min-max optimization tasks as bi-level *optimization on the training data* and its instantiation, *advKIP*, to optimize for adversarial robustness. Adversarial training gives an approximation to the solution of Eq. (4) by iterating gradient steps on model parameters and on data. However, once we focus on non-parametric models f (such as kernel ridge regression), we can pose a more "direct" problem

$$\inf_{\mathcal{D}_{\text{train}}} \mathbb{E}_{(x,y)\sim\mathcal{P}} \sup_{\delta \in \Delta} \mathcal{L}(f(x+\delta; \mathcal{D}_{\text{train}}), y), \tag{6}$$

where instead of optimizing the model parameters, we optimize the *training data*. The above formulation has the benefit of directly optimizing the quantity of interest, that is the robust loss at the end of "training"/deployment. Additionally, since the outcome of this optimization is a dataset, it can be deployed with any other model, and, given favorable transfer properties, might yield good performance even outside the scope it was optimized for, without the need for costly adversarial training on the new model. This latter hope is not unfounded, since adversarial examples themselves have been shown to be rather universal and transferable across models (Papernot et al., 2017; Moosavi-Dezfooli et al., 2017).

We propose a gradient-based approach for solving the optimization in Eq. (6), focusing on kernel regression; specifically with a particular class of kernel functions, *Neural Tangent Kernels* (NTKs). Kernel regression with NTKs is known to describe the training process of infinitely wide, suitably initialized networks (Jacot et al., 2018; Lee et al., 2019; Arora et al., 2019), yet in some cases has shown considerable transfer properties to commonly used neural nets.

Dataset Distillation. The inspiration for our advKIP framework comes from a number of recent works on *Dataset Distillation* (Wang et al., 2018): the procedure of distilling knowledge from a large dataset to a smaller one. Following Nguyen et al. (2021a;b), our method works with kernel machines, and especially with NTKs. However, the goal here is slightly different; instead of deriving a dataset of reduced size, we aim to create one that induces better robustness properties *on the original unmodified test set*. For further motivation and comparison to previous work, see App. A.1.

We depart from the KIP setting to introduce our framework for dataset optimization for robust classification. Our method is a natural extension of the KIP algorithm outlined in the previous section, but suitably adjusted for adversarially robust classification.

In particular, instead of optimizing the data $(\mathcal{X}_S, \mathcal{Y}_S)$ with respect to the "clean" loss of Eq. (5), we minimize

$$\mathcal{L}_{\text{advKIP}}(\mathcal{X}_S, \mathcal{Y}_S) = \|\mathcal{Y}_T - K(\tilde{\mathcal{X}}_T, \mathcal{X}_S)^\top K(\mathcal{X}_S, \mathcal{X}_S)^{-1} \mathcal{Y}_S\|_2, \tag{7}$$

where, in a slight abuse of notation, $\tilde{\mathcal{X}}_T = \mathcal{X}_T + \tilde{\delta}$, and

$$\tilde{\delta} = \arg\max_{\delta \in \Delta} \mathcal{L}(K(\tilde{\mathcal{X}}_T, \mathcal{X}_S)^\top K(\mathcal{X}_S, \mathcal{X}_S)^{-1} \mathcal{Y}_S, \mathcal{Y}_T). \tag{8}$$

In what follows we will take the loss $\mathcal{L}$ in Eq. (8) to be the cross-entropy loss $\mathcal{L}_{\mathrm{ce}}$ as is very common in adversarial training, but note that we have the freedom to choose any loss function, for instance losses used for alternative attacks like the CW-loss (Carlini & Wagner, 2017). The loss in Eq. (7) can also be adapted to other losses that balance clean and robust accuracy (see e.g. Zhang et al. (2019)).

We observe that this approach follows what we advertised in Eq. (6). It adds an inner maximization to the KIP framework. Solving this optimization now requires an inner loop that tackles the maximization in Eq. (8). Here, we choose to apply a similar iterative procedure as in the PGD approach of Eq. (2). For the remainder of the paper, we restrict ourselves to the case of an ℓ_∞ adversary. However, note that our method is easily extendable to any constraint set Δ.

Algorithm 1: Adversarial KIP

```
Input: A training dataset $\mathcal{D}_{\mathrm{train}} = \{\mathcal{X}, \mathcal{Y}\}$.
Output: A new dataset $\mathcal{D}_{\mathrm{rob}}$.
1 Sample data $\mathcal{S} = \{\mathcal{X}_S, \mathcal{Y}_S\}$ from $\mathcal{D}_{\mathrm{train}}$;
2 for i ← 1 to epochs do
3     Sample data $\mathcal{T} = \{\mathcal{X}_T, \mathcal{Y}_T\}$ from $\mathcal{D}_{\mathrm{train}}$;
4     for j ← 1 to pgd_steps do
5         $\mathcal{X}_T \leftarrow \mathcal{X}_T + \alpha \cdot$
            $\mathrm{sign}(\nabla_{\mathcal{X}_T} \mathcal{L}_{\mathrm{ce}}(K_{\mathcal{X}_T \mathcal{X}_S} K^{-1}_{\mathcal{X}_S \mathcal{X}_S} \mathcal{Y}_S, \mathcal{Y}_T))$;
6         $\mathcal{X}_T \leftarrow \Pi_{\mathcal{B}_\epsilon}(\mathcal{X}_T)$;
7     $\mathcal{X}_S \leftarrow \mathcal{X}_S - \lambda \nabla_{\mathcal{X}_S} \mathcal{L}(K_{\mathcal{X}_T \mathcal{X}_S} K^{-1}_{\mathcal{X}_S \mathcal{X}_S} \mathcal{Y}_S, \mathcal{Y}_T)$;
8     $\mathcal{Y}_S \leftarrow \mathcal{Y}_S - \lambda \nabla_{\mathcal{Y}_S} \mathcal{L}(K_{\mathcal{X}_T \mathcal{X}_S} K^{-1}_{\mathcal{X}_S \mathcal{X}_S} \mathcal{Y}_S, \mathcal{Y}_T)$;
9 $\mathcal{D}_{\mathrm{rob}} \leftarrow (\mathcal{X}_S, \mathcal{Y}_S)$
```

Algorithm choices: Algorithm 1 describes our generic robust training data set distillation framework. There are several options to specialize:
Outer loss function (lines 7 and 8): We have considered both Mean Squared Error (mse) (as in Eq. (7)) and Cross Entropy loss (ce). Experiments on MNIST suggest *ce* as the marginally better choice to achieve PGD-robustness.
Optimization of labels: We have considered Algorithm 1 both as is (learned labels) and without line 8 (fixed labels). We find little difference and opted to include label learning.

$|\mathcal{X}_S|$ and $|\mathcal{X}_T|$: We observe in all our experiments that the larger the source (training) data set $\mathcal{X}_S$, the better performance, though larger sets incur higher computational cost. Sensitivity to test set size $|\mathcal{X}_T|$ is much less pronounced.

