The LogBarrier adversarial attack: making effective use of decision boundary information

Chris Finlay, Aram-Alexandre Pooladian, and Adam Oberman
McGill University
{christopher.finlay, aram-alexandre.pooladian}@mail.mcgill.ca, adam.oberman@mcgill.ca

Abstract

Adversarial attacks for image classification are small perturbations to images that are designed to cause misclassification by a model. Adversarial attacks formally correspond to an optimization problem: find a minimum norm image perturbation, constrained to cause misclassification. A number of effective attacks have been developed. However, to date, no gradient-based attacks have used best practices from the optimization literature to solve this constrained minimization problem. We design a new untargeted attack, based on these best practices, using the well-regarded logarithmic barrier method.

On average, our attack distance is similar or better than all state-of-the-art attacks on benchmark datasets (MNIST, CIFAR10, ImageNet-1K). In addition, our method performs significantly better on the most challenging images, those which normally require larger perturbations for misclassification. We employ the LogBarrier attack on several adversarially defended models, and show that it adversarially perturbs all images more efficiently than other attacks: the distance needed to perturb all images is significantly smaller with the LogBarrier attack than with other state-of-the-art attacks.

1. Introduction

Deep learning models have achieved impressive results in many areas of application. However, deep learning models remain vulnerable to adversarial attacks [21]: small changes (imperceptible to the human eye) in the model input may lead to vastly different model predictions. In security-based applications, this vulnerability is of utmost concern. For example, traffic signs may be modified with small stickers to cause misclassification, causing say a stop sign to be treated as speed limit sign [7]. Facial recognition systems can be easily spoofed using colourful glasses [19].

This security flaw has led to an arms race in the research community, between those who develop defences to adversarial attacks, and those working to overcome these defences with stronger adversarial attack methods [22, 17]. Notably, as the community develops stronger adversarial attack methods, claims of model robustness to adversarial attack are often proved to be premature [5, 2].

There are two approaches to demonstrating a model is resistant to adversarial attacks. The first is *theoretical*, via a provable lower bound on the minimum adversarial distance necessary to cause misclassification [22, 17, 9]. Theoretical lower bounds are often pessimistic: the gap between the theoretical lower bound and adversarial examples generated by state-of-the-art attack algorithms can be large. Therefore, a second *empirical* approach is also used: an upper bound on the minimum adversarial distance is demonstrated through adversarial examples created by adversarial attacks [11, 14, 5, 4]. The motivation to design *strong* adversarial attacks is therefore twofold: on the one hand, to validate theoretical lower bounds on robustness; and on the other, to construct empirical upper bounds on the minimum adversarial distance. Ideally, the gap between the theoretical lower bound and the empirical upper bound should be small. As adversarial attacks become stronger, the gap narrows from above.

The process of finding an adversarial example with an adversarial attack is an optimization problem: find a small perturbation of the model input which causes misclassification. This optimization problem has been recast in various ways. Rather than directly enforcing misclassification, many adversarial attacks instead attempt to maximize the loss function. The Fast Gradient Signed Method (FGSM) was one of the first adversarial attacks to do so [21], measuring perturbation size in the ℓ_∞ norm. Iterative versions of FGSM were soon developed [11, 14, 24]. When perturbations are measured in the ℓ_2 norm, the iterative version performs Projected Gradient Descent (PGD); when measured in ℓ_∞ the iterative version is known as IFGSM. Both iterative versions maximize the loss function subject to a constraint enforcing small perturbation in the appropriate norm. Other works have studied sparse adversarial attacks, as in [16]. Rather than maximizing the loss, Carlini and

Wagner [5] developed a strong adversarial attack by forcing misclassification to a predetermined target class. If only the decision of the model is available (but not the loss or the model gradients), adversarial examples can still be found using gradient-free optimization techniques [4].

