Can we achieve robustness from data alone?

Nikolaos Tsilivis [1] **Jingtong Su** [1] **Julia Kempe** [1]

Abstract

Adversarial training and its variants have come to be the prevailing methods to achieve adversarially robust classification using neural networks. However, its increased computational cost together with the significant gap between standard and robust performance hinder progress and beg the question of whether we can do better. In this work, we take a step back and ask: *Can models achieve robustness via standard training on a suitably optimized set?* To this end, we devise a meta-learning method for robust classification, that optimizes the dataset prior to its deployment in a principled way, and aims to effectively remove the non-robust parts of the data. We cast our optimization method as a multi-step PGD procedure on kernel regression, with a class of kernels that describe infinitely wide neural nets (Neural Tangent Kernels - NTKs). Experiments on MNIST and CIFAR-10 demonstrate that the datasets we produce enjoy very high robustness against PGD attacks, when deployed in both kernel regression classifiers and neural networks. However, this robustness is somewhat fallacious, as alternative attacks manage to fool the models, which we find to be the case for previous similar works in the literature as well. We discuss potential reasons for this and outline further avenues of research.

1. Introduction

The discovery of the adversarial vulnerability of neural nets (Szegedy et al., 2014; Goodfellow et al., 2015; Papernot et al., 2017; Carlini & Wagner, 2017) - their brittleness when exposed to imperceptible perturbations in the data - has shifted the focus of the machine learning community from standard gradient techniques to more complex training algorithms that are rooted in robust optimization. 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)

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

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.

Since solving this problem is generally intractable, it is common to employ an iterative algorithm that interchangeably performs steps of gradient ascent/descent, a procedure called adversarial training (Goodfellow et al., 2015; Madry et al., 2018; Zhang et al., 2019). Here, training data is augmented on the fly through perturbations coming from the very model it is training. While adversarial training in its many variants has been successful and constitutes state-of-the-art, it is often expensive and not easily analysable.

However, once we focus on non-parametric models f (such as kernel ridge regression), we can pose a more "direct" problem

$$\inf_{\mathcal{D}_{\text{train}}} \mathbb{E}_{(x,y)\sim\mathcal{P}} \sup_{\delta\in\Delta} \mathcal{L}(f(x+\delta;\mathcal{D}_{\text{train}}),y), \qquad (2)$$

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. (2), 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., 2019b), yet in some cases has shown considerable transfer properties to commonly used neural

[1]Center for Data Science, New York University. Correspondence to: Nikolaos Tsilivis <nt2231@nyu.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

nets. The analytical tools afforded by the rich theory of kernels have resulted in progress in understanding the optimization landscape and generalization capabilities of neural networks (Du et al., 2019; Arora et al., 2019a), together with the discovery of interesting deep learning phenomena (Fort et al., 2020; Ortiz-Jiménez et al., 2021), while also inspiring practical advances in diverse areas of applications such as the design of better classifiers (Shankar et al., 2020), efficient neural architecture search (Chen et al., 2021), low-dimensional tasks in graphics (Tancik et al., 2020) and dataset distillation (Nguyen et al., 2021a;b).

Dataset Distillation. The inspiration for this work 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 has better robustness properties.

Distributionally Robust Optimization and Adversarial Augmentation. Related to our paper 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).

Our work 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.

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.

In summary, our contributions are as follows:

1. We propose adv-KIP, an algorithm of *training-data optimization* to achieve robustness via standard training, using NTKs. We point to the underlying framework that can be adapted to a range of two-loop (min-max) opimization tasks.

2. We show experimentally on MNIST and CIFAR-10 that our algorithm produces robust classifiers with (NTK) kernel regression. We further demonstrate that the extracted data set transfers well to finite width wide neural nets of similar architecture, leading to even better robustness than for the underlying kernels.

