Adversarial Dictionary Learning

Jordan Frecon, Lucas Anquetil, Yuan Liu, Gilles Gasso, Stéphane Canu

ADVERSARIAL DICTIONARY LEARNING

A PREPRINT

Jordan Frecon[†]
Normandie Univ, INSA Rouen
UNIROUEN, UNIHAVRE, LITIS
Saint-Etienne-du-Rouvray, France
jordan.frecon@insa-rouen.fr

Lucas Anquetil[*]
Normandie Univ, INSA Rouen
UNIROUEN, UNIHAVRE, LITIS
Saint-Etienne-du-Rouvray, France
lucas.anquetil@insa-rouen.fr

Yuan Liu[*]
Normandie Univ, INSA Rouen
UNIROUEN, UNIHAVRE, LITIS
Saint-Etienne-du-Rouvray, France
yuan.liu@insa-rouen.fr

Gilles Gasso
Normandie Univ, INSA Rouen
UNIROUEN, UNIHAVRE, LITIS
Saint-Etienne-du-Rouvray, France
gilles.gasso@insa-rouen.fr

Stéphane Canu
Normandie Univ, INSA Rouen
UNIROUEN, UNIHAVRE, LITIS
Saint-Etienne-du-Rouvray, France
stephane.canu@insa-rouen.fr

ABSTRACT

To bridge the gap between specific and universal attacks on deep classification networks, the present work frames the learning of multiple adversarial attacks as linear combinations of atoms from a dictionary of universal attacks. In order to learn such adversarial dictionary, a non-convex proximal splitting framework, termed as Adversarial Dictionary Learning (ADiL) is proposed. Numerical experiments evidence that the posteriori study of the dictionary atoms unveils the most common patterns to attack the classifier which, in turn, can be used to craft adversarial perturbations to new examples achieving great transferability on different deep network architectures.

Keywords Adversarial attacks; Dictionary learning

1 Introduction

With recent technological advances, the use of deep neural networks (DNN) have widespread to numerous applications ranging from biomedical imaging Min et al. [2017] to the design of autonomous vehicles Blin et al. [2019]. The reasons of their prosperity strongly rely on the increasingly large datasets becoming available, their high expressiveness and their empirical successes in various tasks (e.g. computer vision Akhtar and Mian [2018], natural language processing Young et al. [2018] or speech recognition Deng et al. [2013]).

However, their high representation power is also a weakness that some adversary might exploit to craft adversarial attacks which could potentially lead the DNN model to take unwanted actions Szegedy et al. [2014], Finlayson et al. [2019]. More precisely, adversarial attacks are almost imperceptible transformations aiming to modify an example well classified by a DNN into a new example, called adversarial, which is itself wrongly classified. To date, various attacks have been developed, the majority of them producing perturbations which, added to the original image, will change few pixels that cause the misclassification. In this regard, among the most popular are the fast gradient sign method (FGSM) Goodfellow et al. [2015], Kurakin et al. [2017], DeepFool Moosavi-Dezfooli et al. [2016], the projected gradient method (PGD) Madry et al. [2018] or the approach of Carlini and Wagner (CW) Carlini and Wagner [2017] which relies on the minimization of the perturbation's ℓ_p-norm. More recently, a functional attack, called ReColorAV, has been devised in Laidlaw and Feizi [2019] and intends to learn a perturbation function while applied to the input produces an adversarial example. A peculiarity of all these attacks is that they are *specific*, meaning they specifically produce one adversarial noise paired to one clean example. A contrario, in Moosavi-Dezfooli et al. [2017] the authors devised a *universal* perturbation, coined UAP, that can be applied to a whole set of images. However, although it is

[†] Corresponding author
[*] Equal contribution

universal, it is difficult to get precisely why it works in a case to case basis. More generally, state-of-the-art attacks still suffer from a lack of universality and understanding of the attack features from one image to another.

Contributions and Outline. This paper aims to bridge the gap between specific and universal attacks of deep networks for image classification. More precisely, it introduces an adversarial dictionary learning framework that allows each individual attack to be written as different linear combinations of the same dictionary elements (i.e. dictionary atoms). To get the dictionary, we first devise in Section 3.1 a supervised adversarial dictionary learning problem Mairal et al. [2009] where the supervision term is the fooling rate of the network. Two constraints, on the dictionary atoms and coding vectors, permit to enforce some maximum ℓ_p budget on the perturbation. The interests of the formulation are many: i) the learned dictionary can leverage relevant information across different images from different classes to efficiently fool the network, ii) some prior or properties such as orthogonality constraint can be enforced on the atoms, and iii) given the dictionary, the attack for any image can be scrutinized to gain in interpretation due to the linear combination.
To solve the involved dictionary learning problem, we design an algorithmic solution in Section 3.1.2 called Adversarial Dictionary Learning (ADiL) by hinging on the adaptation of the non-convex proximal splitting scheme of Sra [2012]. The proposed adversarial dictionary-based attacks are showcased on experimental evaluations in Section 4. Conducted experiments on both CIFAR-10 and Imagenet datasets reveal that our designed attacks achieve performance in between universal and specific attacks.

2 Preliminaries and Related Works

Consider a DNN $f\colon \mathbb{R}^P \mapsto \mathbb{R}^c$ used for image classification. DNN training consists of minimizing a loss H (typically cross-entropy) with respect to parameters θ to classify images drawn from an unknown joint distribution $\mathcal{P}(x, y)$ in which x belongs to some manifold $\mathcal{X} \subseteq \mathbb{R}^P$ (e.g., $\mathcal{X} = [0,1]^P$ or $[0,255]^P$) and $y \in \{1,\dots,c\}$ denotes one of the possible labels. Given an observed (also called natural or clean) example x, an adversarial example x' is a slight modification of x (e.g. such that $\|x - x'\| \leq \delta$, for some small $\delta > 0$) but having a different label prediction by f (i.e. fooling f), that is,

$$\underset{k\in\{1,\dots,c\}}{\operatorname{argmax}} f_k(x') \neq \underset{k\in\{1,\dots,c\}}{\operatorname{argmax}} f_k(x), \tag{1}$$

considering $f(x) = \Big(f_1(x),\dots,f_c(x)\Big)^\top$.