Number of PGD-steps: We have run the algorithm with either 1,6 or 10 PGD-steps. 1 PGD-step corresponds to the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015). In our experiments we observe that single-step attacks (FGSM) are strictly weaker than iterative ones, and training against a higher number of PGD-steps provides defense against FGSM attacks for kernels, as is the case for adversarial training.

Algorithmic Framework for Min-Max Problems: Algorithm 1 is an approach to a particular min-max problem given in Eq. (6). The advantage of deploying the NTK here is that it affords an analytic surrogate expression for the output of the trained model, which allows to compute gradients with respect to the input dataset. As we show in Sec. 5.3, the resulting datasets enjoy surprising transfer properties to common neural nets, even in the case where the kernel has an entirely different architecture. For other bi-level problems we can imagine modifying the inner and outer loop objective function of Algorithm 1 to provide dataset based solutions to other min-max optimizations, in particular in meta-learning. For example, the problem of few-shot learning might be cast in this framework, where the inner loop would optimize for accuracy on a small out-of-distribution target set. We thus hope that our approach, bolstered by its supporting transfer results might be useful in other settings.

5 Experimental Evaluation of Data Based Approaches

Here we evaluate the approaches described or introduced in Sec. 3. We perform experiments on MNIST (Deng, 2012) and CIFAR-10 (Krizhevsky, 2009) against ℓ_∞ adversaries of size ϵ equal to 0.3 and 8/255, respectively, or ℓ_2-adversaries of size 128/255 for CIFAR-10. In all cases, robust performance is measured on adversarially perturbed original (test) images. For MNIST, we train a simple three-layer CNN of width 64 with Max-Pooling. For CIFAR-10, we train this three-layer CNN, as well as two more complex architectures, AlexNet (Krizhevsky et al., 2012) and VGG11 (Simonyan & Zisserman, 2015). We also analyze transfer results on a set of wide, fully-connected networks (see App. A.6). Further experimental details can be found in App. A.2.

To evaluate adversarial robustness, we compute the earlier gradient-based FGSM (Goodfellow et al., 2014) and PGD (Madry et al., 2018; Kurakin et al., 2017) metrics, but also evaluate robustness against the adaptive suite of the AutoAttack benchmark (Croce & Hein, 2020), which contains attacks that do not use gradient information.

5.1 Robustness of RFD

We start by revisiting the "robust-feature" dataset of Ilyas et al. (2019) which is presumed to only contain robust features and hence to provide a general sense of robustness, as described in Sec. 3.1. To replicate these results on our architectures, we use an ℓ_∞-adversarially trained ResNet50 to generate a variant of this dataset on CIFAR-10[1]. Table 1 shows performance of our models.

CIFAR-10 Accuracy with ℓ_∞ Robust Feature dataset (Ilyas et al., 2019)			
Neural Net	Clean	PGD ℓ_∞ 20	AA ℓ_∞
Simple CNN	59.15 ± 0.37	52.91 ± 0.66	0.00 ± 0.00
AlexNet	51.62 ± 1.14	25.64 ± 4.32	0.02 ± 0.03
VGG11	61.59 ± 0.80	34.64 ± 8.47	0.40 ± 0.42

Table 1: Test accuracies for various models trained on a 50K ℓ_∞ "robust feature" dataset (RFD) for CIFAR-10.

Indeed, the trained models record high robustness against PGD attacks, confirming the findings of Ilyas et al. (2019) (see Fig. 2 for the evolution of test accuracies during training). However, we observe almost 0% accuracy against the adaptive suite of AutoAttack. This is a surprising finding, since the dataset was generated using adversarially trained networks that guarantee a wide sense of robustness. It is a first indication that achieving *true* robustness from data alone might be a challenging task when decoupled from the training algorithm.

5.2 Using data generated from Adversarial Training

Following Sec. 3.2 we have also tested our convolutional architectures on CIFAR-10 on the dataset obtained from epoch-wise worst-case augmentations of the training data during adversarial training. We evaluate on same model, changing only its initialization. Table 6 in App. A.4 shows test results for our convolutional architectures. Perhaps surprisingly, none of these models retain any robustness, not even against FGSM and PGD attacks. We conclude that data alone here is not sufficient: the model dependency of the perturbed inputs cannot be eliminated.

5.3 Robustness from adv-KIP

We now proceed to implement our advKIP algorithm and evaluate the robustness of the distilled dataset on kernels and neural nets.

AdvKIP Setup: To apply Adversarial KIP for learning robust datasets we consider several different fully connected (FC) and convolutional (Conv) kernels, whose expressions are available through the Neural Tangents library (Novak et al., 2020), built on top of JAX (Bradbury et al., 2018). In particular, for MNIST we implement fully connected kernels[2] of depth $3, 5$ and 7 ($FC3, FC5, FC7$) and a 3-layer convolutional kernel (Conv3); and for CIFAR-10 fully connected kernels of depth 2 and 3 ($FC2, FC3$). For FC kernels data set sizes are $|\mathcal{X}_S| = 30K$ (MNIST) and $40K$ (CIFAR10) with $|\mathcal{X}_T| = 10K$. For CIFAR-10 and the $FC3$ kernel we also generate a dataset with $|X_S| = 50K$ to be deployed on the convolutional neural nets. For $Conv3$, computational resources have restricted us to runs with data sets of size $|\mathcal{X}_S| = 5K$ and $|\mathcal{X}_T| = 1K$. We set the number of PGD-steps in the inner loop (line 4) to 10 for MNIST and 6 for CIFAR10. We implement early stopping if robust validation accuracy ceases to increase.

[1] Note that the *publicly* available dataset of Ilyas et al. (2019) is derived from an adversarially trained network trained against an ℓ_2 adversary, so for completeness we include an ℓ_2 evaluation of that dataset in App. A.3; it also does not give any robustness against AA attacks.

[2] We are currently unable to produce large datasets with Convolutional Kernels. We leave the exploration and potential performance improvement to future work.

Robustness on Kernels: In a first set of experiments with MNIST, we verify on the validation set that Algorithm 1 succeeds to converge. Fig. 1 shows accuracy throughout training for 3-layer kernels. We see how robust validation accuracy against FGSM and PGD attacks increases with the number of outer loop steps, essentially without compromising performance on clean data. Note that at the start of optimization the robustness of the dataset is effectively 0%, as expected from studies on neural nets. We also note that the convolutional architecture achieves better performance, despite the fact that we can only deploy it with a smaller dataset $\mathcal{X}_S$ of size $5K$. This indicates that convolutional architectures might be more optimizable than their fully connected counterparts.

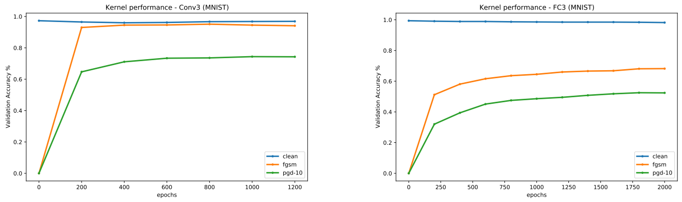

Figure 1: MNIST: Training curves on kernels. Shows validation performance as a function of training epochs. Left: CONV3, $|\mathcal{X}_S| = 5K$, $|\mathcal{X}_T| = 1K$ Right: FC3, $|\mathcal{X}_S| = 30K$, $|\mathcal{X}_T| = 10K$.