3. We then show that our kernel-generated datasets give rather surprising robustness against PGD attacks on a set of commonly used neural nets (a simple 3-layer CONV-net, AlexNet and VGG) for CIFAR-10, even though they are optimized via a simple fully-connected kernel.

4. Lastly, we discuss current shortcomings of our method. In particular, while our current algorithm, which optimizes against PGD attacks, achieves PGD-robustness, it remains susceptible to other attacks (e.g. (Carlini & Wagner, 2017)) and in particular does not stand the test of the current best-practice *AutoAttack* suite (Croce & Hein, 2020; 2021). We show that the same, incidentally, to even stronger extent, holds for the data set of (Ilyas et al., 2019). We discuss possible reasons and outline avenues to modify our algorithm with the goal of achieving true robustness.

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

$$\mathbf{x}^{k+1} = \Pi_{\mathcal{B}_{\mathbf{x}^0}^{\epsilon}} \left(\mathbf{x}^k + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}^k} \mathcal{L}(f(\mathbf{x}^k), y) \right), \quad (3)$$

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

$$\hat{f}(\mathbf{x}) = K(\mathbf{x}, \mathcal{X})^\top K(\mathcal{X}, \mathcal{X})^{-1}\mathcal{Y}, \tag{4}$$

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., 2019b). 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. (4), with a network-dependent kernel function k. Focusing on a scalar neural net for ease of notation, it is given by:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \nabla_\theta f(\mathbf{x}_i; \theta)^\top \nabla_\theta f(\mathbf{x}_j; \theta), \tag{5}$$

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 SOTA 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. (4) allows to express this objective via a loss function on a *target* data set $(\mathcal{X}_T, \mathcal{Y}_T)$ as:

$$\mathcal{L}_{\text{KIP}}(\mathcal{X}_S,\mathcal{Y}_S) = \|\mathcal{Y}_T - K(\mathcal{X}_T, \mathcal{X}_S)^\top K(\mathcal{X}_S, \mathcal{X}_S)^{-1}\mathcal{Y}_S\|_2. \tag{6}$$

The error of Eq. (6) 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 kernel functions for kernel regression renders the datasets suitable for deployment on actual neural nets as well.

3. Our Adv-KIP Algorithm

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. (6), 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 $\mathcal{L}_{\text{ce}}$ as is very common in adversarial training, but note that we have the freedom to chose any loss function, for instance 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. (2). 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 chose to apply a similar iterative procedure as in the PGD approach of Eq. (3). 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 $\mathcal{D}_{\text{train}} = \{\mathcal{X}, \mathcal{Y}\}$.
Output: A new dataset $\mathcal{D}_{\text{rob}}$.

1 Sample data $\mathcal{S} = \{\mathcal{X}_S, \mathcal{Y}_S\}$ from $\mathcal{D}_{\text{train}}$;
2 **for** i $\leftarrow 1$ **to** epochs **do**
3 Sample data $\mathcal{T} = \{\mathcal{X}_T, \mathcal{Y}_T\}$ from $\mathcal{D}_{\text{train}}$;
4 **for** j $\leftarrow 1$ **to** pgd_steps **do**
5 $\mathcal{X}_T \leftarrow \mathcal{X}_T + \alpha \cdot \text{sign}(\nabla_{\mathcal{X}_T}\mathcal{L}_{\text{ce}}(K_{\mathcal{X}_T\mathcal{X}_S}K^{-1}_{\mathcal{X}_S\mathcal{X}_S}\mathcal{Y}_S, \mathcal{Y}_T))$;
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 $\mathcal{D}_{\text{rob}} \leftarrow (\mathcal{X}_S, \mathcal{Y}_S)$

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 Squares Error (mse) (as in Eq. (7)) and Cross Entropy loss (ce). Experiments on MNIST suggest *ce* as the marginally better choice.

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 perfor-

mance, 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, as is the case for adversarial training.