Specific Attacks. Numerous methods to generate adversarial examples exist (see for instance Silva and Najafirad [2020] for a review), with FGSM Goodfellow et al. [2015], Kurakin et al. [2017] being one of the first effective algorithms aiming to craft adversarial examples. The underlined idea is to perform a one-step δ optimization in the direction given by the sign of the gradient of the training loss $\nabla_x H(f(x), y)$ with respect to the input image x. Hence, the adversarial example to x_i associated with class y_i is expressed as:

$$x_i' = x_i + \varepsilon(x_i) \quad \text{where} \quad \varepsilon(x_i) = \delta \operatorname{sign}(\nabla_{x_i} H(f(x_i), y_i)), \tag{2}$$

with $\varepsilon(x_i)$ being the added perturbation to get the closest adversarial example to x_i. The PGD method Madry et al. [2018] improves upon FGSM by updating ε_i via projected gradient descent until fooling the DNN. In the same spirit, Carlini and Wagner Carlini and Wagner [2017] proposed a more elaborated approach that solves $\min_{\varepsilon_i\in\mathbb{R}^P} \|\varepsilon_i\|_p + \lambda g(x_i + \varepsilon_i)$ where g is an objective function enforcing the fooling of the classifier. A similar idea is pursued by DeepFool Moosavi-Dezfooli et al. [2016] which finds the specific perturbation as $\operatorname{argmin}_{\varepsilon_i\in\mathbb{R}^P} \|\varepsilon_i\|$, s.t. $\operatorname{argmax}_k f(x_i+\varepsilon_i) \neq \operatorname{argmax}_k f(x_i)$. All these methods boil down to generating, for a given x_i, a sample-dependent perturbation $\varepsilon(x_i)$ such that $x_i' = x_i + \varepsilon(x_i)$ is adversarial to the DNN f. This requires, to attack f, to have access to the x_i to be processed. In that setting, the effectiveness of an adversarial perturbation hinges on the computation budget.

Universal Attacks. To overcome these difficulties, it has been shown that a universal perturbation ε, independent of x and, thus, image-agnostic, can be designed Moosavi-Dezfooli et al. [2017]. This universal perturbation ε allows, with high probability, to set up for any x an adversarial example $x' = x + \varepsilon$. While the prior attack, coined UAP, was built by finding a universal DeepFool perturbation, a gradient-based solution, hereafter termed UAP-PGD, was then proposed in Shafahi et al. [2020] as the solution of the following optimization problem:

$$\underset{\varepsilon\in\mathbb{R}^P}{\text{maximize}} \sum_{i=1}^{N} H(f(x_i + \varepsilon), y_i) \quad \text{s.t.} \quad \|\varepsilon\|_p \leq \delta\,, \tag{3}$$

for a given dataset $\{x_i, y_i\}_{i=1}^N$. More recently, two variants of UAP-PGD have been proposed. First, the work of Zhang et al. [2020a] have designed a universal perturbation CD-UAP fooling solely a given subset of classes. Second, the authors of Benz et al. [2021] have considered instead class-wise universal perturbations where a perturbation is crafted for each of the classes. Let us also note Fast-UAP Dai and Shu [2020] improving upon UAP Dai and Shu [2020] by additionally building upon the orientations of the individual DeepFool perturbations into the design of a universal perturbation. The reader is invited to refer to Zhang et al. [2021] for a survey on universal attacks. However, mainly because of poor fooling rates, *the actual applicability of those universal perturbation to fool DNN is still far-fetched* Chaubey et al. [2020].

Proposed Dictionary-Based Attack. We propose to learn the perturbation ε to any x as a linear coding over a dictionary D of M basic adversarial perturbations (called atoms hereafter). The atoms themselves are also learned based on some adversary training set $\{x_i, y_i\}_{i=1}^N$. Hence such dictionary is independent from any specific input images to attack and may be deemed universal. On the contrary, the linear combination have to be tailored for each image at hand. In fact, our adversarial dictionary-based attack bridges the gap between specific and universal attacks. Close to our approach is the recent principal component (PC) adversarial example method Zhang et al. [2020b] that can be seen as an unsupervised adversarial dictionary learning with the dictionary given by the M leading PC of the training data. Instead, we rely on a supervised approach by taking into account the targeted fooling classes. In the following, we introduce this adversarial dictionary learning problem and expose a corresponding algorithmic solution.

3 Adversarial Dictionary Framework

The originality of the proposed method is to find an additive perturbation $\varepsilon(x)$ of some clean image x expressed as a linear combination of a few shared attacks. More formally, let μ be a distribution of images on $\mathcal{X} \subseteq \mathbb{R}^P$. We aim to learn a dictionary of attacks $D \in \mathbb{R}^{P\times M}$, made of $M \ll P$ atoms $d_j \in \mathbb{R}^P$, so that for every $x_i \sim \mu$, with high probability, $x_i + \varepsilon_i(x_i)$, with $\varepsilon_i(x_i) = Dv_i(x_i)$, is an adversary example for some vector $v_i(x_i) \in \mathbb{R}^M$. Note that D is shared across all attacks while $v_i(x_i)$ is tailored to the image x_i. In that sense, the proposed attack is semi-universal. To ease the reading, in what follows we drop the dependency and simply denotes v_i in place of $v_i(x_i)$.