Table 2 tabulates clean and robust *test* accuracy after the algorithm has converged or stopped on the validation set. Here, we record accuracy for our deepest models, $Conv3$ and $FC7$ on MNIST. We see that robust performance generalizes well to the test set (where we have tested against pgd-attacks with the same number of steps as optimized for in line 4 of Algorithm 1). Producing these robust classifiers with kernel regression is an encouraging step, in particular since kernel machines are not directly amenable to adversarial training, and thus robustness to PGD attacks has not been observed before.

			Robust	
Dataset	**Kernel, Dataset Size**	**Clean**	FGSM	PGD
MNIST	Conv3, 5k	96.31	94.82	76.62
MNIST	FC7, 30k	97.21	67.04	50.34
CIFAR-10	FC2, 40k	59.65	20.49	20.37
CIFAR-10	FC3, 40k	59.95	21.67	21.56

Table 2: Kernel results on MNIST and CIFAR-10: Clean and Robust test accuracy ($\%$). For MNIST, we test with PGD-10, and for CIFAR, we test with PGD-6.

For the CIFAR-10 results, we see a marked drop in both clean and robust accuracy; but note that generally fully connected architectures are not very suitable classifiers for the CIFAR-10 dataset. To achieve *some* level of robustness with these simple architectures gives credence to our approach.

In addition, to verify the necessity of bi-level optimization with the adv-KIP algorithm (as opposed to single-loop KIP-dataset distillation), we check robust accuracy on datasets of the same size as ours, but produced by the original KIP algorithm (Nguyen et al., 2021a;b) which is optimized for clean test accuracy only (see Appendix A.5). We find that KIP datasets do not provide any robust accuracy, neither against FGSM nor PGD attacks . This indicates the clear need to adjust the optimization objective to robust performance, as is done in the adv-KIP algorithm.

Transfer of PGD-Robustness to wide neural nets. We perform a first set of transfer studies of robustness from FC kernels to their corresponding *wide* neural nets of the same architecture (Appendix A.6). We find that PGD-robustness transfers well to these large-width counterparts. This is an indication that features learned by kernels are similar to those learned by neural nets.

Robustness for Common Architectures. We now turn our attention to commonly employed convolutional neural networks to study the relevance of our datasets for robust classification using modern architectures. In particular we report how datasets generated by FC kernels transfer to such convolutional architectures, since the datasets we can currently distill with CONV kernels are too small.

Tables 3 and 4 summarize our findings for MNIST and CIFAR-10. We note an astonishing "boost" in robust test accuracy against gradient-based attacks on these convolutional networks when compared to the fully connected kernel results in Table 2 and results for wide FC networks in Table 8. Very remarkably, it seems that datasets optimized for relatively simple kernels "transfer" their pgd-performance to networks far removed from this "idealistic" regime, even to more expressive architectures.

		Robust		
Train method	**Clean**	FGSM	PGD40	AA
FC3	98.15 ± 0.12	98.06 ± 0.18	97.17 ± 0.10	0.00 ± 0.00
FC5	97.96 ± 0.55	97.87 ± 0.64	97.20 ± 0.74	0.00 ± 0.00
FC7	98.03 ± 0.16	97.91 ± 0.22	97.14 ± 0.43	0.00 ± 0.00
Adversarial Training	99.11	97.52	95.82	88.77

Table 3: MNIST with Simple-CNN: Test accuracies when trained on Adv-KIP datasets optimized with FC kernels (first 3 rows). We also show test accuracies for the adversarially trained simple-CNN (without any data augmentation). AA refers to the *AutoAttack* test suite with ℓ_∞.

	CIFAR-10 Adv-KIP Results				**CIFAR-10 AT Baseline**				
Neural Net	**Clean**	FGSM	PGD20	AA	Clean	FGSM	PGD ℓ_∞ 20	PGD ℓ_2 20	AA
Simple CNN	72.10 ± 0.10	67.45 ± 0.37	67.03 ± 0.24	0.00 ± 0.00	58.07	33.94	31.49	43.89	26.18
AlexNet	68.87 ± 0.76	49.30 ± 0.69	49.06 ± 0.63	0.89 ± 1.41	44.35	30.12	24.41	16.68	18.95
VGG11	74.88 ± 0.45	53.98 ± 9.71	53.18 ± 10.32	0.27 ± 0.18	69.65	31.30	24.68	46.67	23.85

Table 4: CIFAR-10: Test accuracies of several convolutional architectures trained on our Adv-KIP dataset from the FC3 kernel (left), and from Adversarial Training (right).

Fig. 2 shows the evolution of test PGD-accuracies during training. We point out that while clean accuracy increases rapidly, robust accuracy only starts to increase once clean accuracy is essentially optimized. We hypothesize that this might be due to the fact that our distillation optimizes using the expression of the kernel at the *end* of training. A similar behaviour can be observed for training on RFD, perhaps for similar reasons: the data synthesis utilizes an adversarially trained neural net at *convergence*.

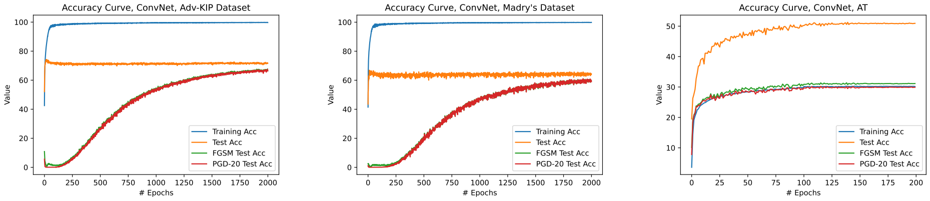

Figure 2: Evolution of accuracies during training with our 50K Adv-KIP dataset, the RFD dataset of App. A.3, and normal Adversarial Training for CIFAR-10 on ConvNet.

It seems promising at first that modern networks trained with adv-KIP datasets without much tuning enjoy astonishing defense properties against PGD-attacks in various settings, similar, or in some cases even higher, than what truly robust models (i.e adversarially trained) obtain. However, when we deploy the AutoAttack suite to our convolutional nets, we observe a sharp drop in robust test accuracy close to zero, as can be seen in the "AA"-columns of Tables 3 and 4. We hence observe a similar failure to produce *true* robustness as in the case of the RFD-dataset (Sec. 5.1), despite an arguably more principled approach to optimize for robustness.

6 Analysis of dataset-based approaches and discussion

In this section, we analyze the models trained with optimized datasets that yield superficial robustness. We demonstrate that, surprisingly, they share many common properties and signatures of failure which can be identified during training, and contrast them to adversarially trained neural networks. These similarities might be germane to all attempts that explicitly optimize the dataset with gradient based approaches. Figures can be found in Apps. A.7 and A.8.