In this paper, rather than using the loss as a proxy for misclassification, we design an adversarial attack that solves the adversarial optimization problem *directly*: minimize the size of the input perturbation subject to a misclassification constraint. Our method is gradient-based, but does not use the training loss function. The method is based on a sound, well-developed optimization technique, namely the logarithmic barrier method [15]. The logarithmic barrier is a simple and intuitive method designed specifically to enforce inequality constraints, which we leverage to enforce misclassification. We compare the LogBarrier attack against current benchmark adversarial attacks (using the Foolbox attack library [18]), on several common datasets (MNIST [13], CIFAR10 [10], ImageNet-1K [6]) and models. On average, we show that the LogBarrier attack is comparable to current state-of-the-art adversarial attacks. Moreover, we show that on challenging images (those that require larger perturbations for misclassification), the LogBarrier attack consistently outperforms other adversarial attacks. Indeed, we illustrate this point by attacking models trained to be adversarially robust, and show that the LogBarrier attack perturbs all images more efficiently than other attack methods: the LogBarrier attack is able to perturb all images using a much smaller perturbation size than that of other methods.

2. Background material

Adversarial examples arise in classification problems across multiple domains. The literature to date has been concerned primarily with adversarial examples in image classification: adversarial images appears no different (or only slightly so) from an image correctly classified by a model, but despite this similarity, are misclassified.

We let $\mathcal{X}$ be the space of images. Typically, pixel values are scaled to be between 0 and 1, so that $\mathcal{X}$ is the unit box $[0,1]^M \subset \mathbb{R}^M$. We let $\mathcal{Y}$ be the space of labels. If the images can be one of N classes, $\mathcal{Y}$ is usually a subset of $\mathbb{R}^N$. Often $\mathcal{Y}$ is the probability simplex, but not always. In this case, each element y_i of a label y correspond to the probability an image is of class i. Ground-truth labels are then one-hot vectors.

A trained model, with fixed model weights w, is a map $f(\cdot; w) : \mathcal{X} \to \mathcal{Y}$. For brevity, in what follows we drop dependence on w. For an input image x, the model's predicted classification is the $\arg\max$ of the model outputs. Given image-label pair (x, y), let c be the index of the correct label (the $\arg\max$ of y). A model is correct if $\arg\max f(x) = c$.

An adversarial image is a perturbation of the original image, $x + \delta$, such that the model misclassifies:

$$\arg\max f(x+\delta) \neq c \tag{1}$$

The perturbation must be small in a certain sense: it must be small enough that a human can still correctly classify the perturbed image. There are various metrics for measuring the size of the perturbation. A common choice is the ℓ_∞ norm (the max-norm); others use the (Euclidean) ℓ_2 norm. If perturbations must be sparse – for example, if the attacker can only modify a small portion of the total image – then the count of non-zero elements in δ may be used. Throughout this paper we let $m(\delta)$ be a generic metric on the size of the perturbation δ. Typically $m(\delta)$ is a specific ℓ_p norm, $m(\delta) = \|\delta\|_p$, such as the Euclidean or max norms. Thus the problem of finding an adversarial image may be cast as an optimization problem, minimize the size of the perturbation subject to model misclassification.

The misclassification constraint is difficult to enforce, so a popular alternative is to introduce a loss function $\mathcal{L}$. For example, $\mathcal{L}$ could be the loss function used during model training. In this case the loss measures the 'correctness' of the model at an image x. If the loss is large at a perturbed image $x + \delta$, then it is hoped that the image is also misclassified. The loss function is then used as a proxy for misclassification, which gives rise to the following indirect method for finding adversarial examples:

$$\begin{aligned} \underset{\delta}{\text{maximize}} \quad & \mathcal{L}(x+\delta) \\ \text{subject to} \quad & m(\delta) \leq \varepsilon, \end{aligned} \tag{2}$$

maximize the loss subject to perturbations being smaller than a certain threshold. The optimization approach taken by (2) is by far the most popular method for finding adversarial examples. In one of the first papers on this topic, Szegedy et al [21] proposed the Fast Signed Gradient Method (FGSM), where $m(\delta)$ is the ℓ_∞ norm of the perturbation δ, and the solution to (2) is approximated by taking one step in the signed gradient direction. An iterative version with multiple steps, Iterative FGSM (IFGSM) was proposed in [11], and remains the method of choice for adversarial attacks measured in ℓ_∞. When perturbations are measured in ℓ_2, (2) is solved with Projected Gradient Descent (PGD) [14].