3.1 Adversarial Dictionary Learning

In this section, we first present the proposed framework to find a dictionary of shared universal attacks, named ADiL for *Adversarial Dictionary Learning*. Then, an algorithmic solution is provided to learn such adversarial dictionary.

3.1.1 Principle

In order to learn the dictionary of shared attacks, we propose to address the following optimization problem reminiscent of classical dictionary learning problems (see, e.g., Mairal et al. [2009], Rakotomamonjy [2013]).

Problem 3.1 (Adversarial dictionary learning)**.** Let a classifier $f : \mathbb{R}^P \to \mathbb{R}^c$ and a dataset $\{x_i, y_i\}_{i=1}^N$ made of $N \in \mathbb{N}^+$ samples where each $x_i \in \mathbb{R}^P$ is a valid instance associated to the label $y_i \in \{1, \dots, c\}$. Solve

$$\underset{\substack{D\in\mathcal{D}\\ V=[v_1,\dots,v_N]\in\mathcal{V}^N}}{\text{minimize}} \quad \sum_{i=1}^{N} J_f(x_i + Dv_i, y_i), \tag{4}$$

where J_f is an adversarial loss whereas $\mathcal{D} \subseteq \mathbb{R}^{P\times M}$ and $\mathcal{V} \subseteq \mathbb{R}^M$ encode some constraints on D and the v_i's, respectively.

The choice of the adversarial loss plays a central role since it measures the closeness between the predicted target and a chosen target. It is typically quantified using a loss H such as the cross-entropy or the difference of logit outputs [Carlini and Wagner, 2017, see function f in Section VI.A]. As such, we can distinguish two settings:

$$\text{(untargeted)} \quad J_f(\cdot, y) = -H(f, y) \tag{5}$$

$$\text{(targeted)} \quad J_f(\cdot, y) = H(f, t) \tag{6}$$

On the one hand, the *untargeted* setting (5), promotes attacks so that the predicted target by f is far from the true label y. On the other hand, the *targeted* setting (6) aims to find attacks whose predicted label is an adversary target $t \in \{1, \dots, c\}$ chosen among the c classes.

Algorithm 1 ADiL

Require: Step-sizes $\{\gamma_k\}_{k=0}^{K-1}$
Set $D^{(0)} \sim \mathcal{D}$
Set $V^{(0)} = 0_{M\times N}$
for $k = 0$ to $K-1$ **do**
$D^{(k+1/2)} = \text{Proj}_{\mathcal{D}}(D^{(k)} - \gamma_k \nabla_D \mathcal{L}(D^{(k)}, V^{(k)}))$
$V^{(k+1/2)} = \text{Proj}_{\mathcal{V}^N}(V^{(k)} - \gamma_k \nabla_V \mathcal{L}(D^{(k)}, V^{(k)}))$
end for
return Adversarial dictionary $D^{(K)}$

Remark 3.1. There exist many ways to choose the adversary target t to an clean example x. A common choice is to set t as the second most probable label predicted by the DNN f. In other words, let $\hat{y} = \text{argmax}_{j\in\{1,\dots,c\}} f(x)$, then $t = \text{argmax}_{k\in\{1,\dots,c\}\setminus\{\hat{y}\}} f_k(x)$.

Depending on the choice of the constrained sets $\mathcal{D}$ and $\mathcal{V}$, one can ensure that the attacks are ℓ_p-norm constrained. This is the subject of the following proposition.

Proposition 3.1 (ℓ_p-Attacks)**.** *Given some budget $\delta > 0$, we have that $\|Dv\|_p \leq \delta$ for every $D \in \mathcal{D}$ and $v \in \mathcal{V}$ where*

$$\mathcal{D} = \{D \in \mathbb{R}^{P\times M} \mid (\forall m \in \{1,\dots,M\}), \|d_m\|_p \leq 1\}, \tag{7}$$

and

$$\mathcal{V} = \{v \in \mathbb{R}^M \mid \|v\|_1 \leq \delta\}. \tag{8}$$

Note that, for such choice, both $\mathcal{D}$ and $\mathcal{V}$ are convex sets. Additional comments can be found in the supplementary material.

3.1.2 Algorithmic Solution

Herein we propose a procedure for solving Problem 3.1. Note that minimizing the objective in (4) is a challenge due to the nonconvexity inherent to the dictionary learning formulation and the neural network f. We stress out that we are only interested in finding a good stationary point in a limited time. Since classical dictionary learning problems are bi-convex, they are usually solved by alternating the optimization over D and V since each alternating problem is convex. However, here this no longer the case because of the adversarial loss involving a neural network. Hence, we embrace a direct optimization scheme over (D, V) in the spirit of the nonconvex proximal splitting framework of Sra [2012] which has also been applied in the context of classical dictionary learning in Rakotomamonjy [2013]. To that purpose, we begin by recasting Problem 3.1 as follows.

$$\underset{\substack{D\in\mathcal{D}\\ V\in\mathcal{V}^N}}{\text{minimize}} \left\{\mathcal{L}(D,V) \triangleq \sum_{i=1}^{N} J_f(x_i + Dv_i, y_i)\right\}, \tag{9}$$

where $\mathcal{L}$ is smooth, provided that f is smooth as well. Note that, when f is not smooth, such as in the case of neural networks with non-smooth activation functions, it is unlikely that one lies at a discontinuity point during the forward pass when f is trained. However, in these cases, we consider a sub-gradient instead while still making use of the gradient notation for the ease of reading. Given some sequence of step-sizes $\{\gamma_k\}_{k\in\mathbb{N}}$, problem (9) is addressed by successive projected gradient steps of the form

$$\begin{cases} D^{(k+1/2)} &= \text{Proj}_{\mathcal{D}}\left(D^{(k)} - \gamma_k \nabla_D \mathcal{L}(D^{(k)}, V^{(k)})\right) \\ V^{(k+1/2)} &= \text{Proj}_{\mathcal{V}^N}\left(V^{(k)} - \gamma_k \nabla_V \mathcal{L}(D^{(k)}, V^{(k)})\right) \end{cases} \tag{10}$$

The full scheme is sketched in Algorithm 1.