First, in Fig. 5 notice how test loss behaves differently in correctly classified vs wrongly classified samples during training on the synthetic datasets. We see that after a few epochs the model "sacrifices" performance on the subset of data that is misclassified, in order to fit the rest. This is in sharp contrast with the situation during adversarial training (Fig. 5, right). This is already an indication of failure of learning. Remarkably, Fig. 7 shows that this failure of learning is also evident in the gradient norms of the loss with respect to the *input*. The average gradient norm on the wrongly classified points explodes with the number of epochs for both the advKIP dataset and the RFD. This behavior, together with the false sense of robustness that AutoAttack evaluation reveals, suggests that the model learns to shatter the gradients locally in the neighborhood of correctly classified examples, causing simple gradient-based attacks to fail. This is evidently similar to what is commonly termed the *obfuscated gradient* phenomenon (Athalye et al., 2018), a situation where model gradients do not provide good directions for generating successful adversarial examples. However, in the past, this has only been observed with techniques that were introducing non-differentiable parts in the inference pipeline, or stochasticity, or multiple iterations of neural network evaluation to the model. Interestingly, we now observe this phenomenon from altering the *training* data alone and, even more remarkably, from data optimized using kernels. Indeed, we see in Fig. 9 how the distillation procedure effectively shrinks the gradients of the model. Further analysis shows that models trained on the robustified datasets are geared towards becoming *overconfident*. To illustrate this we provide confidence histograms and reliability diagrams (Guo et al., 2017) to compare standard training, adversarial training, KIP-training (see App. A.5), adv-KIP and RFD for the simple CNN in Figs. 3 and 4. They reveal that the models trained on both our dataset and the RFD dataset are extremely confident on nearly all test examples, whether correctly labeled or not (see App. A.10 for details).

Stepping back, we have established that datasets optimized for robustness provide a false sense of security against gradient-based attacks, but fail to show true robustness. Note that the advKIP Algorithm 1 applies PGD-optimization in the inner maximisation of steps 4, 5 and 6. This is similar to adversarial training where the optimization is only carried out against inner-loop PGD-attacks on the data. We thus remain with a conundrum: while in the case of adversarial training optimizing against PGD-attacks yields the stronger AA-robustness (see Table 4 right), in the case of optimization of datasets with advKIP or with the RFD procedure the same is not true. Giving credence to our hypothesis that these modes of failure seem to be inherent to all gradient-based approaches to data optimization, we try several variations of Algorithm 1 (by either changing the outer or the inner loop of the optimization), and we observe no qualitative difference (see App. A.9)

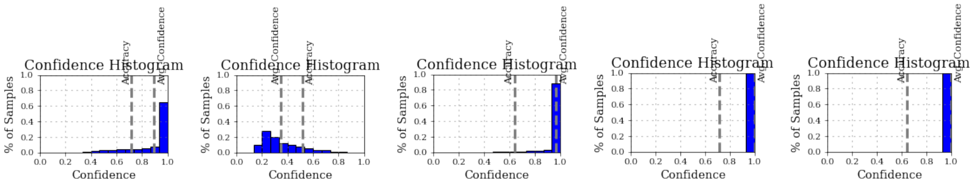

Figure 3: Confidence Histograms of several models. From left to right: standard model; AT model; KIP model; Adv-KIP model and RFD model.

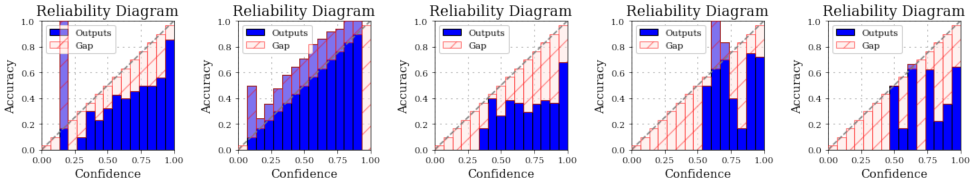

Figure 4: Reliability Diagrams of several models. From left to right: standard model; AT model; KIP model; Adv-KIP model and RFD model.

7 Conclusion

In this work, we reexamine the idea of robust features. Through extensive experiments on common computer vision tasks, we show that robust representations can only induce robustness in models trained with gradient descent on a superficial level. We feel that this clarifies the importance of specialized robust training algorithms, and switches the focus to the model side. Of independent interest, we believe that the adv-KIP algorithm, a principled method for dataset optimization, provides insights into the relationship between kernels and neural nets, and serves as a framework that can be adapted to other data-optimization tasks, as described in Sec. 4.

References

Zeyuan Allen-Zhu and Yuanzhi Li. Feature purification: How adversarial training performs robust deep learning. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 977–988. IEEE, 2022.

Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 8139–8148, 2019.

Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 274–283. PMLR, 2018.

Pranjal Awasthi, Vaggos Chatziafratis, Xue Chen, and Aravindan Vijayaraghavan. Adversarially robust low dimensional representations. In Mikhail Belkin and Samory Kpotufe (eds.), *Conference on Learning Theory, COLT 2021, 15-19 August 2021, Boulder, Colorado, USA*, volume 134 of *Proceedings of Machine Learning Research*, pp. 237–325. PMLR, 2021.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pp. 39–57. IEEE Computer Society, 2017.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

Francesco Croce and Matthias Hein. Mind the box: l_1-apgd for sparse adversarial attacks on image classifiers. In *ICML*, 2021.

Morris H DeGroot and Stephen E Fienberg. The comparison and evaluation of forecasters. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 32(1-2):12–22, 1983.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Shivam Garg, Vatsal Sharan, Brian Hu Zhang, and Gregory Valiant. A spectral view of adversarially robust features. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 10159–10169, 2018.

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

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew M Dai, and Dustin Tran. Training independent subnetworks for robust prediction. *arXiv preprint arXiv:2010.06610*, 2020.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 125–136, 2019.

Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 8580–8589, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*, 2018.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 86–94. IEEE Computer Society, 2017.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021a.

Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. Dataset distillation with infinitely wide convolutional networks. *CoRR*, abs/2107.13034, 2021b.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pp. 625–632, 2005.

Roman Novak, Lechao Xiao, Jiri Hron, Jaehoon Lee, Alexander A. Alemi, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. Neural tangents: Fast and easy infinite neural networks in python. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In Ramesh Karri, Ozgur Sinanoglu, Ahmad-Reza Sadeghi, and Xun Yi (eds.), *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, pp. 506–519. ACM, 2017.

Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 3353–3364, 2019.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Aman Sinha, Hongseok Namkoong, and John C. Duchi. Certifying some distributional robustness with principled adversarial training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

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

Nikolaos Tsilivis and Julia Kempe. What can the neural tangent kernel tell us about adversarial robustness? In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2022.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C. Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 5339–5349, 2018.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A. Efros. Dataset distillation. *CoRR*, abs/1811.10959, 2018.