4. Results on Kernels and Wide Neural Nets

Setup: We apply Adversarial KIP for learning robust datasets on MNIST (Deng, 2012) and CIFAR-10 (Krizhevsky, 2009) against ℓ_∞ adversaries of size ϵ equal to 0.3 and 8/255, respectively. 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 of depth $3, 5$ and 7 ($FC3, FC5, FC7$) and a 3-layer convolutional kernel (Conv3), and fully connected kernels of depth 2 and 3 for CIFAR-10. For FC kernels data set sizes are $|\mathcal{X}_S| = 30K$ (MNIST) and $40K$ (CIFAR10) with $|\mathcal{X}_T| = 10K$. 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 decrease. In all cases, robust performance is measured on adversarially perturbed original images.

Robustness on Kernels: In a first set of experiments with MNIST, we verify on the validation set that Algorithm 1 learns "robustness". 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.

Table 1 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

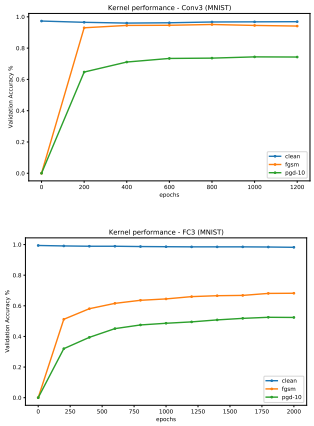

Figure 1. MNIST: Training curves on kernels. Shows validation performance as a function of training epochs. Setting: ce outer loss, trained labels, trained against 10 pgd. Top: CONV3, $|\mathcal{X}_S| = 5K$, $|\mathcal{X}_T| = 1K$ Bottom: FC3, $|\mathcal{X}_S| = 30K$, $|\mathcal{X}_T| = 10K$.

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	
Kernel, Dataset Size	**Clean**	FGSM	PGD10
Conv3, 5k	96.31	94.82	76.62
FC7, 30k	97.21	67.04	50.34

Table 1. MNIST Kernel results: Clean/robust test accuracy ($\%$).

Similar test results for CIFAR-10 are shown in Table 2. 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.

		Robust	
Kernel, Dataset Size	**Clean**	FGSM	PGD6
FC2, 40k	59.65	20.49	20.37
FC3, 40k	59.95	21.67	21.56

Table 2. CIFAR-10 Kernel results. Clean/robust test accuracy ($\%$).

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. It could be reasonable to hypothesize that such information compression might possibly lead to an increase of robustness as well, but we find that this is not at all the case. We follow up by producing a larger data set using the KIP algorithm (of the same size as used in our adv-KIP algo-

rithm) but find that even these large data sets do not provide any robust accuracy, neither against FGSM nor PGD attacks (see Appendix A.3). This indicates the clear need to adjust the optimization objective to robust performance, as is done in the adv-KIP algorithm.

Transfer results for wide FC networks: In a next step, 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). Tables 3 and 4 summarize our results for MNIST and CIFAR-10.

		Robust	
Kernel, Dataset Size	**Clean**	FGSM	PGD40
FC3, 30k	80.08	77.67	53.85
FC5, 30k	97.75	64.83	35.14
FC7, 30k	97.45	70.58	40.70

Table 3. Transferability (MNIST): Kernel to Neural Network of same architecture, test accuracy in %.

		Robust	
Kernel, Dataset Size	**Clean**	FGSM	PGD20
FC2, 40k	46.29	20.98	16.89
FC3, 40k	46.33	40.07	39.15

Table 4. Transferability (CIFAR-10): Kernel to Neural Network of same architecture, test accuracy in %.

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.

5. Experiments on common neural nets

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.

Models and Training Strategies. For MNIST, we train a simple three-layer CNN of width 64 with Max-Pooling on several Adv-KIP distilled datasets generated by Fully-Connected Kernels[1] ($FC3, FC5, FC7$) with dataset size of 30K. For CIFAR-10, we train this three-layer CNN, as well as two modern architectures, namely AlexNet (Krizhevsky et al., 2012) and VGG11 (Simonyan & Zisserman, 2015), in a similar manner for a dataset of size 50K produced with an $FC3$ kernel. In each case $|\mathcal{X}_T| = 10K$ and we optimize with 10 (MNIST) or 6 (CIFAR-10) pgd-steps. 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.