Remark 3.2. In order to promote different dictionary elements while still being the least informative possible, we suggest to resort to a randomize initialization of D. The coding vectors are simply initialized as zero-valued vectors.

Remark 3.3. In order to lighten the memory load associated to both large-scale datasets and very deep neural architectures, one may consider stochastic variants instead.

3.2 Attacking Unseen Examples

We now turn to the crafting of dictionary-based attacks to unseen examples, coined ADiL attacks.

Provided that the dictionary D is known, we propose two ways to learn an attack $x + Dv$ to a new example $x \sim \mu$.

$$\text{(unsupervised)} \quad v \sim \mathcal{V} \tag{11}$$

$$\text{(supervised)} \quad v = \underset{u \in \mathcal{V}}{\operatorname{argmin}} J_f(x + Du, y) \tag{12}$$

The first is an *unsupervised* technique where v is randomly sampled in $\mathcal{V}$, i.e., the ℓ_1-ball of radius δ. In order to find perturbation closer to the given budget, we suggest to sample on the ℓ_1 sphere, instead. Various sampling strategies can be considered. The reader is invited to report to the supplementary material for additional details.

Remark 3.4. One can repeat the sampling of (11) multiple times until either the classifier f has been fooled or a number of maximum trials has been reached.

The second adversarial attack is, a contrario, a *supervised* technique where v is optimized by solving Problem 3.1 for a fixed D. This can be done by resorting to Algorithm 1 where the optimization steps over D have to be omitted. However, the learned perturbation Dv might still be far from the ℓ_p-budget δ. Instead, we suggest to solve an alternative problem of the form

$$\min_{v \in \mathbb{R}^M} J_f(x + Dv, y) \quad \text{s.t.} \quad \|Dv\|_p \leq \delta. \tag{13}$$

For the sake, let define $z = Dv$. As $D \in \mathbb{R}^{P \times M}$ with $M \ll P$ and by assuming that D is of full rank, we may write $v = D^\dagger z$ with $D^\dagger = (D^\top D)^{-1} D^\top$. Therefore, problem (13) becomes

$$\min_{z \in \mathbb{R}^P} J_f(x + DD^\dagger z, y) \quad \text{s.t.} \quad \|z\|_p \leq \delta \tag{14}$$

which can be solved via Algorithm 1 restricted to the optimization of z and by substituting $DD^\dagger$ to D in the optimization scheme.

For both *unsupervised* and *supervised* techniques, in order to ensure that the adversarial example $x + Dv$ is a valid image, we additionally perform a projection onto the input manifold $\mathcal{X} \subseteq \mathbb{R}^P$, i.e., $x' = \text{Proj}_{\mathcal{X}}(x + Dv)$.

4 Numerical Experiments

In this section, the proposed *supervised untargeted* ADiL attack (see (14) and (5)) is evaluated and compared with state-of-the-art attacks on benchmark datasets. To this effect, we resort to the TorchAttack repository Kim [2020] which contains Pytorch implementation of the most popular specific attacks. On the contrary, we have implemented by ourselves the universal attacks since their code was not made publicly available[1].

4.1 CIFAR-10 Experiments

In order to illustrate the good behavior of the proposed attack and gain some insights, we conducted the following experiments on the CIFAR-10 dataset Krizhevsky and Hinton [2009].

Setting. We consider a pre-trained VGG 11-layer model with batch normalization neural network f achieving a validation accuracy of 91.95% Phan [2021]. To this regard, the dictionary D is learned using a first subset made of $N = 700$ images (70 images per class out of the 10 classes) of the validation set. Then, a second subset of 150 images is sampled from the same validation set and is considered as the test set to evaluate to what extent D is relevant to craft attacks to unseen examples. In what follows, we resort to the cross-entropy loss H in (5).

Illustration. We report in Fig. 1 the learned adversarial dictionary (made of $M = 5$ atoms) for performing ℓ_2 attacks with a budget $\delta = 0.5$. Although all atoms are different, they still exhibit patterns in common. Indeed, since they are all used to fool a peculiar neural network f on a given dataset, they should bear the mark of both. For the VGG 11-layer

[1] All codes will be made available to contribute to the TorchAttack repository.

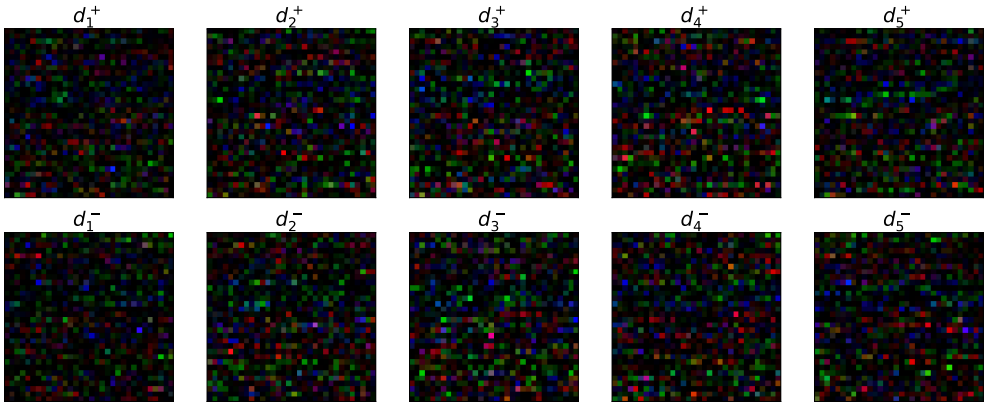


Figure 1: **Atoms of ℓ_2-ADiL on VGG trained on CIFAR-10.** Atoms are represented from *left* to *right* while their positive and negative parts are represented at the *top* and *bottom*, respectively. All atoms have been rescaled for display purposes

