Adversarial Example Games

Avishek Joey Bose[*]
Mila, McGill University
joey.bose@mail.mcgill.ca

Gauthier Gidel[*]
Mila, Université de Montréal
gauthier.gidel@umontreal.ca

Hugo Berard[*]
Mila, Université de Montréal
Facebook AI Research

Andre Cianflone
Mila, McGill University

Pascal Vincent[†]
Mila, Université de Montréal
Facebook AI Research

Simon Lacoste-Julien[†]
Mila, Université de Montréal

William L. Hamilton[†]
Mila, McGill University

Abstract

The existence of adversarial examples capable of fooling trained neural network classifiers calls for a much better understanding of possible attacks, in order to guide the development of safeguards against them. It includes attack methods in the highly challenging *non-interactive blackbox* setting, where adversarial attacks are generated without any access, including queries, to the target model. Prior works in this setting have relied mainly on algorithmic innovations derived from empirical observations (e.g., that momentum helps), and the field currently lacks a firm theoretical basis for understanding transferability in adversarial attacks. In this work, we address this gap and lay the theoretical foundations for crafting transferable adversarial examples to entire function classes. We introduce *Adversarial Examples Games* (AEG), a novel framework that models adversarial examples as two-player min-max games between an attack generator and a representative classifier. We prove that the saddle point of an AEG game corresponds to a generating distribution of adversarial examples against entire function classes. Training the generator only requires the ability to optimize a representative classifier from a given hypothesis class, enabling BlackBox transfer to unseen classifiers from the same class. We demonstrate the efficacy of our approach on the MNIST and CIFAR-10 datasets against both undefended and robustified models, achieving competitive performance with state-of-the-art BlackBox transfer approaches.

1 Introduction

Adversarial attacks on deep neural nets expose critical vulnerabilities in traditional machine learning systems [55, 3, 64, 8]. In order to develop better machine learning models robust to such adversarial attacks, it is imperative to improve our theoretical understanding of different attack strategies. While there has been considerable progress in understanding the theoretical underpinnings of adversarial attacks in relatively permissive settings (e.g. *whitebox* adversaries [53]), there remains a substantial gap between theory and practice in more demanding and realistic threat models.

In this work, we provide a theoretical framework for understanding and analyzing adversarial attacks in the highly-challenging ***No**n-interactive black**Box** adversary* (NoBox) setting, where the attacker

[*]Equal Contribution, order chosen via randomization.
[†]Canada CIFAR AI Chair

Preprint. Under review.

has no direct access, including input-output queries, to the target classifier it seeks to fool. Instead, the attacker must generate attacks by optimizing against some representative classifiers, which are assumed to come from a similar hypothesis class as the target, before applying these attacks in a single attempt against the target classifier.

The NoBox setting is a much more challenging setting than more traditional threat models, yet it is representative of many real-world attack scenarios, where the attacker cannot interact with the target model [14]. Indeed, this setting—as well as the general notion of transferring attacks between classifiers—has generated an increasing amount of empirical interest [23, 51, 73, 71], though the field currently lacks the necessary theoretical foundations to understand the feasibility of such attacks.

Present Work. To address this theoretical gap, we cast NoBox attacks as a kind of *adversarial examples game* (AEG). In this game, an attacker generates adversarial examples to fool a representative classifier from a given hypothesis class, while the classifier itself is trained to detect the correct labels from the adversarially generated examples. Our first main result shows that the Nash equilibrium of an AEG leads to a distribution of adversarial examples effective against *any* classifier from the given function class. More formally, this adversarial distribution is guaranteed to be the most effective distribution for attacking the hardest-to-fool classifiers within the hypothesis class, providing a worst-case guarantee for attack success against an arbitrary target. We further show that this optimal adversarial distribution admits a natural interpretation as being the distribution that maximizes a form of restricted conditional entropy over the target dataset, and we provide detailed analysis on simple parametric models to illustrate the characteristics of this optimal adversarial distribution. Note that while AEGs are latent games [28], they are distinct from the popular generative adversarial networks (GANs) [30]. In AEGs, there is *no* discrimination task between two datasets (generated one and real one); instead, there is a standard supervised (multi-class) classification task on an adversarial dataset.

Empirical results on standard benchmarks using models that directly instantiate the AEG framework show that we can achieve competitive performance—compared to existing heuristic approaches—in the NoBox setting while maintaining a firm theoretical grounding. In particular, we find that our framework can achieve state-of-the-art performance in many settings (e.g., attacking a held-out classifier with DenseNet architectures on CIFAR-10), while remaining highly competitive in others.

2 Background and Preliminaries

Suppose we are given a classifier $f : \mathcal{X} \to \mathcal{Y}$, an input datapoint $x \in \mathcal{X}$, and a class label $y \in \mathcal{Y}$, where $f(x) = y$. The goal of an adversarial attack is to produce an adversarial example $x' \in \mathcal{X}$, such that $f(x') \neq y$, and where the distance[3] $d(x, x') \leq \epsilon$. Intuitively, the attacker seeks to fool the classifier f into making the wrong prediction on a point x', which is ϵ-close to a real data example x.

2.1 Adversarial Attacks and Optimality

A popular setting in previous research is to focus on generating *optimal* attacks on a single classifier f [12, 53]. Given a loss function ℓ, used to evaluate f, an adversarial attack is said to be optimal if,

$$x' \in \operatorname{argmax}_{x' \in \mathcal{X}} \ell(f(x'), y)\,, \quad \text{s.t.} \quad d(x, x') \leq \epsilon\,. \tag{1}$$

In practice, attack strategies that aim to realize Equation 1 are conducted by optimizing adversarial examples x' directly using the gradient of f and then evaluating the attack success rate (i.e., how often these x' successfully fool f).

In this work, we consider the more general setting of generating attacks that are optimal against entire hypothesis classes $\mathcal{F}$. Approaches that seek to be optimal in this sense can be tested by evaluating attack success rates against unseen classifiers from the hypothesis class $\mathcal{F}$. As we later show in §3, under a game-theoretic lens the optimal adversary learns *conditional distribution* of adversarial examples that can attack the whole class $\mathcal{F}$.

2.2 NoBox Attacks

Threat models specify the formal assumptions of an attack (e.g., the information the attacker is assumed to have access to), which is a core aspect of adversarial attacks. For example, in the popular

[3] We assume that the ℓ_∞ is used in this work, [31, 53] , but our results generalize to any distance d.

whitebox threat model, the attacker is assumed to have full access to the model f's parameters and outputs [65, 31, 53, 55, 12, 17, 18]. In contrast, the *blackbox* threat model, the attacker has restricted access to the model, such as the type of observed input-output pairs from the target model, —i.e. logits, final prediction, or even a finite resource in the number of staged queries [16, 24, 39–41, 56]. Overall, while they consider different access to the model, traditional whitebox and blackbox attacks both attempt to generate adversarial examples that are optimal for a specific target (i.e., Equation 1).

In this paper, we consider the challenging setting of ***no**n-interactive black**Box** (**NoBox**)* attacks, intending to generate successful attacks against an unknown target.

In the NoBox setting, we assume no interactive access to a target model; instead, we only assume access to a dataset and knowledge of the function class to which a target model belongs. Specifically, the NoBox threat model relies on the following key definitions:

- **The target model,** f_t. The adversarial goal is to attack some target model $f_t : \mathcal{X} \to \mathcal{Y}$, which belongs to an hypothesis class $\mathcal{F}$. Critically, the adversary has *no access* to f_t *at any time*. Thus, in order to attack f_t, the adversary must develop attacks that are effective against the entirety of $\mathcal{F}$.
- **The test dataset,** $\mathcal{D}$. We assume there is a test dataset $\mathcal{D}$, which contains the examples (x, y) that the attacker seeks to corrupt. In addition, we may assume to have access to a dataset $\mathcal{D}_{\text{train}}$ sampled from the same or a similar data distribution to the one that was used to train the target model f_t. Notably, we can use such a dataset to train new models against which one can test our attacks.
- **An hypothesis class,** $\mathcal{F}$. As noted above, we assume that the attacker has access to a hypothesis class $\mathcal{F}$ to which the target model f_t belongs.[4] One can incorporate in $\mathcal{F}$ as much prior knowledge one has on f_t (e.g., the architecture, dataset, training method, or regularization), going from exact knowledge of the target $\mathcal{F} = \{f_t\}$ to almost no knowledge at all (e.g., $\mathcal{F} = \{f \in \text{DenseNets}\}$).
- **A representative classifier,** f_c. Finally, we assume that the attacker has the ability to optimize a representative classifier f_c from the hypothesis class $\mathcal{F}$.

Given these four key components, we formalize the NoBox setting as follows:

Definition 1. *The NoBox threat model corresponds to the setting where the attacker (i) has access to an attack dataset $\mathcal{D}$, and to a dataset $\mathcal{D}_{train}$ sampled from the data distribution as the dataset used to train f_t (ii) knows a hypothesis class $\mathcal{F}$ that the target model f_t belongs to, and (iii) can optimize a representative classifier $f_c \in \mathcal{F}$. The attacker has no other knowledge of—or access to—the target model f_t (e.g., no queries to f_t are allowed).*

Our definition of a non-interactive blackbox (NoBox) adversary (Def. 1) formalizes similar notions used in previous work (e.g., see Def. 3 in [67] with $\mathcal{D}_{\text{train}}$ added as in their experiments). Previous work also often refers to related settings as generating *blackbox transfer* attacks, since the goal is to attack the target model f_t while only having access to a representative classifier f_c [23, 51, 73].