Tables 5 and 6 summarize our findings for MNIST and CIFAR-10. We note an astonishing "boost" in robust test accuracy on these convolutional networks when compared to the fully connected kernel results in Tables 1 and 2. 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
FC3	98.15 ± 0.12	98.06 ± 0.18	97.17 ± 0.10
FC5	97.96 ± 0.55	97.87 ± 0.64	97.20 ± 0.74
FC7	98.03 ± 0.16	97.91 ± 0.22	97.14 ± 0.43
Adversarial Training	99.11	97.52	95.82

Table 5. Simple-CNN test accuracies when trained on Adv-KIP MNIST datasets optimized with FC kernels (first 3 rows). We also show test accuracies obtained through adversarially training the simple-CNN (without any data augmentation).

		Robust	
Neural Net	**Clean**	FGSM	PGD20
Simple CNN	72.10 ± 0.10	67.45 ± 0.37	67.03 ± 0.24
AlexNet	68.87 ± 0.76	49.30 ± 0.69	49.06 ± 0.63
VGG11	74.88 ± 0.45	53.98 ± 9.71	53.18 ± 10.32

Table 6. Test accuracies of several convolutional architectures trained on our Adv-KIP CIFAR-10 dataset from the FC3 kernel.

	CIFAR-10 AT Baseline				
Neural Net	Clean	FGSM	PGD ℓ_∞ 20	PGD ℓ_2 20	AA
Simple CNN	58.07	33.94	31.49	43.89	26.18
AlexNet	44.35	30.12	24.41	16.68	18.95
VGG11	69.65	31.30	24.68	46.67	23.85

Table 7. Test accuracies for the adversarial training baseline on CIFAR-10. AA refers to *AutoAttack* test suite with ℓ_∞ (Sec. 6).

Figure 3 (App. A.2) shows the evolution of test 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.

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

Remarkably, it seems that modern networks trained with adv-KIP datasets 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 (see Table 7). This is particularly remarkable given we have not fine tuned or optimized our algorithm (for instance through preprocessing or data augmentation). We view this as a very promising direction for getting robustness though data alone, even if, as we outline in the next section, the current approach still falls short to guarantee widespread robustness.

6. Shortcomings and discussion

We have established that data sets optimized for robustness with the adv-KIP algorithm show excellent FGSM and PGD test accuracies when deployed in the wild, with modern convolutional architectures. Note that Algorithm 1 applies PGD-optimization in steps 4, 5 and 6 and we have so far only tested against PGD-type attacks.

When we deploy the AutoAttack suite, introduced in (Croce & Hein, 2020; 2021), to the convolutional nets studied in Sec. 5, we observe a sharp drop in robust test accuracy close to zero (see App. A.4). The purpose of the AutoAttack benchmark, which includes 4 different attacks, some of which do not use gradient information, is to provide a minimal adaptive attack suite to uncover shortcomings in the defense. In particular, adversarially trained networks retain their robustness against adaptive attacks such as AutoAttack (Croce & Hein, 2020) (see also Table 7).

We find that datasets produced with Algorithm 1 suffer from 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 either introducing non-differentiable parts in the inference pipeline or stochasticity to the model. Interestingly, we now observe this phenomenon from altering the *training* data alone and, even more remarkably, from data optimized using kernels.

To check whether this is a shortcoming of our optimization method or a more general phenomenon related to "robustified" data sets, we analyze the only dataset we are aware of that provides some notable robustness via standard training (Ilyas et al., 2019) ("Robust Feature Dataset" RFD). Recall that RFD is generated by traversing the representation layer of an adversarially trained (with PGD) neural network, and is thus believed to provide a general sense of robustness (Ilyas et al., 2019). We train the same networks as in Section 5 to ensure a fair comparison with our methods. Note that the publicly available RFD is derived from an adversarially trained network trained against an ℓ_2 adversary, so we include such an ℓ_2 evaluation in our results. Table 8 shows the test accuracies.