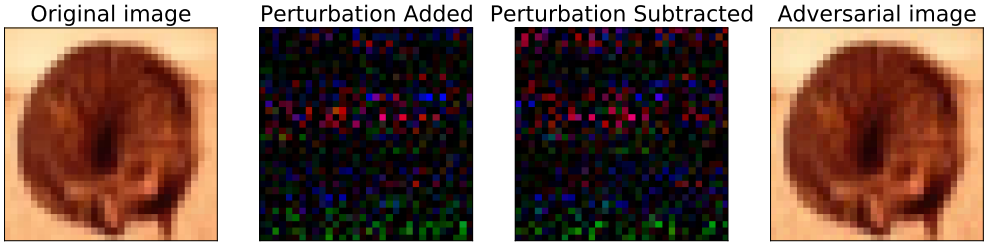


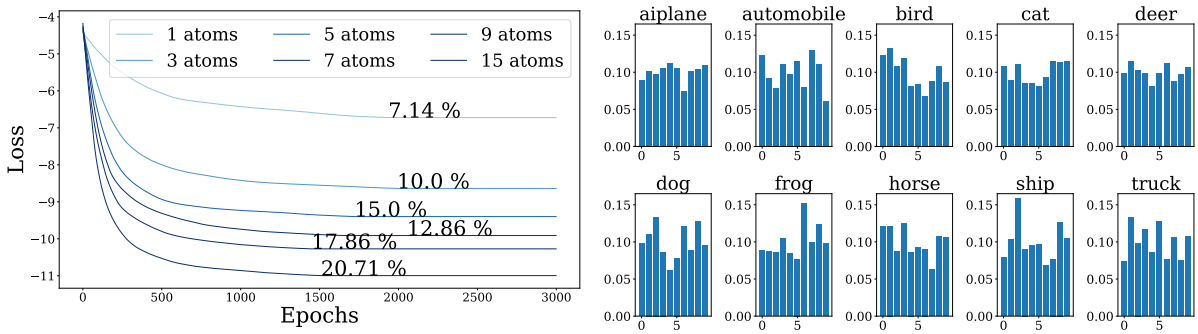
Figure 2: **Examples of ℓ_2-ADiL attacks on VGG trained on CIFAR-10.** Note that the adversarial noises have been magnified for display purposes. The original example being a cat has been attacked to be predicted as a frog. The coding vector for this attack is: $v = [0.3625, 0.1189, 0.0163, -0.2157, 0.1877]$

model studied here, this translates into small horizontal and vertical strides. Complementary experiments reported in the supplementary material highlight the difference in patterns for different f. We additionally report one ADiL attack in Fig. 2 where an image of a *cat* is attacked to be predicted as a *frog*. The corresponding coding vector v permits to gain some insight about how the atoms are linearly combined to produce the adversarial perturbation.

Impact of the Number of Atoms. In the proposed framework, the number of atoms M in the dictionary acts as an hyper-parameter controlling to what extent the attacks on different images are similar. On the one hand, for $M = 1$ each attack only differs by the intensity of the added perturbation through the quantity $v_i \in \mathbb{R}^M$. On the other hand, for $M = N$, there exist enough degrees of freedom so that attacks can be crafted specifically for each image. Here, we investigate how the choice of M impacts the performance of the attacks. We report the training losses for various number of atoms M in Fig. 3 (a) as well as the corresponding test fooling rates. We do observe that ADiL reaches lower loss values as M grows, hence confirming that better minima can be found by increasing the learning space. This fact is also seconded by the test fooling rates indicating better performance with larger M.

Distribution of Coding Vectors. For each of the 10 original classes, we *a posteriori* study the empirical distribution of the coding vectors v. To this purpose, we report in Fig. 3 (b) the quantity $|v|/\|v\|_1$ averaged over all attacks and conditioned to the original label. These distributions are good indicators of which of the $M = 10$ atoms are the most important to fool the network depending on the original class. We see that, on average, all atoms are used for each of the class. Indeed, this makes sense since no sparsity constraint is explicitly imposed on v. Interestingly, we observe different distributions for each of the classes, hence suggesting that the atoms might play different roles.

Now, we compare the proposed ADiL attack with both ℓ_2 and ℓ_∞ baseline attacks with budget 0.5 and $8/255$, respectively.



(a) Training loss and test fooling rate for various number of atoms M

(b) Histogram of coding vectors $v \in \mathbb{R}^M$ for each original classes

Figure 3: **Insights about ℓ_∞-ADiL attacks on CIFAR-10.**

Baselines. Comparisons are drawn with the following specific ℓ_p-attacks. For the sake of reproducibility, we also provide the grids on which the hyper-parameters are selected. If not mentioned, hyper-parameters values are kept as the default ones Kim [2020].

- FGSM Goodfellow et al. [2015] (ℓ_∞);
- FFGSM Wong et al. [2020a] (ℓ_∞) with step-size selected in $\{0.01, 0.02, 0.03\}$;
- MIFGSM Dong et al. [2018] (ℓ_∞) with step-size validated in $\{0.001, 0.006, 0.011, 0.016\}$;
- PGD Madry et al. [2018] (ℓ_2, ℓ_∞) with step-size in $\{0.1, 0.2, 0.3, 0.4\}$;
- C&W Carlini and Wagner [2017] (ℓ_2) with box-constraint parameter in $\{0.1, 1.77, 31.6, 562, 10^4\}$;
- APGD Croce and Hein [2020] (ℓ_2, ℓ_∞) with step-size chosen $\{0.5, 0.75, 1., 1.25\}$.