3 Adversarial Example Games

In order to understand the theoretical feasibility of NoBox attacks, we propose to view the attack generation task as a form of *adversarial game*, where the players are the *generator* network g—which learns a conditional distribution over adversarial examples—and the representative classifier f_c. The goal of the generator network is to learn a conditional distribution of adversarial examples, which can fool the representative classifier f_c. The latter—on the other hand—is optimized to detect the true label y from the adversarial examples (x', y) generated by g. A critical insight in this framework is that the generator and the representative classifier are *jointly* optimized in a maximin game, making the generator's adversarial distribution at the Nash equilibrium theoretically effective against *any* classifier from the hypothesis class $\mathcal{F}$ that f_c is optimized over. At the same time, we will see in Proposition 1 that the min and max in our formulation (AEG) can be switched. It implies that, while optimized, the model f_c converges to a *robust classifier* against any attack generated by the generator g [53, 70], and leads to increasingly transferable attacks as the adversarial game progresses.

Framework. Given an input-output pair of datapoints $(x, y) \sim \mathcal{D}$, the generator network g is trained to learn a distribution of adversarial examples $p_{\text{cond}}(\cdot|x, y)$ that—conditioned on an example to attack

[4] Previous work [67] usually assumes to have access to the architecture of f_t; we are more general by assuming access to a hypothesis class $\mathcal{F}$ containing f_t; e.g., DenseNets can represent ConvNets.

(x, y)—maps a prior distribution p_z on $\mathcal{Z}$ onto a distribution on $\mathcal{X}$. The classifier network f_c is simultaneously optimized to perform robust classification over the resulting distribution p_g defined in (2). Overall, the generator g and the classifier f_c play the following, two-player zero-sum game:

$$\max_{g \in \mathcal{G}_\epsilon} \min_{f_c \in \mathcal{F}} \mathbb{E}_{(x,y)\sim\mathcal{D}, z\sim p_z}[\ell(f_c(g(x, y, z)), y)] =: \varphi(f_c, g) \tag{AEG}$$

where the generator $g \in \mathcal{G}_\epsilon$ is restricted by the similarity constraint $d(g(x, y, z), x) \leq \epsilon\,,\ \forall x, y, z \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$. Once the generator g is trained, one can generate adversarial examples against any classifier in $f_t \in \mathcal{F}$, without queries, by simply sampling $z \sim p_z$ and computing $g(x, y, z)$.

Connection with NoBox attacks. The NoBox threat model (Def. 1) corresponds to a setting where the attacker does not know the target model f_t but only a hypothesis class $\mathcal{F}$ such that $f_t \in \mathcal{F}$. With such knowledge, one cannot hope to be better than the *most pessimistic situation* where f_t is the best defender in $\mathcal{F}$. Our maximin formulation (AEG) encapsulates such a worst-case scenario where the generator aims at finding attacks against the best performing f in $\mathcal{F}$.

Objective of the generator. When trying to attack infinite capacity classifiers—i.e., $\mathcal{F}$ contains any measurable function—the goal of the generator can be seen as generating the adversarial distribution p_g with the highest expected conditional entropy $\mathbb{E}_x[\sum_y p_g(y|x) \log p_g(y|x)]$, where p_g is defined as

$$(x', y) \sim p_g \Leftrightarrow x' = g(x, y, z)\,,\ (x, y) \sim \mathcal{D}\,,\ z \sim p_z \quad \text{with} \quad d(x', x) \leq \epsilon\,. \tag{2}$$

When trying to attack a specific hypothesis class $\mathcal{F}$ (e.g., a particular convolutional architecture), the generator aims at maximizing a notion of restricted entropy defined implicitly through the class $\mathcal{F}$. Thus, the optimal generator in an (AEG) is primarily determined by the statistics of $\mathcal{D}$ itself, rather any specifics of a target model. We formalize these high level concepts in §4.2.

Biasing training with prior knowledge. In practice, if one knows that the target f_t was trained on a non-adversarial dataset, one can bias the adversarial examples toward performing well on a classifier f_s pretrained on a similar dataset $\mathcal{D}_{train}$, leading to the following game:

$$\max_{g \in \mathcal{G}_\epsilon} \min_{f_c \in \mathcal{F}} \mathbb{E}_{(x,y)\sim\mathcal{D}, z\sim p_z}[\ell(f_c(g(x, y, z)), y)] + \lambda \ell(f_s(g(x, y, z)), y)] =: \varphi_\lambda(f, g). \tag{3}$$

Note that f_s is *fixed* and that $\lambda = 0$ recovers (AEG). Such modifications in the maximin objective as well as setting the way f_c is trained (e.g., optimizer, regularization, additional dataset) biases the training of the representative classifier and corresponds to an implicit incorporation of prior knowledge on the target f_t in the hypothesis class $\mathcal{F}$.

4 Theoretical results

We note $\mathcal{D}$ to be the dataset used in (AEG) and p_g the joint adversarial distribution on (x, y) induced by g defined in (2). The representative classifier is the *score function* $f_c : \mathcal{X} \to \mathbb{R}^{|\mathcal{Y}|}$ with which we make predictions by selecting the label with the highest score. If we assume that we can generate any adversarial distribution respecting the proximity constraint (2), we have the following proposition.

Proposition 1. *If ℓ is the cross entropy loss, one has access to any measurable g respecting the proximity constraint in (2), and the class $\mathcal{F}$ of classifier is convex, then we can switch min and max in (AEG), i.e.,*

$$\min_{f_c \in \mathcal{F}} \max_{g \in \mathcal{G}_\epsilon} \varphi_\lambda(f_c, g) = \max_{g \in \mathcal{G}_\epsilon} \min_{f_c \in \mathcal{F}} \varphi_\lambda(f_c, g) \tag{4}$$

The assumption that we have access to any measurable g is standard in the literature [31, Prop. 2] and has often been referred to as "if g has enough capacity". Such a minimax result implies the existence of a Nash equilibrium for the game, which is a well defined target for learning. Moreover, a Nash equilibrium is a stationary point for gradient descent-ascent dynamics; we can thus hope for achieving such a solution by using a gradient-based learning algorithm on (AEG). Prop. 1 applies in two main cases of interest: (i) infinite capacity, i.e., when $\mathcal{F}$ is any measurable function. (ii) linear classifiers with *fixed* features $\psi : \mathcal{X} \to \mathbb{R}^p$, i.e., $\mathcal{F} = \{w^\top \psi(\cdot)\,,\ w \in \mathbb{R}^{|\mathcal{Y}| \times p}\}$. This second setting is particularly useful to build intuitions on the properties of (AEG), as we will see in §4.1 and Fig. 1.

4.1 A simple setup: binary classification with logistic regression

Let us now consider a binary classification setup where $\mathcal{Y} = \{\pm 1\}$ and $\mathcal{F}$ is the class of linear classifiers with linear features, i.e $f_w(x) = w^\top x$. In this case, the payoff of the game (AEG) is,

$$\varphi(f_\omega, g) := \mathbb{E}_{(x,y)\sim\mathcal{D},\, z\sim p_z}[\log(1 + e^{-y\cdot w^\top g(x,y,z)})] \tag{5}$$

This example is similar to the one presented in [31]. However, our purpose is different since we focus on characterizing the optimal generator of the minimax game (4). We show that the optimal generator can attack any classifier in $\mathcal{F}$ by shifting the means of the two classes of the dataset $\mathcal{D}$.

Proposition 2. *If the generator is allowed to generate any ℓ_∞ perturbations. The optimal representative classifier is the solution of the following ℓ_1 regularized logistic regression*

$$w^* \in \arg\min_w \mathbb{E}_{(x,y)\sim\mathcal{D}}[\log(1 + e^{-y\cdot w^\top x + \epsilon\|w\|_1})]\,. \tag{6}$$

Moreover if ω^* has no zero entry, the optimal generator is $g^*(x,y) = x - y\cdot\epsilon\,\mathrm{sign}(w^*)$, is deterministic *and the pair (f_{w^*}, g^*) is a Nash equilibrium of the game (5).*

A surprising fact is that, unlike in the general setting of Proposition 1, the generator is deterministic (i.e., does not depend on a latent variable z). This follows from the simple structure of classifiers in this class, which allow for a closed form solution for g^*. In general, one cannot expect to achieve an equilibrium with a deterministic generator. As previously noted by Goodfellow et al. [31], ℓ_∞ adversarial perturbation leads to a ℓ_1 regularization.[5] With this example, we want to illustrate how the optimal generator can attack an entire class of functions with limited capacity: linear classifiers are mostly sensitive to the mean of the distribution of each class; the optimal generator leverages this fact by moving these means closer to the decision boundary.

4.2 General multi-class classification

In this section, we show that, for a given hypothesis class $\mathcal{F}$, the generated distribution achieving the global maximin against $f_c \in \mathcal{F}$ can be interpreted as the distribution with the highest $\mathcal{F}$-entropy. For a given distribution p_g, its $\mathcal{F}$-entropy is the minimum expected risk under p_g one can achieve in $\mathcal{F}$.

Definition 2. *For a given distribution $(x,y) \sim p_g$ we define the $\mathcal{F}$-entropy of p_g as*

$$H_\mathcal{F}(p_g) := \min_{f_c\in\mathcal{F}} \mathbb{E}_{(x,y)\sim p_g}[\ell(f_c(x), y)] \qquad \textit{where } \ell \textit{ is the cross entropy loss.} \tag{7}$$

Thus $\mathcal{F}$-entropy quantifies the amount of "classification information" available in p_g using the class of classifiers $\mathcal{F}$. If the $\mathcal{F}$-entropy is large, $(x,y) \sim p_g$ cannot be easily classified with a function f_c in $\mathcal{F}$. Moreover, it is an upper-bound on the *expected conditional entropy* of the distribution p_g.