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

We find that neural networks trained with the RFD also suffer from the obfuscated gradient phenomenon, as they record high robustness against PGD attacks, but almost 0% against the adaptive suite of AutoAttack. This is a perhaps surprising finding, since the dataset was generated using adversarially trained networks that guarantee a wide sense of robustness. It could be an indication that achieving *true* robustness from data alone might be a challenging task when decoupled from the training algorithm. Better understanding how adversarial training with PGD techniques alone yields models that achieve broad robustness is key for improving the final robustness of data-centered approaches, like ours.

To further elucidate this mystery, we analyze both gradients and loss values during neural net training on our data set and the RFD dataset of (Ilyas et al., 2019). Figure 2 shows the loss value and the gradient magnitude of the model throughout training, where we further decomposed the values into 2 parts; one coming from correctly classified (clean) images and one from misclassified ones. We observe that for both datasets the loss increases on the misclassified examples, concurrently with an increase of the average norm of the gradient. In contrast, for correctly classified examples we see both quantities progressively vanish. 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. We find that during our distillation procedure this is indeed the case (Figure 4 in App. A.5), and the data optimization effectively shrinks the gradients of the model.

This analysis suggests several ideas on how to improve the distillation procedure. Penalizing small gradient norm in Algorithm 1 could prevent the data to settle for these ill-behaved representations, while modifying the inner loop objective with variations of different attacks (like the ones considered in AutoAttack) should provide a broader defense. A mixture of clean and robust loss for the outer loop of Algorithm 1, as in (Chen et al., 2021), could also be a promising direction, that could help prevent differing behaviors on misclassified and correct data. Based on our observations on the dataset of (Ilyas et al., 2019), we expect our insights from this work to be relevant for other data-based approaches for robust classification.

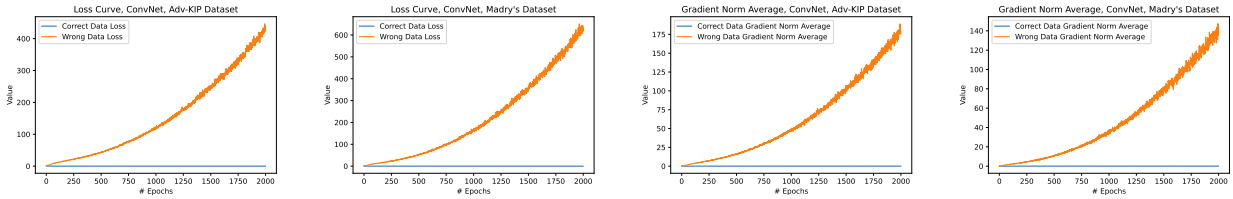

Figure 2. Loss and Gradient curves on test data during training of simple-CNN on datasets from adv-KIP and the RFD method of (Ilyas et al., 2019).

Acknowledgements

The authors would like to acknowledge support through the National Science Foundation under NSF Award 1922658.

References

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. Fine-Grained Analysis of Optimization and Generalization for Overparameterized Two-Layer Neural Networks. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 322–332. PMLR, 09–15 Jun 2019a.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. On exact computation with an infinitely wide neural net. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (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, 2019b.

Athalye, A., Carlini, N., and Wagner, D. A. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Dy, J. G. and Krause, A. (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.

Awasthi, P., Chatziafratis, V., Chen, X., and Vijayaraghavan, A. Adversarially robust low dimensional representations. In Belkin, M. and Kpotufe, S. (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.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Carlini, N. and Wagner, D. A. 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.