We additionally consider the following two universal attacks.

- UAP-PGD Shafahi et al. [2020] (ℓ_2, ℓ_∞) with step-size in $\{0.1, 2.575, 5.05, 7.525, 10\}$;
- FastUAP Dai and Shu [2020] (ℓ_2, ℓ_∞).

Attacks of State-of-the-Art Classifiers. The purpose of this experiment is to compare ADiL with both specific and universal attacks on a panel of pre-trained models (i.e., VGG11, DenseNet121 and ResNet50) depicting the variety of state-of-the-art neural network architectures. Performance, in terms of fooling rate, are reported in Table 1 (a) and Table 1 (b) for ℓ_2 and ℓ_∞-attacks, respectively.

Unsurprisingly, we observe an important difference of performance between universal and specific attacks. This difference is even greater with ℓ_2-based attacks (see Table 1 (a)) since universal attacks are known to perform relatively poorly in such setting. Overall, ADiL yields balanced results in between both ends of the spectrum. Such balance can be tuned by choosing the number atoms M, hence acting as a trade-off between specific and universal attacks.

4.2 ImageNet Experiments

In this section, we further investigate how ADiL performs in a more challenging large scale scenario such as ImageNet.

Experimental Setting. Experiments are conducted on the ILSVC2012 Russakovsky et al. [2015] validation subset of ImageNet consisting of 50K RGB images. They are resized and cropped around the center to match the input size of 224×224. The dataset is split into three parts with an equal distribution of samples among the 1000 classes. If not mentioned otherwise, we use 10 images per class for learning the attack, 2 for validating the hyper-parameters and 5 for for evaluating the attack performance. All experiments are carried out on GPU Volta V100-SXM2-32GB[2].

[2]Courtesy of CRIANN (`https://www.criann.fr`)

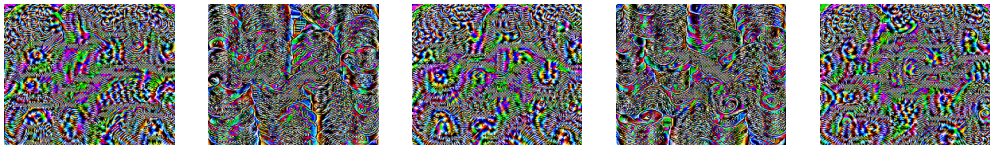
Attacks	VGG	DenseNet	ResNet50
PGD	96.43	95.89	**98.58**
CW	96.43	95.21	97.87
APGD	**97.14**	**97.26**	**98.58**
ADiL-2	05.00	05.48	04.26
ADiL-5	07.14	09.59	06.38
ADiL-10	09.29	10.27	07.09
UAP-PGD	01.43	00.68	00.71
Fast-UAP	01.43	02.05	01.42

(a) ℓ_2-attacks

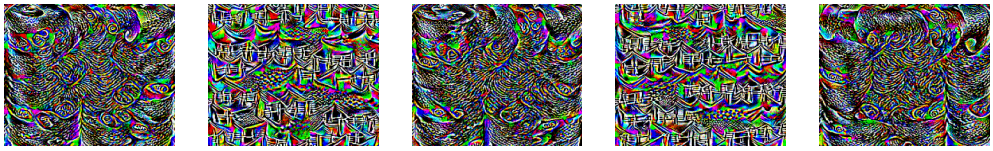
Attacks	VGG	DenseNet	ResNet50
APGD	**100**	**100**	**100**
FGSM	63.57	51.37	56.03
FFGSM	65.71	43.15	59.57
MIFGSM	**100**	**100**	**100**
PGD	**100**	**100**	**100**
ADiL-2	08.57	16.44	07.80
ADiL-5	15.00	19.18	11.35
ADiL-10	16.43	22.60	12.77
UAP-PGD	14.29	17.12	10.64
Fast-UAP	08.57	10.96	09.93

(b) ℓ_∞-attacks

Table 1: **Performance of ℓ_p-attacks on CIFAR-10.** Comparisons are drawn in terms of fooling rates between specific attacks (*top cell*), ADiL attacks with $M \in \{2, 5, 10\}$ atoms (*middle cell*) and universal attacks (*bottom cell*)



(a) MobileNet



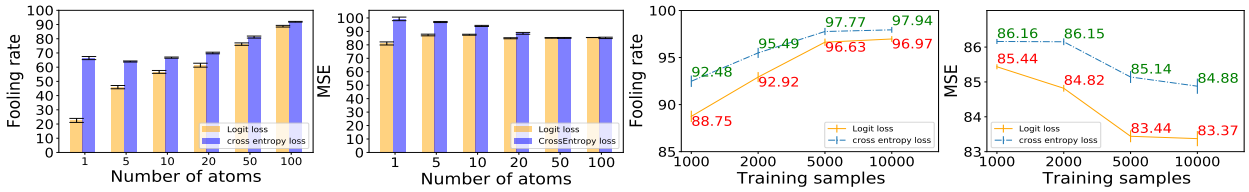
(b) GoogleNet

Figure 4: **Illustration of some ADiL's atoms on ImageNet.**

Departing from the CIFAR-10 experiments restricted to the cross-entropy loss H (see (5)), we additionally compare with the difference of logit outputs [Carlini and Wagner, 2017, see function f in Section VI.A], hereafter called logits loss. In order to deal with the computational cost inherent to the ImageNet dataset, we resort to the AdamW Loshchilov and Hutter [2017] stochastic gradient in place of the full-batch gradient of Algorithm 1. In what follows, we restrict the study to ℓ_∞-attacks of budget $\delta = 8/255$. The maximum number of iterations is set to 500 and 30 for learning the dictionary and crafting the attacks, respectively. Details and additional information about the selection of the hyper-parameters can be found in the supplementary material.