Proposition 3. *The $\mathcal{F}$-entropy is a decreasing function of $\mathcal{F}$, i.e., for any $\mathcal{F}_1 \subset \mathcal{F}_2$,*

$$H_{\mathcal{F}_1}(p_g) \geq H_{\mathcal{F}_2}(p_g) \geq H_y(p_g) := \mathbb{E}_{x\sim p_x}[H(p_g(\cdot|x))]\,.$$

where $H(p(\cdot|x)) := \sum_{y\in\mathcal{Y}} p(y|x)\ln p(y|x)$ is the entropy of the conditional distribution $p(y|x)$.

For a given class $\mathcal{F}$, the solution to an (AEG) game can be seen as one which finds an adversarial distribution of maximal $\mathcal{F}$-entropy regularized towards attacking the particular model f_s,

$$\max_{g\in\mathcal{G}_\epsilon}\min_{f_c\in\mathcal{F}} \varphi_\lambda(f_c, g) = \max_{g\in\mathcal{G}_\epsilon} H_\mathcal{F}(p_g) + \lambda\mathbb{E}_{p_g}[\ell(f_s(x), y)]\,. \tag{8}$$

This alternative perspective on the game (AEG) shares similarities with the divergence minimization perspective on GANs [37]: in both cases, the inner optimization with respect to the class of classifiers implicitly defines a meaningful quantity to optimize for the generator. However, while in GANs it represents a divergence between two distributions, in (AEG) this corresponds to a notion of entropy.

A high-level interpretation of $\mathcal{F}$-entropy maximization is that it implicitly defines a metric for distributions which are challenging to classify with only access to classifiers in $\mathcal{F}$. Overall, the optimal generated distribution p_g can be seen as the most adversarial dataset against the class $\mathcal{F}$.

Properties of the $\mathcal{F}$-entropy. We illustrate the idea that the optimal generator and the $\mathcal{F}$-entropy

[5] One can actually generalize Proposition 2 to a perturbation with respect to a general norm $\|\cdot\|$, in that case, the ϵ-regularization for the classifier would be with respect to the dual norm $\|\cdot\|_* := \max_{\|u\|\leq 1}\langle\cdot, u\rangle$.

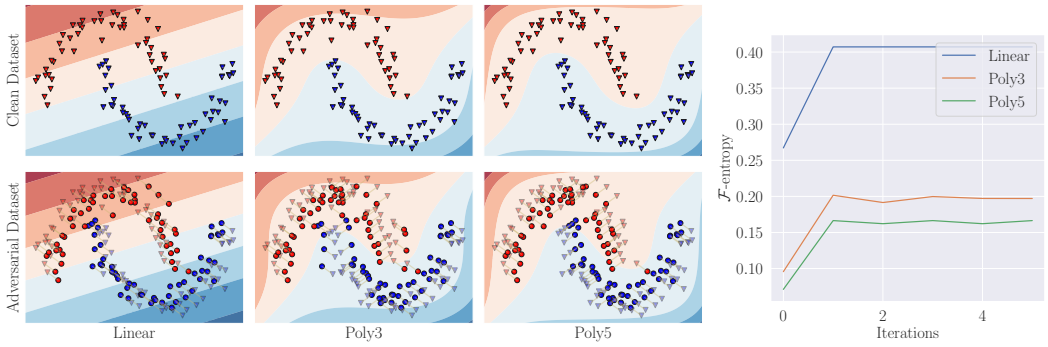

Figure 1: Illustration of Proposition 3 for three classes of classifiers in the context of logistic regression for the two moon dataset of scikit-learn [59] with linear and polynomial (of degree 3 and 5) features. **Left:** Scatter plot of the clean or adversarial dataset and the associated optimal decision boundary. For the adversarial dataset, each corresponding clean example is represented with a ▲/▲ and is connected to its respective adversarial example ●/●. **Right:** value of the $\mathcal{F}$-entropy for the different classes as a function of the number of iterations.

depend on the hypothesis class $\mathcal{F}$ using a simple example. To do so, we perform logistic regression (5) with linear and polynomial (of degree 3 and 5) features (respectively called Linear, Poly3, and Poly5) on the two moon dataset of scikit-learn [59]. Note that we have Linear $\subset$ Poly3 $\subset$ Poly5. For simplicity, we consider a deterministic generator $g(x, y)$ that is realized by computing the maximization step via 2D grid-search on the ϵ neighborhood of x. We train our models by successively fully solving the minimization step and the maximization step in (5).

We present the results in Figure 1. One iteration corresponds to the computation of the optimal classifier against the current adversarial distribution p_g (also giving the value of the $\mathcal{F}$-entropy), followed by the computation of the new optimal adversarial p'_g against this new classifier. The left plot illustrates the fact that the way of attacking a dataset depends on the class considered. For instance, when considering linear classifiers, the attack is a uniform translation on all the data-points of the same class. While when considering polynomial features, the optimal adversarial dataset pushes the the corners of the two moons closer together. In the right plot, we can see an illustration of Proposition 3, where the $\mathcal{F}$-entropy takes on a smaller value for larger classes of classifiers.

5 Attacking in the Wild: Experiments and Results

We investigate the application of our AEG framework to produce adversarial examples against classifiers trained on the MNIST and CIFAR-10 datasets.[6] Through our experiments we seek to answer the following questions:

- (**Q1**) **Known-Architecture Attacks.** Can we effectively generate attacks when we only know the architecture (but not the parameters) of the target model? In this case, the architecture specifies the hypothesis class of the target model, but have no access to the target model itself.
- (**Q2**) **Architecture Transfer Attacks.** Can we generate effective attacks even when the representative classifier (f_c) that we use during optimization implements a distinct architecture from the target model (f_t)? In this experiment, we do not even know the specific architecture of the target model and must implicitly attack a much larger hypothesis class.
- (**Q3**) **Attacking Robust Classifiers.** Can we successfully attack robustified classifiers [53, 67]? Under standard *PGD adversarial training* [53] this corresponds to a worst-case setting for our attacker. While for *ensemble adversarial training* [67] this involves injecting adversarial examples from other architectures, thus expanding the hypothesis class of the target model.

5.1 Experimental Setup

We perform all attacks, including baselines, on test datasets $\mathcal{D}$ of size 256 and with respect to the ℓ_∞ norm constraint with $\epsilon = 0.3$ for MNIST and $\epsilon = 0.03125$ for CIFAR-10.

Model details. We train both the generator (g) and representative classifier (f_c) using stochastic gradient descent-ascent with the ExtraAdam optimizer [27] and held out target models, f_t, are trained

[6] Code: https://github.com/hugobb/NoBoxAttack

offline using SGD with Armijo line search [69]. Full details of our model architectures employed in our AEG framework, as well as details on hyperparameter tuning, can be found in Appendix §C. In order to generate adversarial perturbations over images that obey ϵ-ball constraints, we employ a scaled tanh output layer to scale the output of the generator to $(0, 1)$, subtract the clean images, and finally apply an elementwise multiplication by ϵ. Note that—in addition to training against f_c—in practice we also find it beneficial to include a pre-trained and fixed classifier f_s as a mechanism to incorporate prior knowledge during AEG optimization. Intuitively, f_s plays an analogous role as the representative classifier f_c but is pre-trained on $\mathcal{D}_{\text{train}}$ and remains fixed during optimization, providing additional gradient signal and a source of stability during the optimization process.

Baselines. Throughout our experiments we rely on four representative blackbox attack strategies adapted to the NoBox setting: the Momentum-Iterative Attack (MI-Attack) [22], the Input Diversity (DI-Attack)[73], the Translation-Invariant (TID-Attack) [23] and the Skip Gradient Method (SGM-Attack) [71]. For fair comparison, we inherit all hyperparameter settings from their respective papers and note that certain attacks are architecture dependent such as the SGM-attack, which is only permitted in architectures that contain skip connections.

5.2 Results

We now address the core experimental questions (**Q1**-**Q3**), highlighting the performance of adversarial examples crafted using our AEG framework.

Q1: Known-Architecture Attacks. We evaluate the ability of AEG adversarial examples to attack different models from a given hypothesis class, specified by a neural network architecture. Specifically, we train 20 LeNet [47] classifiers on MNIST [46] and 10 classifiers using the Wide ResNet architecture [75] for CIFAR-10 [44]. To evaluate the baselines, we randomly select one trained classier as the source model that the baselines attack while the remaining classifiers are taken to be held out target models. Our results in this setting are summarized in Table 1. As can be observed, adversarial examples found using AEG exhibit the highest transferability to other MNIST classifiers. On CIFAR-10 we find that AEG adversarial examples are competitive with the baselines (i.e., within 1.4%), though some baselines achieve near perfect transfer in this setting. In addition to NoBox baselines, we also compare our approach with strong whitebox and blackbox query baselines adapted to the NoBox setting [17, 18, 50, 2, 68]. These results and samples of adversarial examples generated by g are provided in Appendix B.2.