Chen, W., Gong, X., and Wang, Z. Neural architecture search on imagenet in four GPU hours: A theoretically inspired perspective. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

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

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

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

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In Chaudhuri, K. and Salakhutdinov, R. (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. 1675–1685. PMLR, 2019.

Fort, S., Dziugaite, G. K., Paul, M., Kharaghani, S., Roy, D. M., and Ganguli, S. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. *CoRR*, abs/2010.15110, 2020.

Garg, S., Sharan, V., Zhang, B. H., and Valiant, G. A spectral view of adversarially robust features. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (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.

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

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (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.

Jacot, A., Hongler, C., and Gabriel, F. Neural tangent kernel: Convergence and generalization in neural networks. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (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.

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

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

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

Kurakin, A., Goodfellow, I. J., and Bengio, S. 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.

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

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*, 2018.

Moosavi-Dezfooli, S., Fawzi, A., Fawzi, O., and Frossard, P. 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.

Nguyen, T., Chen, Z., and Lee, J. 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.

Nguyen, T., Novak, R., Xiao, L., and Lee, J. Dataset distillation with infinitely wide convolutional networks. *CoRR*, abs/2107.13034, 2021b.

Novak, R., Xiao, L., Hron, J., Lee, J., Alemi, A. A., Sohl-Dickstein, J., and Schoenholz, S. S. 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.

Ortiz-Jiménez, G., Moosavi-Dezfooli, S., and Frossard, P. What can linearized neural networks actually say about generalization? *CoRR*, abs/2106.06770, 2021.

Papernot, N., McDaniel, P. D., Goodfellow, I. J., Jha, S., Celik, Z. B., and Swami, A. Practical black-box attacks against machine learning. In Karri, R., Sinanoglu, O., Sadeghi, A., and Yi, X. (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.

Papernot, N., Faghri, F., Carlini, N., Goodfellow, I., Feinman, R., Kurakin, A., Xie, C., Sharma, Y., Brown, T., Roy, A., Matyasko, A., Behzadan, V., Hambardzumyan, K., Zhang, Z., Juang, Y.-L., Li, Z., Sheatsley, R., Garg, A., Uesato, J., Gierke, W., Dong, Y., Berthelot, D., Hendricks, P., Rauber, J., and Long, R. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

Shafahi, A., Najibi, M., Ghiasi, A., Xu, Z., Dickerson, J. P., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (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.

Shankar, V., Fang, A., Guo, W., Fridovich-Keil, S., Schmidt, L., Ragan-Kelley, J., and Recht, B. Neural kernels without tangents. *CoRR*, abs/2003.02237, 2020.

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

Sinha, A., Namkoong, H., and Duchi, J. C. 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.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks, 2014.

Tancik, M., Srinivasan, P. P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J. T., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Volpi, R., Namkoong, H., Sener, O., Duchi, J. C., Murino, V., and Savarese, S. Generalizing to unseen domains via adversarial data augmentation. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (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.

Wang, T., Zhu, J., Torralba, A., and Efros, A. A. Dataset distillation. *CoRR*, abs/1811.10959, 2018.

Wong, E., Rice, L., and Kolter, J. Z. 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.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically principled trade-off between robustness and accuracy. In Chaudhuri, K. and Salakhutdinov, R. (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. 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. 4, 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, we again train fully connected networks of width 1024 in Sec. 4, 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, 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.

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). For MNIST the radius is 0.3, and for CIFAR the radius is 8/255.

A.2. Evolution of accuracy during training for Simple ConvNet

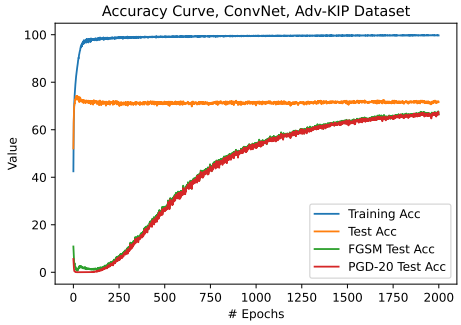

Figure 3. Evolution of accuracies during training with our 50K Adv-KIP dataset for CIFAR-10 on ConvNet.