Impact of Number of Atoms. In order to complement the study done in Section 4.1, we further take a glance at the impact of the number of atoms M on the performance. To this regard, we have trained multiple dictionaries on $N = 1000$ samples so as to attack a MobileNet classifier. Results, displayed in Fig. 5a still support that a larger number of atoms relates to higher fooling rates but, more interestingly, also goes in pair with a lower mean squared error (mse). Let us also note that although ADiL with both cross-entropy and logits loss perform equally well for large M, they exhibit very different performance for small values of M. In that setting, we suggest to resort the cross-entropy instead. In the remaining of the experiments we solely consider $M = 100$ and further compare the cross-entropy loss against the logits loss.

Impact of Number of Training Samples. We additionally investigate the influence of the number of training samples on the quality of the learned dictionary on MobileNet. To this effect, we consider $N \in \{1000, 2000, 5000, 10000\}$ corresponding to respectively 1, 2, 5 and 10 samples per classes. Results are illustrated in Fig 5. As expected, the fooling rate increases as N grows. Note that the slight decrease in slope around 5000 samples suggest saturating performance beyond some sufficient number samples. Simultaneously, the mse decreases as N increases, thus indicating that the learned atoms are better fitted to the examples distribution. Finally, it is worth mentioning that, on merely



(a) Impact of number of atoms M

(b) Impact of number of samples N

Figure 5: **Impact of parameters.** We investigate how the parameters M and N influence the performance (fooling, rate) of ADiL with either cross-entropy loss (*blue*) or logits loss (*orange*)

1000 samples, one can already achieve competitive fooling rates about 88.75% using logits loss or even 92.48% for cross-entropy.

Comparison with State-of-the-Art Attacks. We now compare ADiL with the baseline attacks on multiple pre-trained models on ImageNet from the TorchVision repository. Namely, we consider MobileNetV2, DenseNet121, InceptionV3, ResNet18, GoogleNet and VGG11, achieving accuracy of 71.88%, 74.43%, 69.54, 69.76%, 69.78% and 69.02%, respectively. Performance in terms of fooling rate (fr), mean squared error (mse) and time are reported in Table 3.

On the one hand, results once again shed light on the superiority of specific attacks (e.g. AutoAttack Croce and Hein [2020]) managing to achieving up to 100% fooling rate. However, such performance comes at a price of high computational cost up to 10 times more than that required to perform universal attacks. The only two exceptions are FGSM Goodfellow et al. [2015] and its variant FFGSM Wong et al. [2020b] being both one-shot specific attacks. Still, their efficiency and low complexity comes at the expense of the highest mse of any attacks.

On the other hand, universal attacks perform relatively well in this large scale experiment mostly because of the many existing classes that allows them to easily find an adversarial label to target. Nonetheless, their fooling rates are still significantly far behind those of specific attacks.

Concerning ADiL, it displays competitive results at a much lower computational cost than specific attacks while also benefiting from low mse. Indeed, it performs at most 5 times faster than iterative specific attacks and still achieves relatively high fooling rates (e.g., 98.14% for ADiL-ce on MobileNet). In addition, ADiL always drastically improves upon universal attacks in terms of fooling for at most twice or three times their execution times and a comparable mse. Finally, it is unclear whether using the logits loss over the cross-entropy is beneficial since both yields equivalent results on average. Hence, confirming the observation drawn in the previous experiment. Therefore, in what follows we restrict to the cross-entropy loss.

Illustration of Atoms. We display in Fig. 4 a selection of 5 atoms of ADiL-100 learned on ImageNet for fooling either a MobileNet or a GoogleNet network. We observe that they all exhibit strong structured patterns reminiscent of those found in the UAP perturbation Shafahi et al. [2020]. However, contrary to universal attacks which merge all individual perturbations into a single one, the proposed dictionary framework allows to split the individual contributions onto multiple diverse atoms.

Cross-Model Performance. Finally, we study the cross-model performance, that is, to what extent the dictionary learned for attacking one model is also pertinent to attack another target model. Results, presented in the Table 4, show the powerful transferability of the proposed ADiL attack.

For most models, we do observe that the learned dictionary on one model is also the best dictionary to attack that aforementioned model (see diagonal of Table 4). Nonetheless, ADiL dictionary elements show great transferability across various models with an average decrease of 5% in fooling rate. It is also worth noting that learning a dictionary on the InceptionV3 model allows to craft attacks roughly matching the best performance on other models. This suggest that learning an adversarial dictionary on a deeper and more complex model may be a good way to craft quality attacks on shallower and simpler networks. More generally, it raises the question of which model attacks yield better generalization properties.