Dataset	MI-Attack	DI-Attack	TID-Attack	SGM-Attack	AEG (Ours)
MNIST	93.7 ± 1.1	**95.9 ± 1.6**	92.8 ± 2.7	N/A	**96.1± 2.3**
CIFAR-10	**99.9 ±0.1**	**99.9 ± 0.1**	19.7 ± 1.5	**99.8 ± 0.3**	98.5 ± 0.7

Table 1: Error rates on $\mathcal{D}$ for NoBox known architecture attacks with $\epsilon = .3$ and $\epsilon = .03125$

Q2: Architecture Transfer Attacks. We now consider NoBox attacks where we do not know the architecture of the target model. For evaluation, we use CIFAR-10 and train 10 instances of VGG-16 [62], ResNet-18 (RN-18) [35], Wide ResNet (WR) [75], DenseNet-121 (DN-121) [36] and Inception-V3 architectures (Inc-V3) [66]. Here, we optimize the attack approaches against a single pre-trained classifier from a particular architecture and then evaluate their attack success on classifiers from distinct architectures. Our findings when using ResNet-18, DenseNet-121 and the Wide ResNet as the source architecture are provided in Table 2. When attacks are performed with a DN-121 source model, we observe that our AEG framework consistently outperforms the baselines. We hypothesize that as DN-121 can represent a relatively large hypothesis class, this leads to a particularly effective generator in the AEG framework. When using RN-18, we find that AEG is on par with and sometimes exceeds the state-of-the-art baselines. Finally, when using WR, we again obtain competitive results, except when the target model is VGG-16, which we believe is driven by the large mismatch in the underlying hypothesis classes for the WR and VGG-16 architectures.

Q3: Attacking Robust Classifiers. We now test the ability of our AEG framework to attack target models that have been robustified using adversarial and ensemble adversarial training [53, 67]. For evaluation against PGD adversarial training, we use the public models as part of the MNIST and

Source	Attack	VGG-16	RN-18	WR	DN-121	Inc-V3
	Clean	11.2 ± 0.9	13.1 ± 2.0	6.8 ± 0.7	11.2 ± 1.4	9.9 ± 1.3
RN-18	MI-Attack	71.2 ± 1.3	83.6 ± 0.3	66.6 ± 1.1	74.5 ± 1.4	76.8 ± 1.4
	DI-Attack	$\mathbf{72.6 \pm 1.3}$	$\mathbf{87.1 \pm 0.5}$	$\mathbf{73.0 \pm 1.3}$	$\mathbf{83.4 \pm 1.0}$	$\mathbf{79.8 \pm 1.3}$
	TID-Attack	22.6 ± 2.0	26.6 ± 1.9	13.5 ± 1.4	22.1 ± 2.1	19.2 ± 1.4
	SGM-Attack	68.4 ± 1.8	79.5 ± 0.5	64.3 ± 1.6	73.8 ± 1.0	70.6 ± 1.7
	AEG (Ours)	$\mathbf{71.4 \pm 3.0}$	$\mathbf{88.9 \pm 0.6}$	$\mathbf{70.0 \pm 2.5}$	$\mathbf{84.5 \pm 2.5}$	$\mathbf{81.8 \pm 1.7}$
DN-121	MI-Attack	54.7 ± 1.4	62.8 ± 1.1	55.5 ± 1.3	66.1 ± 1.5	64.3 ± 0.7
	DI-Attack	61.1 ± 1.9	69.1 ± 0.8	64.1 ± 1.5	77.1 ± 1.2	71.6 ± 1.6
	TID-Attack	20.1 ± 1.9	22.6 ± 2.5	13.3 ± 1.4	20.9 ± 1.6	21.6 ± 2.2
	SGM-Attack	51.6 ± 0.7	60.2 ± 1.3	52.6 ± 0.9	64.7 ± 1.6	61.4 ± 1.3
	AEG (Ours)	$\mathbf{69.0 \pm 3.8}$	$\mathbf{85.6 \pm 1.5}$	$\mathbf{78.6 \pm 2.1}$	$\mathbf{89.4 \pm 1.5}$	$\mathbf{84.9 \pm 1.9}$
WR	MI-Attack	71.2 ± 2.2	86.0 ± 1.7	$\mathbf{99.9 \pm 0.1}$	89.0 ± 2.6	88.2 ± 1.4
	DI-Attack	$\mathbf{75.4 \pm 1.5}$	$\mathbf{88.5 \pm 2.1}$	$\mathbf{99.9 \pm 0.1}$	$\mathbf{91.2 \pm 1.6}$	$\mathbf{91.5 \pm 1.8}$
	TID-Attack	21.9 ± 2.2	25.2 ± 1.7	19.7 ± 1.5	21.9 ± 1.7	21.9 ± 1.9
	SGM-Attack	69.1 ± 2.1	$\mathbf{88.6 \pm 2.0}$	$\mathbf{99.6 \pm 0.4}$	$\mathbf{90.7 \pm 1.9}$	86.8 ± 2.2
	AEG (Ours)	40.8 ± 3.22	70.6 ± 4.9	98.5 ± 0.6	$\mathbf{88.2 \pm 4.6}$	$\mathbf{89.6 \pm 1.8}$

Table 2: Error rates on $\mathcal{D}$ for average NoBox architecture transfer attacks with $\epsilon = 0.03125$

CIFAR-10 adversarial examples challenge. [7] For ensemble adversarial training, we follow the approach of Tramèr et al. [67] (see Appendix C.4). We report our results in Table 3 and average the result of stochastic attacks over 5 runs. We find that the AEG adversarial examples are able to succesfully models robustified through ensemble adversarial training. Specifically, AEG is competitive with most baselines but falls short of the powerful MI-Attack on the MNIST dataset. On CIFAR-10, we find that DI-Attack and TID-Attack to be the strongest and that AEG attacks are less effective. This is unsurprising, due to the inherent difficulties in saddle point optimization [27, 5] and since ensemble adversarial training involves injecting adversarial examples from other architectures, extending the underlying hypothesis class beyond which the generator was optimized on. Thus the chosen representative classifier classifier is suboptimal compared to representational power of the target model which constitutes an ensemble of architectures. Against PGD adversarial training, we observe that AEG adversarial examples matches SOTA for MNIST while exceeds it on CIFAR-10. We reconcile this result by recalling the saddle point of the AEG framework corresponds to attacking the strongest robustified classifier which includes classifiers robustified via PGD adversarial training.

Dataset	Defence	Clean	MI-Att†	DI-Att	TID-Att	SGM-Att †	AEG (Ours)
MNIST	A_{ens4}	0.8	41.4	$\mathbf{42.7 \pm 0.8}$	16.0 ± 1.0	N/A	29.2 ± 1.4
	B_{ens4}	0.7	**42.2**	38.7 ± 0.7	25.8 ± 0.4	N/A	40.2 ± 0.9
	C_{ens4}	0.8	**73.0**	30.0 ± 1.5	9.5 ± 0.3	N/A	37.1 ± 0.0
	D_{ens4}	1.8	**84.4**	76.0 ± 1.4	81.3 ± 0.8	N/A	67.9 ± 1.6
	Madry-Adv	0.8	2.0	$\mathbf{3.1 \pm 0.2}$	2.5 ± 0.2	N/A	$\mathbf{3.0 \pm 0.2}$
CIFAR-10	$RN\text{-}18_{ens3}$	16.8	17.6	21.6 ± 0.8	$\mathbf{33.1 \pm 1.4}$	19.9	19.5 ± 0.0
	WR_{ens3}	12.8	18.4	20.6 ± 1.0	$\mathbf{28.8 \pm 0.8}$	18.0	15.2 ± 0.0
	$DN\text{-}121_{ens3}$	21.5	20.3	22.7 ± 1.1	$\mathbf{31.3 \pm 0.3}$	21.9	19.9 ± 0.0
	$Inc\text{-}V3_{ens3}$	14.8	19.5	$\mathbf{42.2 \pm 1.3}$*	30.2 ± 0.6*	35.5*	23.0 ± 0.0
	Madry-Adv	12.9	17.2	16.6 ± 0.2	16.6 ± 0.5	16.0	$\mathbf{18.7 \pm 0.0}$

Table 3: Error rates on $\mathcal{D}$ for NoBox known architecture attacks against Adversarial Training and Ensemble Adversarial Training. * Attacks were done using WR. † Deterministic attack.

6 Related Work

In addition to non-interactive blackbox adversaries, we compare against there exists multiple hybrid approach's that combine crafting attacks on surrogate models which then serve as a good initialization point for queries to the target model [56, 61, 38]. Other notable approaches to craft blackbox transfer attacks learning ghost networks [48] and transforming whitebox gradients with small ResNets [49]. There is also a burgeoning literature of using parametric models to craft adversarial attacks such as

[7] `https://github.com/MadryLab/[dataset]_challenge`, for `[dataset]` in {`cifar10`, `mnist`}. Note that our threat model is more challenging than these challenges as we use non-robust source models.

the Adversarial Transformation Networks framework and its variants [4, 72]. Similar in spirit to our approach many attacks strategies benefit from employing a latent space to craft attacks [76, 68, 9]. However, unlike our work, these strategies cannot be used to attack entire hypothesis classes.

7 Conclusion

In this paper, we introduce the Adversarial Example Games (AEG) framework which provides a principled foundation for crafting adversarial attacks in the NoBox threat model. Our work sheds light on the existence of adversarial examples as a natural consequence of restricted entropy maximization under a hypothesis class and leads to an actionable strategy for attacking all functions taken from this class. Empirically we observe a high attack success rate in the known architecture variant of the NoBox setting, improving over prior methods on MNIST and competitive performance on CIFAR-10. For architecture transfer attacks, our approach is on par or exceeds all other baselines when the representative classifier is taken to be either a ResNet-18 or DenseNet-121 and is competitive when using Wide-ResNet. Finally, we find that AEG beats prior approaches in attacking PGD adversarially trained models on both MNIST and CIFAR, while for ensemble adversarially trained models AEG is competitive with existing baselines despite a large mismatch in hypothesis classes due to ensembling. While promising, our empirical analysis also highlights the inherent challenges in saddle point optimization [5] for generating theoretically optimal attacks.