Florian Wenzel, Jasper Snoek, Dustin Tran, and Rodolphe Jenatton. Hyperparameter ensembles for robustness and uncertainty quantification. *Advances in Neural Information Processing Systems*, 33:6514–6527, 2020.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Chia-Hung Yuan and Shan-Hung Wu. Neural tangent generalization attacks. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12230–12240. PMLR, 18–24 Jul 2021.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7472–7482. PMLR, 2019.

A Appendix

A.1 Related Work

Distributionally Robust Optimization and Adversarial Augmentation. Related to our work are also works on distributionally robust optimization (Sinha et al., 2018) and adversarial data augmentation for out-of-distribution generalization (Volpi et al., 2018). The latter proposes an algorithm that augments the training dataset *on-the-fly* (i.e. during training of a neural net) with worst-case samples from a target distribution. In contrast, our method optimizes the original dataset against worst-case samples/adversarial examples from the original distribution, which correspond to a final predictor (kernel machine). The only prior work that gives an algorithm based on NTK theory to derive dataset perturbations in some adversarial setting is due to Yuan & Wu (2021), yet with entirely different focus. It deals with what is coined *generalization attacks*: the process of altering the training data distribution to prevent models to generalise on clean data. To our knowledge, KIP and this NTGA algorithm are the only examples of leveraging NTKs for dataset optimization.

Adv-KIP shares similarities with all the above areas, but has distinct differences: the goal of our method is to obtain robust classifiers, as in adversarial training, but it does not alter the training algorithm; it generates worst-case samples, but instead of adding them to the training dataset (as adversarial data augmentation techniques do), it uses them to optimize the dataset itself, similar to a dataset distillation procedure but tailored to adversarially robust classification.

A.2 Experimental Details

For all models trained on our Adv-KIP dataset and the RFD dataset (Ilyas et al., 2019), we use the Adam optimizer to perform a small grid search for the learning rate, and pick the best model with respect to the PGD test accuracy.

On MNIST, we train fully connected networks of width 1024 in Sec. A.6, and the simple-CNN network in Sec. 5. FC networks are trained for 2,000 epochs and the simple-CNN network for 800 epochs.

On CIFAR-10, we again train fully connected networks of width 1024 in Sec. A.6, and the simple-CNN, AlexNet and VGG11 networks in Sec. 5. We train all these networks for 2,000 epochs.

For the Adversarial Training baseline, on MNIST, we adopt the setting of Madry et al. (2018), that is we train the simple-CNN network with the Adam optimizer towards convergence, and set the initial learning rate to 1e-4. In Madry et al. (2018) the number of epochs was set to 100, while we use 200.

On CIFAR-10, since we do not use data augmentation, we train with both SGD and Adam for 200 epochs for each model, and pick the better one in terms of robustness. For the simple-CNN and AlexNet, the Adam optimizer is better. For VGG11, we use the SGD optimizer, with initial learning rate 1e-1, decay rate of 10 at the 100-th and the 150-th epoch, and with weight decay 5e-4.

Simple-CNN architecture: We use a simple convolutional architecture with three convolutional layers and a linear layer. Each convolutional layer computes a convolution with a 3×3 kernel, followed by a ReLU and a max-pooling layer (of kernel size 2×2 and stride 2). The linear layer is fully-connected with ten outputs. All convolutional layers have a fixed width of 64.

Training of Convolutional Nets: We use the Adam optimizer (Kingma & Ba, 2015) and perform a small grid search over the fixed learning rate. We stop training when robust validation accuracy ceases to decrease, where we measure against PGD40 attacks for MNIST and PGD20 attacks for CIFAR-10, as is often standard. We report the best results across the sweep for FGSM and PGD test accuracies. After picking the best learning rate, for each experiment in this paper, we report the mean and standard deviation of three experiments with different seeds.

Description of Evaluation Metrics: For all the adversarial attack related measurements including FGSM, ℓ_∞ PGD and ℓ_2 PGD, we adopt the cleverhans code implementation (Papernot et al., 2018). For ℓ_∞ PGD, on MNIST we use step size 0.1 and radius 0.3, while on CIFAR we use step size 2/255 and radius 8/255. For ℓ_2 PGD on CIFAR, we use step size 15/255 and radius 128/255.

For AutoAttack, we adopt the open-source original implementation (Croce & Hein, 2020; 2021). As for the other attacks, we set $\epsilon = 0.3$ for MNIST and 8/255 for CIFAR-10 for ℓ_∞ attacks and adopt the 128/255 radius for CIFAR-10 for the ℓ_2 adversary.

A.3 Robustness of the publicly available RFD

Here, we replicate the results of Sec. 5.1 for the publicly available RFD dataset of Ilyas et al. (2019). Since it stems from a network trained against an ℓ_2 adversary, we have included robustness against ℓ_2-attacks.

CIFAR-10 Accuracy with Robust Feature dataset (Ilyas et al., 2019)

Neural Net	Clean	PGD ℓ_∞ 20	PGD ℓ_2 20	AA ℓ_2
Simple CNN	65.25 ± 0.44	60.73 ± 0.24	63.73 ± 0.40	0.47 ± 0.11
AlexNet	57.07 ± 1.25	25.12 ± 5.46	26.58 ± 4.80	0.62 ± 0.25
VGG11	68.41 ± 1.95	42.92 ± 11.23	47.49 ± 6.12	6.94 ± 2.47

Table 5: Test accuracies for various models trained on the publicly-available 50K "robust feature" dataset (RFD) for CIFAR-10.

A.4 Using data generated from Adversarial Training

To show its complete lack of induced robustness for the worst-case dataset obtained on-the-fly during adversarial training, we conduct the following experiment: first, we adversarially train a network towards general robustness and record all perturbed data and corresponding labels. Then, we reinitialize the same network and retrain it with precisely the same set of perturbed data in identical order. In other words, we use the data trajectory that produces a robust network with the first initialization, and evaluate whether it is still useful for the second.

Table 6 shows the results discussed in Sec. 5.2.

		Robust	
Neural Net	**Clean**	FGSM	PGD ℓ_∞ 20
Simple CNN	70.22 ± 1.13	0.92 ± 0.33	0.00 ± 0.00
AlexNet	76.53	3.15	0.03
VGG11	84.01 ± 0.39	5.00 ± 0.36	0.06 ± 0.00

Table 6: Test accuracies of several convolutional architectures trained on the AT data trajectory with different initializations. The AlexNet result is unstable: two models got stuck at 10% accuracy. We discard those results.

A.5 KIP baseline

The original KIP algorithm (Nguyen et al., 2021a;b) is designed to reduce the size of the training set, while keeping the induced accuracy close to the original one. It could be reasonable to hypothesize that such information compression might possibly lead to an increase of robustness as well. As a sanity check we evaluate robust accuracy on data sets produced by the original KIP algorithm (Nguyen et al., 2021a;b), which is designed for reduction of dataset size, while keeping (clean) accuracy as uncompromised as possible. For fair comparison, we produce a larger data set using the KIP algorithm (of the same size as used in our adv-KIP algorithm) and check for FGSM-robustness. Table 7 shows that effectively PGD-robustness of the datasets remains close to 0, as is the case for the original datasets. This indicates the clear need to adjust the optimization objective to robust performance, as is done in the adv-KIP algorithm.