Attacks	VGG			DenseNet			GoogleNet		
	fr	mse	time	fr	mse	time	fr	mse	time
APGD	99.96	95.41	689	99.99	71.67	526	**100**	69.57	283
AutoAttack	**100**	95.45	688	**100**	**71.63**	536	**100**	69.57	291
FGSM	97.84	144.43	**21**	94.16	144.35	**55**	91.96	144.38	51
FFGSM	98.76	114.83	**21**	96.43	114.67	**55**	94.00	114.71	51
MIFGSM	99.93	94.05	594	99.99	74.46	460	**100**	73.08	233
PGD	99.96	91.74	593	99.99	73.10	458	**100**	71.69	234
ADiL-ce	86.08	90.76	140	88.97	88.12	187	84.12	88.82	101
ADiL-logits	84.52	89.02	140	87.86	86.41	183	86.76	86.05	101
UAP-PGD	68.44	121.70	59	68.14	102.75	67	72.72	120.65	56
UAP	60.37	**84.62**	71	57.52	79.28	67	42.15	74.32	**48**
Attacks	Inc.V3			ResNet18			MobileNet		
	fr	mse	time	fr	mse	time	fr	mse	time
APGD	99.99	**66.47**	910	**100**	**77.23**	180	**100**	**69.31**	364
AutoAttack	**100**	66.49	913	**100**	**77.23**	180	**100**	**69.31**	365
FGSM	83.40	140.50	65	97.89	144.41	**45**	96.16	144.39	51
FFGSM	85.35	112.93	65	99.08	114.70	**45**	98.77	114.74	51
MIFGSM	99.99	68.51	899	**100**	79.03	171	**100**	73.24	354
PGD	99.99	68.53	899	**100**	77.38	170	**100**	71.78	352
ADiL-ce	86.16	75.41	162	92.46	89.38	88	98.14	85.25	86
ADiL-logits	87.75	78.16	139	89.79	87.78	89	96.69	83.43	86
UAP-PGD	52.27	96.10	66	70.85	107.20	50	91.66	106.27	**46**
UAP	18.33	55.44	**61**	65.37	89.84	53	85.44	91.02	50

Table 3: **Performance of ℓ_∞-attacks on ImageNet.** Comparisons are drawn in terms of fooling rates (fr), mean squared error (mse) and average time for evaluating the attack (in ms). Results are divided between specific (*top cell*), ADiL with $M = 100$ atoms (*middle cell*) and universal attacks (*bottom cell*)

	VGG	DenseNet	GoogleNet	InceptionV3	ResNet18	MobileNet
VGG	86.08	84.02	79.79	82.12	88.37	91.63
DenseNet	84.24	**88.97**	79.16	81.16	89.65	92.02
GoogleNet	84.89	84.40	**84.12**	81.84	88.40	92.30
Inception	**86.09**	85.62	83.61	**86.16**	89.23	93.13
ResNet	84.77	85.87	78.94	80.61	**92.46**	92.40
MobileNet	83.92	81.84	75.74	78.50	86.13	**98.14**

Table 4: **Cross-model performance of ℓ_∞-attacks on ImageNet.** Models used to learn the adversarial dictionary are listed column-wise while models on which the attacks are evaluated are reported row-wise

5 Conclusions

The present paper introduces ADiL, a dictionary learning framework for finding adversarial perturbations in the form of a linear combination of shared dictionary elements. Once the dictionary is learned, it allows to craft an adversary perturbation in the manifold spanned by the dictionary atoms. Numerical experiments on a panel of pre-trained deep architectures show that ADiL offers competitive trade-off between specific and universal attacks by tuning the number of atoms. Future works will be devoted to the design of an efficient *unsupervised* ADiL attack by sampling the coding vectors (see (11)) through a generative adversarial network. We believe that such approach might permit to yield quasi instantaneous attacks without hampering too much the prediction performance. In addition, we plan to embrace a linear programming reformulation for ReLu based networks in order to find a global optima of Problem 3.1.

References

Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in bioinformatics*, 18(5): 851–869, 2017.

Rachel Blin, Samia Ainouz, Stéphane Canu, and Fabrice Meriaudeau. Road scenes analysis in adverse weather conditions by polarization-encoded images and adapted deep learning. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 27–32. IEEE, 2019.

Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8599–8603. IEEE, 2013.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Samuel G. Finlayson, John D. Bowers, Joichi Ito, Jonathan L. Zittrain, Andrew L. Beam, and Isaac S. Kohane. Adversarial attacks on medical machine learning. *Science*, 363(6433):1287–1289, 2019. ISSN 0036-8075.

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

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.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2574–2582. IEEE Computer Society, 2016.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

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

Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 10408–10418. Curran Associates, Inc., 2019.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis Bach. Supervised dictionary learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2009.

Suvrit Sra. Scalable nonconvex inexact proximal splitting. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

Samuel Henrique Silva and Peyman Najafirad. Opportunities and challenges in deep learning adversarial robustness: A survey. *arXiv preprint arXiv:2007.00753*, 2020.

Ali Shafahi, Mahyar Najibi, Zheng Xu, John Dickerson, Larry S. Davis, and Tom Goldstein. Universal adversarial training. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5636–5643, Apr. 2020.

Chaoning Zhang, Philipp Benz, Tooba Imtiaz, and In-So Kweon. Cd-uap: Class discriminative universal adversarial perturbation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):6754–6761, Apr. 2020a.

Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon. Universal adversarial training with class-wise perturbations. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2021.

Jiazhu Dai and Le Shu. Fast-uap: An algorithm for speeding up universal adversarial perturbation generation with orientation of perturbation vectors, 2020.

Chaoning Zhang, Philipp Benz, Chenguo Lin, Adil Karjauv, Jing Wu, and In So Kweon. A survey on universal adversarial attack. In *IJCAI*, pages 4687–4694, 2021.

Ashutosh Chaubey, Nikhil Agrawal, Kavya Barnwal, Keerat K Guliani, and Pramod Mehta. Universal adversarial perturbations: A survey. *arXiv preprint arXiv:2005.08087*, 2020.

Yonggang Zhang, Xinmei Tian, Ya Li, Xinchao Wang, and Dacheng Tao. Principal component adversarial example. *IEEE Transactions on Image Processing*, 29:4804–4815, 2020b.

Alain Rakotomamonjy. Direct optimization of the dictionary learning problem. *IEEE Transactions on Signal Processing*, 61(22):5495–5506, 2013.

Hoki Kim. Torchattacks: A pytorch repository for adversarial attacks. *arXiv preprint arXiv:2010.01950*, 2020.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009.

Huy Phan. Sofware huyvnphan/pytorch_cifar10, January 2021.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. *CoRR*, abs/2001.03994, 2020a.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum, 2018.

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

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020b.