Broader Impact

Adversarial attacks, especially ones under more realistic threat models, pose several important security, ethical, and privacy risks. In this work, we introduce the NoBox attack setting, which generalizes many other blackbox transfer settings, and we provide a novel framework to ground and study attacks theoretically and their transferability to other functions within a class of functions. As the NoBox threat model represents a more realistic setting for adversarial attacks, our research has the potential to be used against a class of machine learning models in the wild. In particular, in terms of risk, malicious actors could use approaches based on our framework to generate attack vectors that compromise production ML systems or potentially bias them toward specific outcomes that may adversely affect certain population groups, such as minority groups. As a concrete example, one can consider creating transferrable examples in the physical world, such as the computer vision systems of autonomous cars. While prior works have shown the possibility of such adversarial examples —i.e., adversarial traffic signs, we note that there is a significant gap in translating synthetic adversarial examples to adversarial examples that reside in the physical world [45]. Understanding and analyzing the NoBox transferability of adversarial examples to the physical world—in order to provide public and academic visibility on these risks—is an essential direction for future research.

Based on the known risks of designing new kinds of adversarial attacks—discussed above—we now outline the ways in which our research is informed by the intent to mitigate these potential societal risks. For instance, our research demonstrates that one can successfully craft adversarial attacks even in the challenging NoBox setting. It raises many important considerations when developing robustness approaches. A straightforward extension is to consider our adversarial example game (AEG) framework as a tool for training robust models. On the theoretical side, exploring formal verification of neural networks against NoBox adversaries is an exciting direction for continued exploration. As an application, ML practitioners in the industry may choose to employ new forms of A/B testing with different types of adversarial examples, of which AEG is one method to robustify and stress test production systems further. Such an application falls in line with other general approaches to red teaming AI systems [10] and verifiability in AI development. In essence, the goal of such approaches, including adversarial examples for robustness, is to align AI systems' failure modes to those found in human decision making. Another natural direction is the auditing of production ML systems by third parties, such as government or enforcement agencies.

Our work also has the potential to provide positive benefits in the realm of privacy, especially when concerned with the digital privacy of images. To combat this, Bose and Aarabi [8] showed typical face detectors, and by extension, facial recognition systems are vulnerable to adversarial attacks. Motivated by this use case, adversarial examples, including those presented in this work, can serve as a privacy-preserving mechanism by adding small amounts of crafted noise to safeguard against the unwanted loss of privacy to automatic facial recognition.

Finally—as mentioned above—some studies show that the use of automatic facial recognition has a disproportionately negative impact on ethnic minorities, such as persons of color [60]. Understanding the origin of model failures on adversarial examples pushes us towards the comprehension of such failure modes. Thus, it may also help explain why models disproportionately fail on people from ethnic minorities.

References

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

[2] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein. Square attack: a query-efficient black-box adversarial attack via random search. *arXiv preprint arXiv:1912.00049*, 2019.

[3] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok. Synthesizing robust adversarial examples. In *ICML*, 2018.

[4] S. Baluja and I. Fischer. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv preprint arXiv:1703.09387*, 2017.

[5] H. Berard, G. Gidel, A. Almahairi, P. Vincent, and S. Lacoste-Julien. A closer look at the optimization landscapes of generative adversarial networks. In *ICLR*, 2020.

[6] S. Bhambri, S. Muku, A. Tulasi, and A. Balaji Buduru. A survey of black-box adversarial attacks on computer vision models. *arXiv*, pages arXiv–1912, 2019.

[7] P. Billingsley. *Convergence of probability measures*. John Wiley & Sons, 1999.

[8] A. J. Bose and P. Aarabi. Adversarial attacks on face detectors using neural net based constrained optimization. In *2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6. IEEE, 2018.

[9] A. J. Bose, A. Cianflone, and W. Hamiltion. Generalizable adversarial attacks using generative models. *arXiv preprint arXiv:1905.10864*, 2019.

[10] M. Brundage, S. Avin, J. Wang, H. Belfield, G. Krueger, G. Hadfield, H. Khlaaf, J. Yang, H. Toner, R. Fong, et al. Toward trustworthy ai development: Mechanisms for supporting verifiable claims. *arXiv preprint arXiv:2004.07213*, 2020.

[11] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. 2018.

[12] N. Carlini and D. Wagner. Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples. *arXiv preprint arXiv:1711.08478*, 2017.

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

[14] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

[15] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

[16] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26. ACM, 2017.

[17] F. Croce and M. Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. *arXiv preprint arXiv:1907.02044*, 2019.

[18] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. *arXiv preprint arXiv:2003.01690*, 2020.

[19] G. S. Dhillon, K. Azizzadenesheli, Z. C. Lipton, J. Bernstein, J. Kossaifi, A. Khanna, and A. Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

[20] G. W. Ding, L. Wang, and X. Jin. AdverTorch v0.1: An adversarial robustness toolbox based on pytorch. *arXiv preprint arXiv:1902.07623*, 2019.

[21] G. W. Ding, Y. Sharma, K. Y. C. Lui, and R. Huang. MMA training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HkeryxBtPB.

[22] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[23] Y. Dong, T. Pang, H. Su, and J. Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4312–4321, 2019.

[24] J. Du, H. Zhang, J. T. Zhou, Y. Yang, and J. Feng. Query-efficient meta attack to deep neural networks. *arXiv preprint arXiv:1906.02398*, 2019.

[25] A. Erraqabi, A. Baratin, Y. Bengio, and S. Lacoste-Julien. A3t: Adversarially augmented adversarial training. *arXiv preprint arXiv:1801.04055*, 2018.

[26] K. Fan. Minimax theorems. *Proceedings of the National Academy of Sciences of the United States of America*, 1953.

[27] G. Gidel, H. Berard, G. Vignoud, P. Vincent, and S. Lacoste-Julien. A variational inequality perspective on generative adversarial networks. In *ICLR*, 2019.

[28] G. Gidel, D. Balduzzi, W. M. Czarnecki, M. Garnelo, and Y. Bachrach. Minimax theorem for latent games or: How i learned to stop worrying about mixed-nash and love neural nets. *arXiv preprint arXiv:2002.05820*, 2020.

[29] Z. Gong, W. Wang, and W.-S. Ku. Adversarial and clean data are not twins. *arXiv preprint arXiv:1704.04960*, 2017.

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[31] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[32] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

[33] C. Guo, M. Rana, M. Cisse, and L. Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[34] C. Guo, J. R. Gardner, Y. You, A. G. Wilson, and K. Q. Weinberger. Simple black-box adversarial attacks. In *ICML*, 2019.

[35] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[36] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*, 2016.

[37] G. Huang, H. Berard, A. Touati, G. Gidel, P. Vincent, and S. Lacoste-Julien. Parametric adversarial divergences are good task losses for generative modeling. *arXiv preprint arXiv:1708.02511*, 2017.

[38] Z. Huang and T. Zhang. Black-box adversarial attack with transferable model-based embedding. *arXiv preprint arXiv:1911.07140*, 2019.

[39] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin. Query-efficient black-box adversarial examples. *arXiv preprint arXiv:1712.07113*, 2017.

[40] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin. Black-box adversarial attacks with limited queries and information. *arXiv preprint arXiv:1804.08598*, 2018.

[41] A. Ilyas, L. Engstrom, and A. Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. *arXiv preprint arXiv:1807.07978*, 2018.

[42] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[43] L. Jiang, X. Ma, S. Chen, J. Bailey, and Y.-G. Jiang. Black-box adversarial attacks on video recognition models. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 864–872, 2019.

[44] A. Krizhevsky, V. Nair, and G. Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 2014.

[45] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[46] Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

[47] Y. LeCun et al. Lenet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*, 2015.

[48] Y. Li, S. Bai, Y. Zhou, C. Xie, Z. Zhang, and A. Yuille. Learning transferable adversarial examples via ghost networks. *arXiv preprint arXiv:1812.03413*, 2018.

[49] Y. Li, S. Bai, C. Xie, Z. Liao, X. Shen, and A. L. Yuille. Regional homogeneity: Towards learning transferable universal adversarial perturbations against defenses. *arXiv preprint arXiv:1904.00979*, 2019.

[50] Y. Li, L. Li, L. Wang, T. Zhang, and B. Gong. Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. *arXiv preprint arXiv:1905.00441*, 2019.

[51] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

[52] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

[53] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[54] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

[55] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[56] N. Papernot, P. McDaniel, and I. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[57] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 506–519. ACM, 2017.

[58] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[59] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011.

[60] I. D. Raji, T. Gebru, M. Mitchell, J. Buolamwini, J. Lee, and E. Denton. Saving face: Investigating the ethical concerns of facial recognition auditing. *arXiv preprint arXiv:2001.00964*, 2020.

[61] Y. Shi, S. Wang, and Y. Han. Curls & whey: Boosting black-box adversarial attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6519–6527, 2019.

[62] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[63] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

[64] L. Sun, M. Tan, and Z. Zhou. A survey of practical adversarial example attacks. *Cybersecurity*, 1(1):9, 2018.

[65] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[66] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[67] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[68] C.-C. Tu, P. Ting, P.-Y. Chen, S. Liu, H. Zhang, J. Yi, C.-J. Hsieh, and S.-M. Cheng. Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

[69] S. Vaswani, A. Mishkin, I. Laradji, M. Schmidt, G. Gidel, and S. Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. In *Advances in Neural Information Processing Systems*, pages 3727–3740, 2019.

[70] A. Wald. Statistical decision functions which minimize the maximum risk. *Annals of Mathematics*, 1945.

[71] D. Wu, Y. Wang, S.-T. Xia, J. Bailey, and X. Ma. Skip connections matter: On the transferability of adversarial examples generated with resnets. *arXiv preprint arXiv:2002.05990*, 2020.