A.3. 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. To check for robustness, we reproduce KIP datasets (Table 9), and find that effectively the robustness of the datasets remains close to 0, as is the case for the original datasets.

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

	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 9. 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.

FGSM robustness to be 0% in all cases. URLs for the datasets we considered: 1st and 2nd.

A.4. AutoAttack results on adv-KIP datasets

Here we provide test accuracies for alternative attacks on networks trained with our adv-KIP optimized datasets.

MNIST: For our simple-CNN, the test accuracy under *AutoAttack* for all three training methods (FC3, FC5, FC7, setting as in Table 5) becomes $0.00 \pm 0.00\%$, while it retains 88.77% accuracy under AA attack when adversarially trained.

CIFAR-10: Table 10 shows AA test accuracy for the setting of Table 6. We also show robust test accuracy against ℓ_2 PGD20 attack to show that our PGD robustness holds to even stronger extend for ℓ_2 (even though adv-KIP optimizes against ℓ_∞ in the inner loop).

Alternative attacks on adv-KIP CIFAR-10 dataset		
Neural Net	AA	PGD ℓ_2 20
Simple CNN	0.00 ± 0.00	70.79 ± 0.07
AlexNet	0.89 ± 1.41	50.07 ± 1.42
VGG11	0.27 ± 0.18	59.60 ± 6.71

Table 10. Test accuracies from alternative attacks of several convolutional architectures when trained on our Adv-KIP CIFAR-10 dataset from FC3 kernel.

A.5. Kernel gradient norms

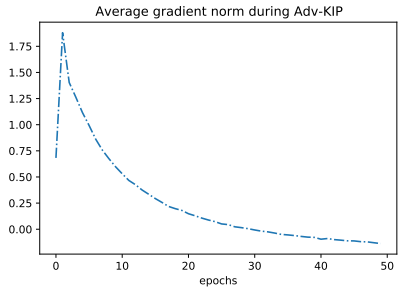

Figure 4. The average gradient norm of an FC3 kernel on a validation set during the distillation procedure of Algorithm 1. We see that the training data evolves to cause gradient shrinkage of the model. Setting: CIFAR-10, FC3, $|\mathcal{X}_S|$ = 40k, $|\mathcal{X}_T|$ = 10k, 10 PGD steps, cross entropy loss in outer loop.