We also evaluated FC$\{3, 5, 7\}$ and Conv$\{3, 5, 7\}$ kernels (together with a Convolutional Kernel with 1 hidden layer followed by global average-pooling) on datasets (with 50 images per class) released by (Nguyen et al., 2021b) and we found their FGSM robustness to be 0% in all cases. URLs for the datasets we considered: 1st and 2nd.

	Kernel, Dataset Size	Clean	FGSM
MNIST	FC3, 5k	97.51 ± 0.03	0.00 ± 0.00
	FC7, 30k	98.23 ± 0.06	0.00 ± 0.00
CIFAR-10	FC3, 1k	48.45 ± 0.34	2.50 ± 0.21
	FC3, 5k	52.48 ± 0.23	0.22 ± 0.05
	FC3, 10k	54.04 ± 0.41	0.10 ± 0.04

Table 7: KIP baseline datasets (reproduced). Setting: No preprocessing/data augmentation, target size 1k images, learned labels, mse loss, lr=1e-3, datasets were optimized for 1000 epochs, with potential early stopping if validation accuracy did not increase across 200 epochs. Random seed denotes different draws of the initial support images.

A.6 Transfer Results to Wide FC Networks

Here, we evaluate how well datasets produced with kernel methods in Algorithm 1 transfer to relatively wide neural nets of the same architecture and depth as used in the adv-KIP optimization. We implement multilayer fully connected neural nets of width 1024 and perform a hyperparameter search for the (constant) learning rate. We use the Adam optimizer (Kingma & Ba, 2015) and test for both FGSM and PGD accuracy, where we apply the most common PGD attacks (PGD40 for MNIST and PGD20 for CIFAR10). Table 8 summarize our results for MNIST and CIFAR-10.

			Robust	
Dataset	Kernel, Dataset Size	Clean	FGSM	PGD
MNIST	FC3, 30k	80.08	77.67	53.85
MNIST	FC5, 30k	97.75	64.83	35.14
MNIST	FC7, 30k	97.45	70.58	40.70
CIFAR-10	FC2, 40k	46.29	20.98	16.89
CIFAR-10	FC3, 40k	46.33	40.07	39.15

Table 8: Transferability : Kernel to Neural Network of same architecture, test accuracy in %. For MNIST, we test with PGD-40, and for CIFAR, we test with PGD-20.

We find that robustness properties transfer well from kernels to their corresponding neural nets, an encouraging sign. Our sweeps also show that this holds for a rather wide range of learning rates, evidencing a certain insensitivity to exact parameter choices.

A.7 Statistics of Loss function and Gradient norm decomposition

Here we provide more illustration of gradient and loss magnitudes during training of our simple-CONV. Figure 5 shows the loss dynamics of several models decomposing the loss according to correctly and incorrectly labeled test samples. We zoom in with Figure 6 that shows the loss on correctly labeled data, which is otherwise dominated by the loss on mislabeled data in Figure 5. We see clearly that for models trained with our Adv-KIP dataset and the RFD, the loss on mislabeled samples increases during training, while for adversarial training it decreases.

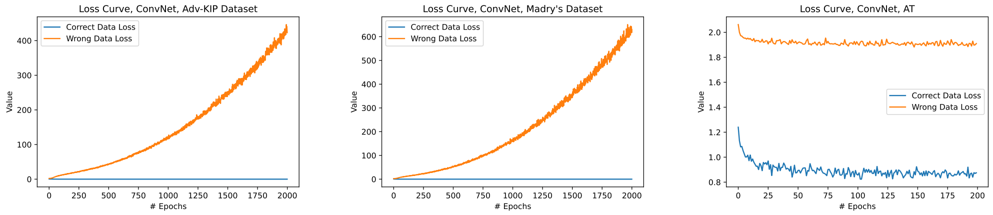

Figure 5: Test Loss Curves of models trained on our Adv-KIP Dataset, on the RFD, and during Adversarial Training. For the first two, the loss on incorrectly labeled samples increases, while for AT it decreases.

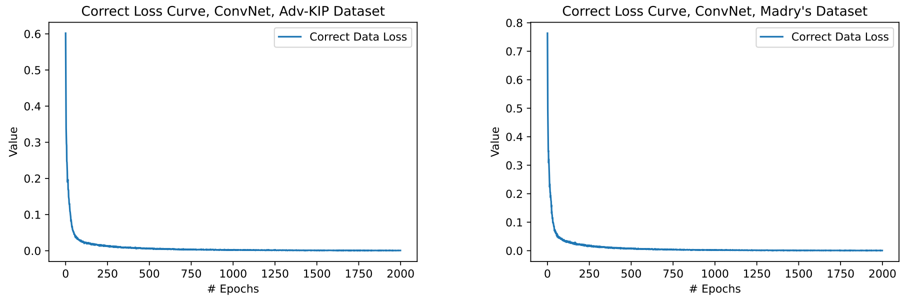

Figure 6: Test Loss Curves of models trained on our adv-KIP Dataset, the RFD, and with Adversarial Training on *correctly labeled* test data.

Figures 7 and 8 (zooming in on correctly labeled data) show a similar dynamics for the gradient norm. Again, models trained with our Adv-KIP Dataset and the RFD have explosively increasing gradients on incorrectly labeled samples.

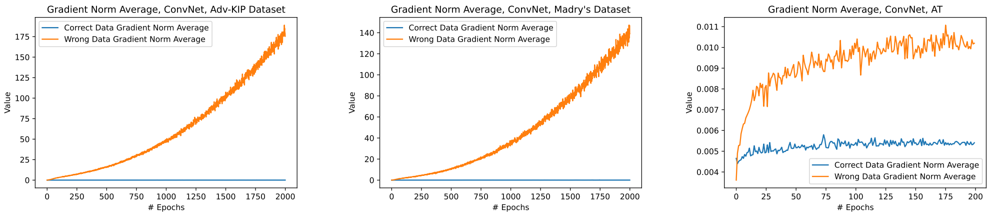

Figure 7: Gradient Norm Curves of models trained on our Adv-KIP Dataset, the RFD, and with adversarial training. For the first two, the loss on incorrectly labeled samples increases, while for AT it converges.

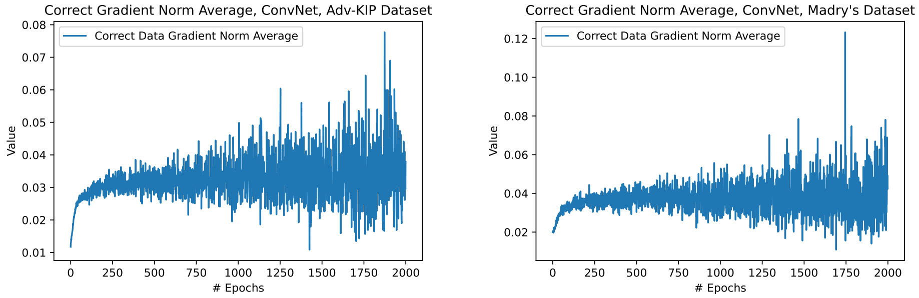

Figure 8: Gradient Norm Curves of models trained on our adv-KIP Dataset, the RFD and with Adversarial Training on correctly labeled data.