[72] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

[73] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2730–2739, 2019.

[74] H. Xu, Y. Ma, H. Liu, D. Deb, H. Liu, J. Tang, and A. Jain. Adversarial attacks and defenses in images, graphs and text: A review. *arXiv preprint arXiv:1909.08072*, 2019.

[75] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[76] Z. Zhao, D. Dua, and S. Singh. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*, 2017.

Adversarial Example Games Supplementary Materials

A Proofs for Section 4 (Theoretical results)

Proposition 1. *If ℓ is the cross entropy loss, one has access to any measurable g respecting the proximity constraint in (2), and the class $\mathcal{F}$ of classifier is convex, then we can switch min and max in (AEG), i.e.,*

$$\min_{f_c \in \mathcal{F}} \max_{g \in \mathcal{G}_\epsilon} \varphi_\lambda(f_c, g) = \max_{g \in \mathcal{G}_\epsilon} \min_{f_c \in \mathcal{F}} \varphi_\lambda(f_c, g) \tag{4}$$

Proof. In this proof we have $\ell(p, y) := -\sum_{i=1}^{K} y_i \ln(p_i)$ where K is the number of classes one-hot encoded in y. However, what we only need in our proof is that the loss ℓ is convex.

Let us recall that the payoff φ is defined as

$$\max_{g \in \mathcal{G}_\epsilon} \min_{f \in \mathcal{F}} \mathbb{E}_{(x,y)\sim p_{data}, z\sim p_z}[\ell(f(g(x, y, z)), y)] =: \varphi(f, g) \tag{9}$$

Let us consider the case where $\mathcal{X}$ is finite as a warm-up. In practice, this can be the case if we consider that for instance one only allow a finite number of values for the pixels, e.g. (integers between 0 and 255 for CIFAR-10). In that case we have that

$$\mathbb{P}_{adv}(x, y) = \sum_{x' \in X} \mathbb{P}_{data}(x', y)\mathbb{P}_g(x|x', y) \quad \text{where} \quad \|x - x'\| > \epsilon \Rightarrow \mathbb{P}_g(x|x', y) = 0\,. \tag{10}$$

Assuming that one can achieve any $\mathbb{P}_g(\cdot|x', y)$ respecting the proximity constraint, the set $\{\mathbb{P}_{adv}\}$ is convex and compact. It is compact because closed and bounded in finite dimension and convex because of the linear dependence in $\mathbb{P}_g$ in (10) (and the fact that if $\mathbb{P}_{g_1}$ and $\mathbb{P}_{g_2}$ respect the constraints then $\lambda\mathbb{P}_{g_1} + (1 - \lambda)\mathbb{P}_{g_2}$ does it too).

For the non finite input case, we can consider that the generator G is a random variable defined on the probability space $(\mathcal{X} \times \mathcal{Y} \times \mathcal{Z}, \mathcal{B}, \mathbb{P}_{(x,y)} \times \mathbb{P}_z := \mathbb{P})$ where $\mathcal{B}$ is the Borel σ-algebra, $\mathbb{P}_{(x,y)}$ the probability on the space of data and $\mathbb{P}_z$ the probability on the latent space.

Then the adversarial distributions we consider is the set of pushforward distributions $\mathbb{P} \circ G^{-1}$ with G such that

$$G(x, y, z) = (g(x, y, z), y) \quad \text{and} \quad d(g(x, y, z), x) \leq \epsilon\,. \tag{11}$$

Assuming that g can achieve any random variable that satisfies (11), the minimax problem (4) can be rewritten as

$$\min_{f} \max_{p_g \in \Delta_\epsilon(\mathcal{X}\times\mathcal{Y})} \varphi(f, g) = \max_{p_g \in \Delta_\epsilon(\mathcal{X}\times\mathcal{Y})} \min_{f} \varphi(f, g) \tag{12}$$

where $\Delta_\epsilon(\mathcal{X} \times \mathcal{Y}) := \{\mathbb{P} \circ G^{-1} \,|\, G \text{ measurable satisfying (11)}\}$. This set is convex: let us consider $\mathbb{P} \circ G_1^{-1}$ and $\mathbb{P} \circ G_2^{-1}$ we have that

$$\lambda\mathbb{P} \circ G_1^{-1} + (1 - \lambda)\mathbb{P} \circ G_2^{-1} = \mathbb{P} \circ G_3^{-1} \tag{13}$$

where $g_3(x, y, z) = \delta(z)g_1(x, y, z) + (1 - \delta(z))g_2(x, y, z)$ and where $\delta \sim Ber(\lambda)$.

By using Skorokhod's representation theorem [7] we can show that this set is closed and thus compact (as closed subsets of compact sets are compact) using the weak convergence of measures as topology. Thus it is a convex compact Haussdorf space and in both cases ($\mathcal{X}$ finite and infinite) and we can apply Fan's Theorem.

Theorem 1. *[26, Theorem 2] Let U be a compact and convex Hausdorff space and V an arbitrary convex set. Let φ be a real valued function on $U \times V$ such that for every $v \in V$ the function $\varphi(\cdot, v)$ is lower semi-continuous on U. If φ is convex-concave then,*

$$\min_{u \in U} \sup_{v \in V} \varphi(u, v) = \sup_{v \in V} \min_{u \in U} \varphi(u, v) \tag{14}$$

□

Proposition 2. *If the generator is allowed to generate any ℓ_∞ perturbations. The optimal representative classifier is the solution of the following ℓ_1 regularized logistic regression*

$$w^* \in \arg\min_w \mathbb{E}_{(x,y)\sim\mathcal{D}}[\log(1 + e^{-y\cdot w^\top x + \epsilon\|w\|_1})]\,. \tag{6}$$

Moreover if ω^* has no zero entry, the optimal generator is $g^*(x,y) = x - y\cdot\epsilon\operatorname{sign}(w^*)$, is deterministic *and the pair (f_{w^*}, g^*) is a Nash equilibrium of the game (5).*

Proof. We prove the result here for any given norm $\|\cdot\|$. Let us consider the loss for a given pair (x,y)

$$\log(1 + e^{y(w^\top g(x,y)+b)}) \tag{15}$$

then by the fact that $x \mapsto \log(1+e^x)$ is increasing, maximizing this term for $\|g(x,y) - x\|_\infty \leq \epsilon$, boils down to solve the following maximization step,

$$\max_{\delta\,,\,\|\delta\|\leq\epsilon} y(w^\top\delta) = \|w\|_* \tag{16}$$

Particularly, for the ℓ_∞ norm we get

$$\arg\max_{\delta\,,\,\|\delta\|_\infty\leq\epsilon} y(w^\top\delta) = \epsilon y\operatorname{sign}(w)\,. \tag{17}$$

and

$$\max_g \mathbb{E}_{(x,y)\sim p_{data}}[\log(1 + e^{y(w^\top g(x,y)+b)})] = \mathbb{E}_{(x,y)\sim p_{data}}[\log(1 + e^{y(w^\top x+b)+\epsilon\|w\|_1})] \tag{18}$$

To show that (f^*, g^*) is a Nash equilibrium of the game (5), we first notice that by construction

$$\varphi(f^*, g^*) = \min_{f\in\mathcal{F}}\max_g \varphi(f,g) \tag{19}$$

where $\mathcal{F}$ is the class of classifier with linear logits. We then just need to notice that for all $f \in \mathcal{F}$ we have,

$$\varphi(f, g^*) = \mathbb{E}_{(x,y)\sim p_{data}}[\log(1 + e^{-y(w^\top x+b)+\epsilon w^\top \operatorname{sign}(w^*)})] \tag{20}$$

That is a convex problem in (w,b). Since, in a neighborhood of w^* we have that $w^\top\operatorname{sign}(w^*) = \|w\big|_{\operatorname{supp}(w^*)}\|_1$. Thus by assuming that w^* is full support and since w^* minimize (6) we have that

$$\nabla_w\varphi(w^*, b^*, g^*) = \nabla_w\mathbb{E}_{(x,y)\sim p_{data}}[\log(1 + e^{-y(w^\top x+b)+\epsilon\|w\|_1})] = 0 \tag{21}$$

Finally, by convexity of the problem (20) we can conclude that w^* is a minimizer of $\varphi(\cdot, g^*)$. To sum-up we have that

$$\varphi(f^*, g) \leq \varphi(f^*, g^*) \leq \varphi(f, g^*) \tag{22}$$

meaning that (f^*, g^*) is a Nash equilibrium of (5). □

Proposition 3. *The $\mathcal{F}$-entropy is a decreasing function of $\mathcal{F}$, i.e., for any $\mathcal{F}_1 \subset \mathcal{F}_2$,*

$$H_{\mathcal{F}_1}(p_g) \geq H_{\mathcal{F}_2}(p_g) \geq H_y(p_g) := \mathbb{E}_{x\sim p_x}[H(p_g(\cdot|x))]\,.$$

where $H(p(\cdot|x)) := \sum_{y\in\mathcal{Y}} p(y|x)\ln p(y|x)$ is the entropy of the conditional distribution $p(y|x)$.