A.6. Visualization of advKIP distilled images

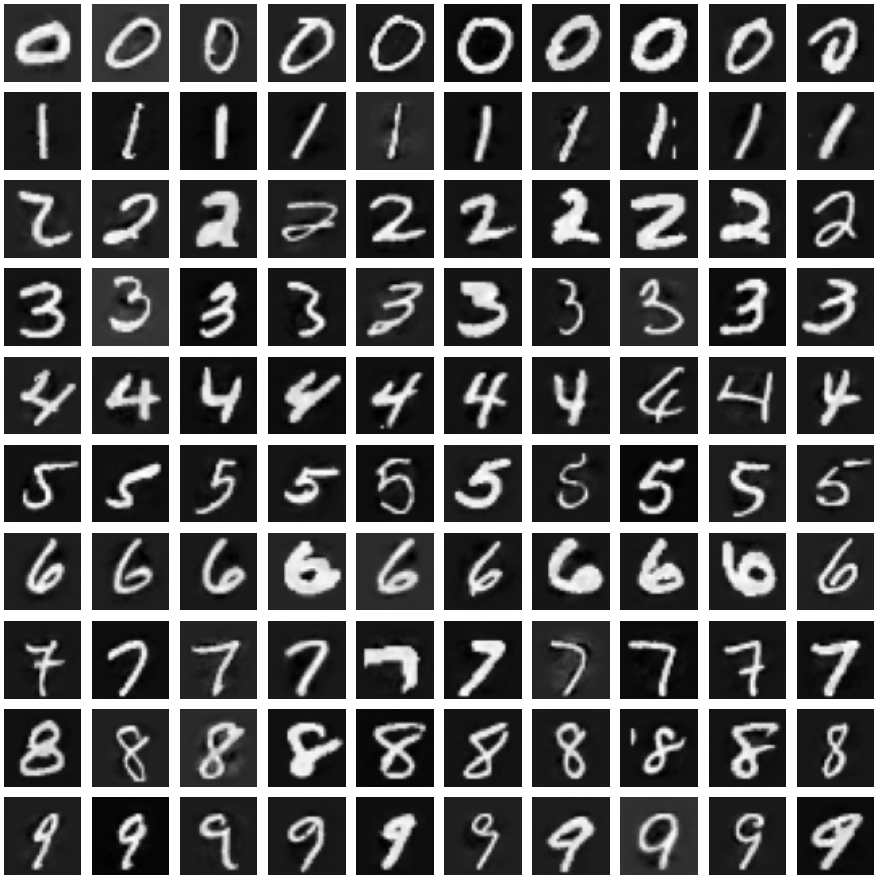

Figure 5. MNIST distilled images with trained labels from an FC7 kernel

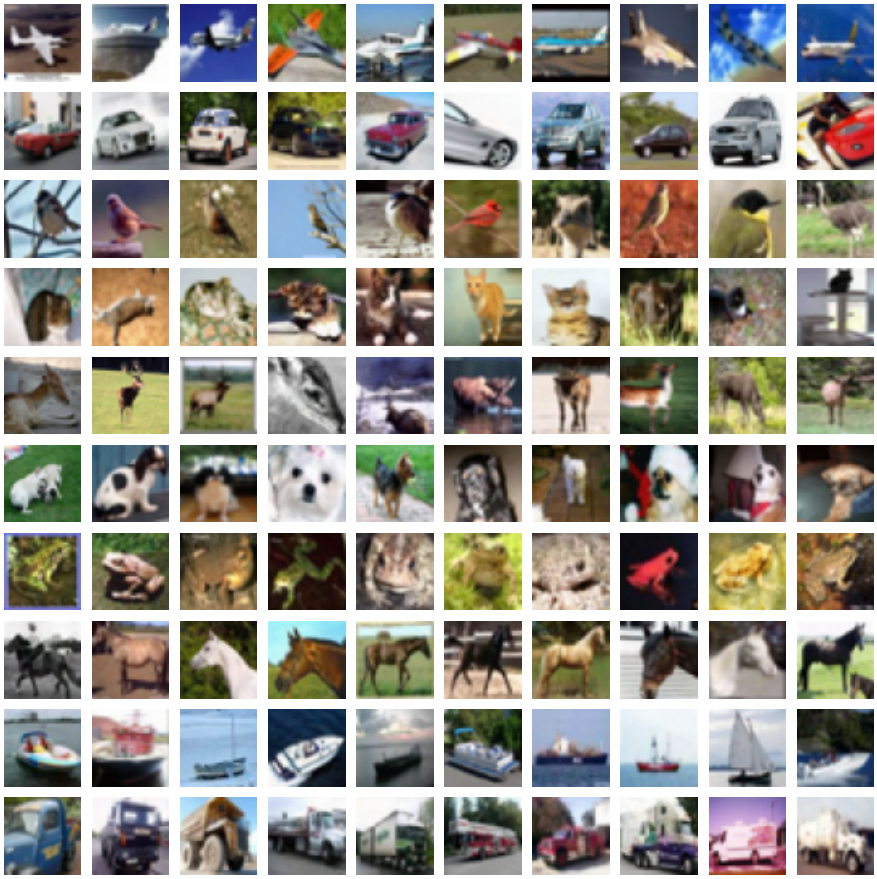

Figure 6. CIFAR-10 distilled images with trained labels from an FC3 kernel