A.8 Kernel gradient norms

A.9 Results Using modifications of adv-KIP

Here we check several modifications of the adv-KIP algorithm to see if they prevent the data to settle for ill-behaved representations. We modify the inner loop objective with variations of different attacks (like the ones considered in AutoAttack) to verify if this provides a broader defense. We test a mixture of clean and robust loss for the outer loop of Algorithm 1, as in (Zhang et al., 2019). Lastly, we modify the target set $X_\mathcal{T}$ to a smaller set that only contain correctly labeled data, to prevent the optimization from ”compensating” on mislabeled target data. However, none of these modifications changes the picture and we invariably find that the resulting networks lack any true robustness.

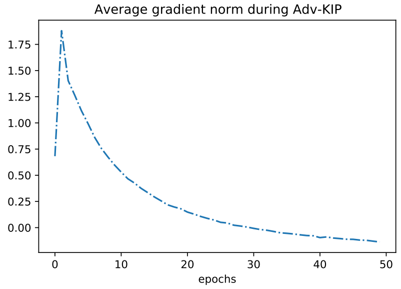

Figure 9: The average gradient norm of an FC3 kernel on a validation set during the distillation procedure of Algorithm 1. We see that the training data evolves to cause gradient shrinkage of the model. Setting: CIFAR-10, FC3, $|\mathcal{X}_S|$ = 40k, $|\mathcal{X}_T|$ = 10k, 10 PGD steps, cross entropy loss in outer loop.

For our Adv-KIP min-max procedure, in the inner loop we choose multi-step PGD with CE loss as the attacker since it is the gold standard in the current literature. For the outer loop, we compared of CE and MSE loss and settled for CE due to slightly better results. Given our failure to extract true robustness, we test whether we can distill more fruitful "robust features" into our dataset, by varying both inner and outer loss functions. For the inner loop, we replace the CE with the DLR loss, which is an extension of the CW loss (Carlini & Wagner, 2017). For the outer loop, instead of using pure CE, we incorporate the TRADES loss (Zhang et al., 2019) as an ablation. We also ran a variant of addKIP (with CE-loss) on CIFAR-10 with a smaller $|X_\mathcal{T}| = 5K$ such that it only contains correctly labeled data, to check if the obfuscated gradient phenomenon results from optimization on the misclassified part of $X_\mathcal{T}$.

Let z represent the pre-softmax logits. Recall that CE loss is defined as:

$$\mathrm{CE}(x, y) = -z_y + \log(\sum_{i=1}^{K} e^{z_j}) \tag{9}$$

Carlini & Wagner (2017) proposed to use the following (CW) loss to perturb the input:

$$\mathrm{CW}(x, y) = -z_y + \max_{i \neq y} z_i \tag{10}$$

We implement a variant of the above loss, namely the Difference of Logits Ratio (DLR) loss proposed in (Croce & Hein, 2020):

$$\mathrm{DLR}(x, y) = -\frac{z_y - \max_{i \neq y} z_i}{z_{\pi_1} - z_{\pi_3}} \tag{11}$$

where π is the ordering of the components of z in decreasing order (the untargeted version). This loss is invariant to scaling of the logits, and it has been used to detect cases where attacking the CE loss fails due to overconfidence of the model.

The TRADES loss (Zhang et al., 2019) aims to trade off robustness and accuracy. Given a specific input (x, y), TRADES optimizes over

$$\mathcal{L}(f(x;\theta), y) + \lambda \max_{x' \in \mathcal{B}(x)} \mathcal{L}(f(x;\theta), f(x';\theta)). \tag{12}$$

In our case,

$$\mathcal{L}(K_{\mathcal{X}_T \mathcal{X}_S} K^{-1}_{\mathcal{X}_S \mathcal{X}_S} \mathcal{Y}_S, \mathcal{Y}_T) + \lambda \max_{\mathcal{X}_{T'} \in \mathcal{B}(\mathcal{X}_T)} \mathcal{L}(K_{\mathcal{X}_T \mathcal{X}_S} K^{-1}_{\mathcal{X}_S \mathcal{X}_S} \mathcal{Y}_S, K_{\mathcal{X}_{T'} \mathcal{X}_S} K^{-1}_{\mathcal{X}_S \mathcal{X}_S} \mathcal{Y}_S). \tag{13}$$

A.9.1 Kernel Results

We first provide robustness on the kernel directly, using either the CE or DLR loss in the iterated attack. Results are shown in Table 9.

For the version of av-KIP that only collects correctly labeled test samples n $X_\mathcal{T}$, CE and DLR robustness do not improve and remain at 19.9% and 15.2% validation robust accuracy respectively for kernels.

	CE	DLR
CE	21.37%	16.89%
DLR	18.16%	13.35%
TRADES ($\lambda = 100$)	5.14%	5.60%

Table 9: Test accuracy evaluated on the best dataset found in advKIP (selected from a separate validation set). Row shows method used inside the inner loop. Column denotes evaluation attack. Setup: CIFAR-10, size of support set is 40k (and 30k for TRADES), size of target set is 10k, FC3 kernel.

A.9.2 Results on Convolutional Nets

Table 10, 11 and 12 show results with our DLR / TRADES datasets for CIFAR-10 for our convolutional neural nets. As we can see, the loss used to optimize the data clearly is **not** the trigger towards fake robustness, since none of these models exhibit robustness against attacks besides PGD.

			Robust	
Neural Net	**Clean**	FGSM	PGD ℓ_∞ 20	AA
Simple CNN	70.87 ± 0.44	65.06 ± 0.52	64.88 ± 0.51	0.00 ± 0.00
AlexNet	63.58 ± 5.50	47.62 ± 8.51	47.08 ± 8.65	0.11 ± 0.11
VGG11	73.72 ± 1.90	63.05 ± 4.14	62.76 ± 4.40	2.11 ± 3.28

Table 10: Test accuracies of several convolutional architectures trained on our Adv-KIP CIFAR-10 **DLR** dataset from the FC3 kernel.

			Robust	
Neural Net	**Clean**	FGSM	PGD ℓ_∞ 20	AA
Simple CNN	67.39 ± 0.18	58.21 ± 0.33	58.03 ± 0.34	0.00 ± 0.00
AlexNet	59.69 ± 1.13	47.44 ± 8.00	47.08 ± 8.47	0.29 ± 0.17
VGG11	68.31 ± 1.17	60.97 ± 3.37	60.61 ± 3.46	3.58 ± 2.01

Table 11: Test accuracies of several convolutional architectures trained on our Adv-KIP CIFAR-10 **TRADES** dataset from the FC3 kernel.