Proof. In the case where $f^*(x) := p(y|x) \in \mathcal{F}$, since f^* a minimizer of the expected cross entropy loss over the class of any function, we have that

$$H_y(p) := \min_{f\in\mathcal{X}^\mathcal{Y}} \mathbb{E}_{(x,y)\sim p}[\ell(f(x), y)] = \mathbb{E}_x[H(p(\cdot|x))] \tag{23}$$

Lemma 1. *Given a data distribution $(x,y) \sim p_{adv}$ a minimizer of the cross entropy loss is*

$$p_{adv}(y|\cdot) \in \arg\min_f \mathbb{E}_{(x,y)\sim p_{adv}}[\ell(f(x), y)]\,. \tag{24}$$

Proof. Let us start by noticing that,

$$\min_{f} \mathbb{E}_{(x,y)\sim p}[\ell(f(x), y)] = \mathbb{E}_{x\sim p_x} \min_{q=f(x)} \mathbb{E}_{y\sim p(\cdot|x)}[\ell(q, y)] \tag{25}$$

Using the fact that ℓ is the cross-entropy loss we get

$$\mathbb{E}_{y\sim p(\cdot|x)}[\ell(q, y)] = \mathbb{E}_{y\sim p(\cdot|x)}[-\sum_{i=1}^{K} y_i \ln(q_i)] = -\sum_{i=1}^{K} p_i \ln(q_i) \tag{26}$$

where we noted $p_i = p(y = i|x)$. Noticing that since q_i is a probability distribution we have $\sum_{i=1}^{K} q_i = 1$, we have,

$$\mathbb{E}_{y\sim p(\cdot|x)}[\ell(q, y)] = -\sum_{i=1}^{K-1} p_i \ln(q_i) - p_K \ln(1 - \sum_{i=1}^{K-1} q_i) \tag{27}$$

we can then differentiate this loss with respect to $q_i \geq 0$ and get,

$$\frac{\partial \mathbb{E}_{y\sim p(\cdot|x)}[\ell(q, y)]}{\partial q_i}(q) = -\frac{p_i}{q_i} + \frac{p_K}{q_K} \tag{28}$$

We can finally notice that $q_i = p_i$ is a feasible solution. □

□

B Additional results

B.1 Quantitative Results

We now provide additional results in the form of whitebox and blackbox query attacks adapted to the NoBox evaluation protocol for Known-Architecture attacks which is the experimental setting in **Q1**. For whitebox attacks we evaluate APGD-CE and APGD-DLR [18] which are improvements over the powerful PGD attack [53]. When ensembled with another powerful perturbation minimizing whitebox attack FAB [17] and the query efficient blackbox Square attacks [2] yields the current SOTA attack strategy called AutoAttack [18] [18]. Additionally, we compare with two parametric blackbox query approaches that both utilize a latent space in AutoZoom [68] and $\mathcal{N}$Attack [50]. To test transferability of whitebox and blackbox query attacks in the NoBox known architecture setting we give generous iteration and query budgets (10x the reported settings in the original papers) when attacking the source models, but only a single query for each target model. It is interesting to note that APGD variant whitebox attacks are significantly more effective than query based blackbox attacks but lack the same effectiveness of NoBox baselines. We hypothesize that the transferability of whitebox attacks may be due to the fact that different functions learn similar decision boundaries but different enough such that minimum distortion whitebox attacks such as FAB are ineffective.

		MNIST	CIFAR-10
Whitebox	AutoAttack*	84.4 ± 5.1	91.0 ± 1.9
	APGD-CE	95.8 ± 1.9	97.5 ± 0.7
	APGD-DLR	83.9 ± 5.4	90.7 ± 2.1
	FAB	5.4 ± 2.2	10.4 ± 1.7
Blackbox-query	Square	60.9 ± 10.3	21.9 ± 2.8
	$\mathcal{N}$-Attack	9.5 ± 3.2	56.7 ± 8.9
Non-Interactive Blackbox	MI-Attack	93.7 ± 1.1	**99.9** ± 0.1
	DI-Attack	95.9 ± 1.6	**99.9** $\pm$ **0.1**
	TID-Attack	92.8 ± 2.7	19.7 ± 1.5
	SGM-Attack	N/A	**99.8** ± 0.3
	AEG (Ours)	**98.9** $\pm$ **1.4**	98.5 ± 0.6

Table 4: Test error rates for average blackbox transfer over architectures at $\epsilon = 0.3$ (higher is better)

B.2 Qualitative Results

As a sanity check we also provide some qualitative results about the generated adversarial attacks. In Figure 2 we show the 256 attacked samples generated by our method on MNIST. In Figure 3 we show on the left the 256 CIFAR samples to attack, and on the right the perturbations generated by our method amplified by a factor 10.

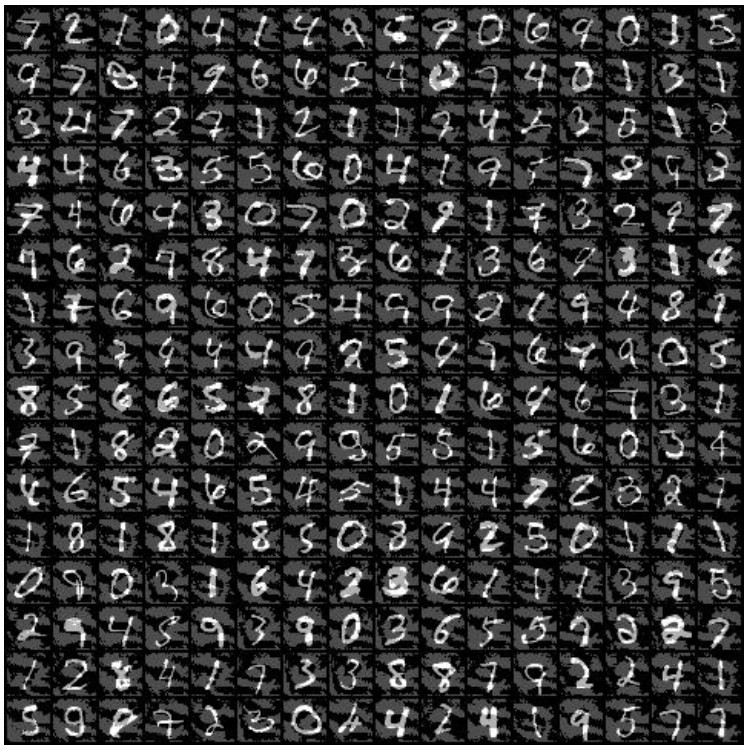

Figure 2: Attacks generated on MNIST by our method.

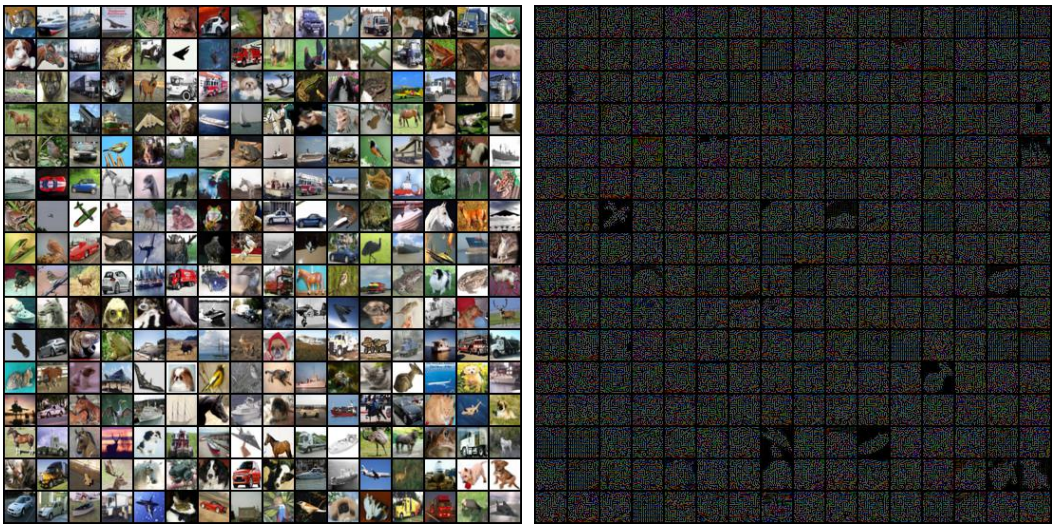

Figure 3: **Left:** CIFAR examples to attack. **Right:** Pertubations generated by our method amplified by a factor 10. An interesting observation is that the generator learns not to attack the pixel where the background is white.

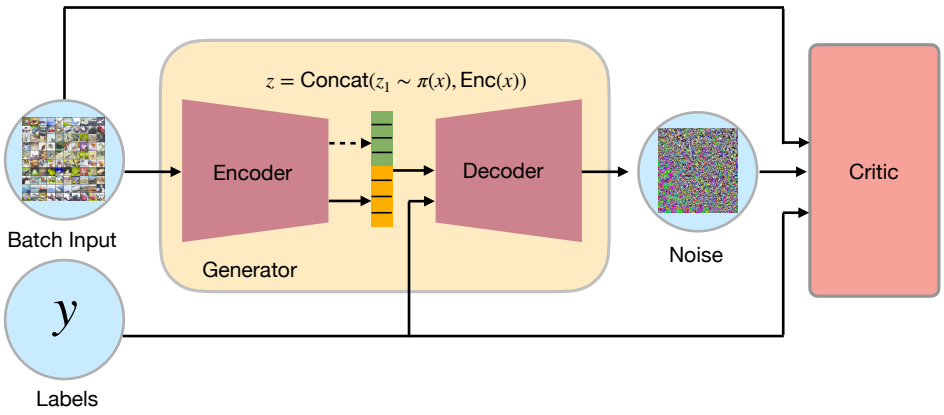

Figure 4: AEG framework architecture

d-**ResBlock**
Input:x
Forward for computing $F(x)$:
Reflection pad (1)
conv. (ker: 3×3, $d \to d$; stride: 1; pad: 1)
Batch Normalization
ReLU
Reflection pad (1)
conv. (ker: 3×3, $d \to d$; stride: 1; pad: 1)
Batch Normalization
Output:$x + F(x)$

Table 5: ResNet blocks used for the ResNet architectures (see Table 6) for the Generator. Each ResNet block contains skip connection (bypass), and a sequence of convolutional layers, normalization, and the ReLU non–linearity.

C Implementation Details

C.1 AEG Architecture

The high-level architecture of our AEG framework is illustraded in Figure 4. The generator takes the input x and encode it into $\psi(x)$, then the generator uses this encoding to compute a probability vector $p(\psi(x))$ in the probability simplex of size K, the number of classes. Using this probability vector the network then samples a categorical variable z according to a multinomial distribution of parameter $p(\psi(x))$. Intuitively, this category may correspond to a target for the attack. Gradient, is backprogated across this categorical variable using the gamble-softmax trick [42, 52]. Finally, the decoder takes as input $\psi(x)$, z and the label y to output an adversarial perturbation δ such that $\|\delta\| \leq \epsilon$. In the case of the ℓ_∞ norm the constraint is insured by having a last non-linearity equals to $\epsilon \cdot \tanh$. We then compute $\ell(f(x+\delta), y)$ where f is the critic and ℓ the cross entropy loss. We solve the game using the ExtraAdam optimizer [27] with a learning rate of 10^{-3}. We allow the generator to update its parameters several times on the same batch of examples before updating the critic. In particular we update the generator until it is able to fool the critic or it reaches some fixed number of iterations. We set this max number of iterations to 20 in all our experiments.

C.2 Generator Architecture

The architecture we used for the encoder and the decoder is described in Table 6 and 7. For MNIST we used a standard convolutional architecture and for CIFAR-10 we used a ResNet architecture.

Encoder
Input: $x \in \mathbb{R}^{3\times32\times32}$
Reflection Padding (3)
conv. (ker: 7×7, $32 \to 63$; stride: 1; pad: 0)
Batch Normalization
ReLU
conv. (ker: 3×3, $63 \to 127$; stride: 2; pad: 0)
Batch Normalization
ReLU
conv. (ker: 3×3, $127 \to 255$; stride: 2; pad: 0)
Batch Normalization
ReLU
255-ResBlock
255-ResBlock
255-ResBlock

Decoder
Input: $(\psi(x), z, y) \in \mathbb{R}^{256\times8\times8}$
256-ResBlock
256-ResBlock
256-ResBlock
Transp. conv. (ker: 3×3, $256 \to 128$; stride: 2; pad: 0)
Batch Normalization
ReLU
Transp. conv. (ker: 3×3, $128 \to 64$; stride: 2; pad: 0)
Batch Normalization
ReLU
ReflectionPadding(3)
conv. (ker: 7×7, $64 \to 32$; stride: 1; pad: 0)
Tanh

Table 6: Encoder and Decoder for the convolutional generator used for the MNIST dataset.

Encoder
Input: $x \in \mathbb{R}^{28\times28}$
conv. (ker: 3×3, $1 \to 64$; stride: 3; pad: 1)
LeakyReLU(0.2)
Max Pooling (stride: 2)
conv. (ker: 3×3, $64 \to 32$; stride: 2; pad: 1)
LeakyReLU(0.2)
Max Pooling (stride: 2)

Decoder
Input: $(\psi(x), z, y) \in \mathbb{R}^{64\times2\times2}$
Transp. conv. (ker: 3×3, $64 \to 32$; stride: 2; pad: 1)
LeakyReLU(0.2)
Max Pooling stride 2
Reflection Padding (3)
Transp. conv. (ker: 5×5, $32 \to 16$; stride: 3; pad: 1)
LeakyReLU(0.2)
Max Pooling stride 2
Transp. conv. (ker: 2×2, $16 \to 1$; stride: 2; pad: 1)
Tanh

Table 7: Encoder and Decoder for the ResNet generator used for the MNIST dataset.

C.3 Baseline Implementation Details

The principal baselines used in the main paper include the Momentum-Iterative Attack (MI-Attack) [22], the Input Diversity (DI-Attack)[73], the Translation-Invariant (TID-Attack) [23] and the Skip Gradient Method (SGM-Attack) [71]. As Input Diversity and Translation invariant are approaches that generally can be combined with existing attack strategies we choose to use the powerful Momentum-Iterative attack as our base attack. Thus the DI-Attack consists of random input transformations when using an MI-Attack adversary while the TID-attack further adds a convolutional kernel ontop of the DI-Attack. We base our implementions using the AdverTorch [20] library and adapt all baselines to this framework using original implementations where available. In particular, when possible we reused open source code in the Pytorch library [58] otherwise we re-implement existing algorithms. We also inherit most hyperparameters settings when reporting baseline results except for number steps used in iterated attacks. We find that most iterated attacks benefit from additional optimization steps when attacking MNIST and CIFAR-10 classifiers. Specifically, we allot a 100 step budget for all iterated attacks which is often a five to ten fold increase than the reported setting in all baselines.

C.4 Ensemble Adversarial Training Architectures

We ensemble adversarially train our models in accordance with the training protocol outlined in [67]. For MNIST models we train a standard model for 6 epochs, and an ensemble adversarial model using adversarial examples from the remaining three architectures for 12 epochs. The specific architectures for Models A-D are provided in Table. 8. Similarly, for CIFAR-10 we train both the standard model and ensemble adversarial models for 50 epochs. For computationally efficiency we randomly sample two out of three held out architectures when ensemble adversarially training the source model.

A	B	C	D
Conv(64, 5, 5) + Relu	Dropout(0.2)	Conv(128, 3, 3) + Tanh	FC(300) + Relu
Conv(64, 5, 5) + Relu	Conv(64, 8, 8) + Relu	MaxPool(2,2)	Dropout(0.5)
Dropout(0.25)	Conv(128, 6, 6) + Relu	Conv(64, 3, 3) + Tanh	FC(300) + Relu
FC(128) + Relu	Conv(128, 6, 6) + Relu	MaxPool(2,2)	Dropout(0.5)
Dropout(0.5)	Dropout(0.5)	FC(128) + Relu	FC(300) + Relu
FC + Softmax	FC + Softmax	FC + Softmax	Dropout(0.5)
			FC(300) + Relu
			Dropout(0.5)
			FC + Softmax

Table 8: MNIST Ensemble Adversarial Training Architectures)

D Further Related Work

Adversarial attacks can be classified under different threat models, which impose different access and resource restrictions on the attacker [1]. The whitebox setting, where the attacker has full access to the model parameters and outputs, thus allowing the attacker to utilize gradients based methods to solve a constrained optimization procedure. This setting is more permissive than the semi-whtiebox and the blackbox setting, the latter of which the attacker has only access to the prediction [56, 57] or sometimes the predicted confidence [34]. In this paper, we focus on a challenging variant of the conventional blackbox threat model which we call the NoBox setting which further restricts the attacker by *not* allowing any query from the target model. While there exists a vast literature of adversarial attacks, we focus on ones that are most related to our setting and direct the interested reader to comprehensive surveys for adversarial attacks and blackbox adversarial attacks [6, 15].

Whitebox Attacks. The most common threat model for whitebox adversarial examples are l_p-norm attacks, where $p \in \{2, \infty\}$ is the choice of norm ball used to define the attack budget. One of the earliest gradient based attacks is the Fast Gradient Sign Method (FGSM) [31], which computes bounded perturbations in a single step by computing the signed gradient of the loss function with respect to a clean input. More powerful adversaries can be computed using multi-step attacks such as DeepFool [55] which iteratively finds the minimum distance over perturbation direction needed to cross a decision boundary. For constrained optimization problems the Carlini-Wagner (CW) attack [13] is a powerful iterative optimization scheme which introduces an attack objective designed to maximize the distance between the target class and the most likely adversarial class. Similarly, projected gradient descent based attacks has been shown to be the strongest class of adversaries for l_2 and l_∞ norm attacks [53] and even provides a natural way of robustifying models through adversarial training. Extensions of PGD that fix failures due to suboptimal step size and problems of the objective function include AutoPGD-CE (APGD-CE) and AutoPGD-DLR (APGD-DLR) and leads to the state of the art whitebox attack in AutoAttack [18] which ensembles two other strong diverse and parameter free attacks.

Blackbox Attacks. Like whitebox attacks the adversarial goal for a blackbox attacker remains the same with the most common threat model also being l_p norm attacks. Unlike, whitebox attacks the adversarial capabilities of the attacker is severely restricted rendering exact gradient computation impossible. In lieu of exact gradients, early blackbox attacks generated adversarial examples on surrogate models in combination with queries to the target model [56]. When given a query budget gradient estimation is an attractive approach with notable approaches utilizing black box optimization schemes such as Finite Differences [16], Natural Evolutionary Strategies [40, 43], learned priors in a bandit optimization framework [41], meta-learning attack patterns [24], and query efficient

Defenses. In order to protect against the security risk posed by adversarial examples there have been many proposed defense strategies. Here we provide a non-exhaustive list of such methods. Broadly speaking, most defense approaches can be categorized into either robust optimization techniques, gradient obfuscation methods, or adversarial example detection algorithms [74]. Robust optimization techniques aim to improve the robustness of a classifier by learning model parameters by incorporating adversarial examples from a given attack into the training process [53, 67, 21]. On the other hand obfuscation methods rely on masking the input gradient needed by an attacker to construct adversarial examples [63, 11, 33, 19].

In adversarial example detection schemes the defender seeks to sanitize the inputs to the target model by rejecting any it deems adversarial. Often this involves training auxiliary classifiers or differences in statistics between adversarial examples and clean data [32, 54, 29].

Finally, Erraqabi et al. [25] developed an adversarial game framework as a mean for building robust representations where an additional disciminator is trained to disciminate adversarial example from natural ones, based on the representation of the current classifier.