A.10 Details on the Confidence and Reliability Visualization

It has been shown that calibration of modern neural networks can be poor, despite advances in accuracy (Guo et al., 2017; Lakshminarayanan et al., 2017; Wenzel et al., 2020; Havasi et al., 2020). Guo et al. (2017) point out that more accurate and larger models tend to have worse calibration. A common measurement of miscalibration is the Expected Calibration Error (ECE) (Naeini et al., 2015), which quantifies the difference in expectation between confidence and accuracy using binning. Since obtaining accurate estimation of ECE is difficult, due to the dependency of the estimator on the binning scheme, we adopt the reliability diagram (DeGroot & Fienberg, 1983; Niculescu-Mizil & Caruana, 2005) and the confidence histogram (Guo et al., 2017), both tools with nice visualization. In the confidence histogram, we display the distribution of the predicted confidence, *i.e.,* the output probability of the predicted label, as a histogram. In the reliability diagram, we calculate the expected sample accuracy as a function of the confidence level by grouping all samples by their confidence. For a well-calibrated model, the reliability diagram should output the identity function, so we also plot the gap between the well-calibrated accuracy v.s. real accuracy.

A.11 Visualization of advKIP distilled images

			Robust	
Neural Net	**Clean**	FGSM	PGD ℓ_∞ 20	AA
Simple CNN	70.70 ± 0.38	65.22 ± 0.60	64.84 ± 0.56	0.00 ± 0.00

Table 12: Test accuracies of several convolutional architectures trained on our Adv-KIP CIFAR-10 dataset which only optimizes over **correctly classified data** from the FC3 kernel as described in Sec. A.9.1.

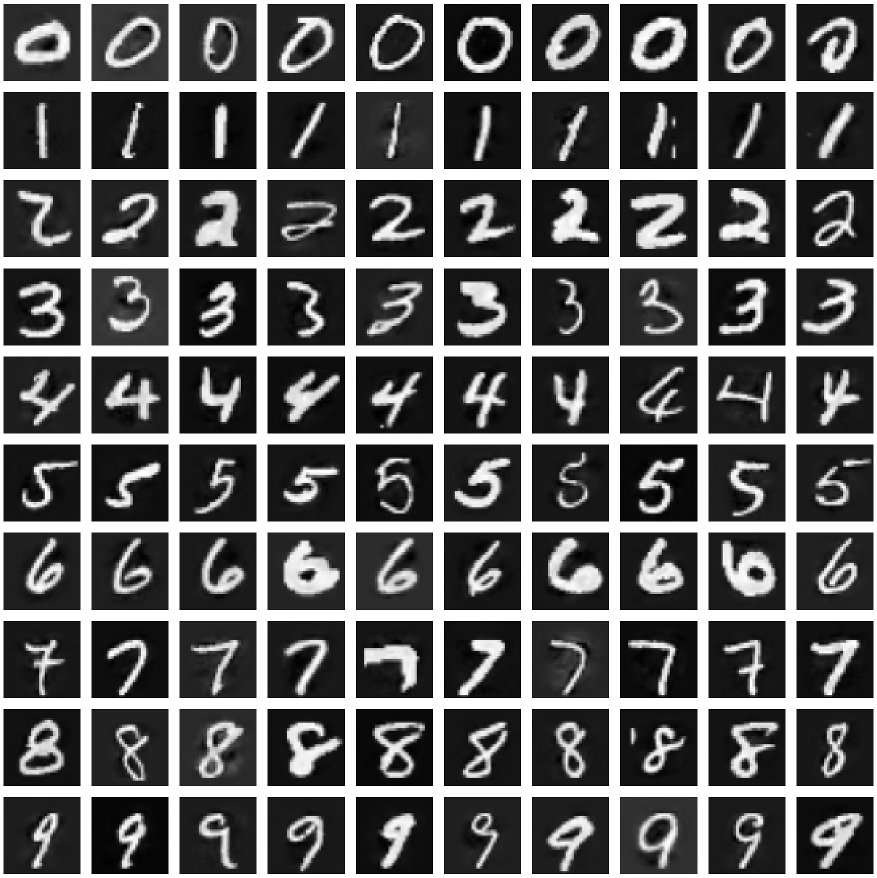

Figure 10: MNIST distilled images with trained labels from an FC7 kernel

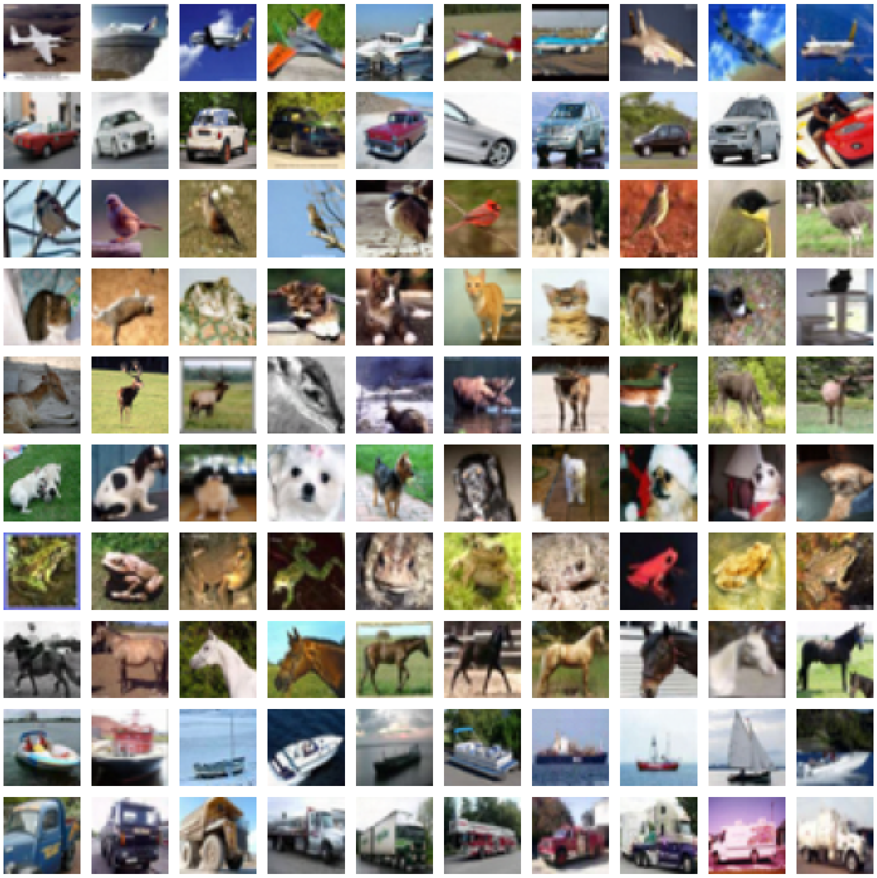

Figure 11: CIFAR-10 distilled images with trained labels from an FC3 kernel