Fewer works have studied the adversarial optimization problem directly, i.e., without a loss function. In a seminal work, Carlini and Wagner [5], developed a *targeted attack*, in which the adversarial distance is minimized subject to a targeted misclassification. In a targeted attack, not just any misclassification will do: the adversarial perturbation must induce misclassification to a pre-specified target class. The Carlini-Wagner attack (CW) incorporates the targeted misclassification constraint as a penalty term into the objective function. The CW attack was able to overcome many

adversarial defence methods that had been thought to be effective, and was an impetus for the adversarial research community's search for rigorous, theoretical guarantees of adversarial robustness.

There is interest in gradient-free methods for finding adversarial examples. In this scenario, the attacker only has access to the classification of the model, but not the model itself (nor the model's gradients). In [4], Brendal et al directly minimize the ℓ_2 adversarial distance while enforcing misclassification using a gradient-free method. Their Boundary attack iteratively alternates between minimizing the perturbation size, and projecting the perturbation onto the classification boundary. The projection step is approximated by locally sampling the model decision near the classification boundary.

3. The LogBarrier Attack

We tackle the problem of finding (untargeted) adversarial examples by directly solving the following optimization problem,

$$\begin{aligned} &\underset{\delta}{\text{minimize}} \quad m(\delta) \\ &\text{subject to} \quad \arg\max f(x+\delta) \neq c, \end{aligned} \tag{3}$$

that is, minimize the adversarial distance subject to misclassification. We use the logarithmic barrier method [15] to enforce misclassification, as follows. We are given an image-label pair (x, y) and correct label $c = \arg\max y$. Misclassification at an image x occurs if there is at least one index of the model's prediction with greater value than the prediction of the correct index:

$$\max_{i \neq c} f_i(x) - f_c(x) > 0 \tag{4}$$

This is a necessary and sufficient condition for misclassification. Thus, we rewrite (3):

$$\begin{aligned} &\underset{\delta}{\text{minimize}} \quad m(\delta) \\ &\text{subject to} \quad \max_{i \neq c} f_i(x+\delta) - f_c(x+\delta) > 0. \end{aligned} \tag{5}$$

The barrier method is a standard tool in optimization for solving problems such as (5) with inequality constraints. A complete discussion of the method can be found in [15]. In the barrier method, inequality constraints are incorporated into the objective function via a penalty term, which is infinite if a constraint is violated. If a constraint is far from being active, then the penalty term should be small. The negative logarithm is an ideal choice:

$$\min_{\delta} m(\delta) - \lambda \log\left(f_{\max} - f_c\right) \tag{6}$$

where we denote $f_{\max} := \max_i f_i(x+\delta)$ and $f_c := f_c(x+\delta)$. If the gap between $f_{\max}$ is much larger than f_c, the logarithmic barrier term is small. However, as this gap shrinks, the penalty term approaches infinity. Thus the penalty acts as a barrier, forcing an optimization algorithm to search for solutions where the constraint is inactive. If (6) is solved iteratively with smaller and smaller values of λ, in the limit as $\lambda \to 0$, the solution to the original problem (5) is recovered. (This argument can be made formal if desired, using Γ-convergence [3].) See Figure 1, where the barrier function is plotted with decreasing values of λ. In the limit as $\lambda \to 0$, the barrier becomes 0 if the constraint is satisfied, and ∞ otherwise.

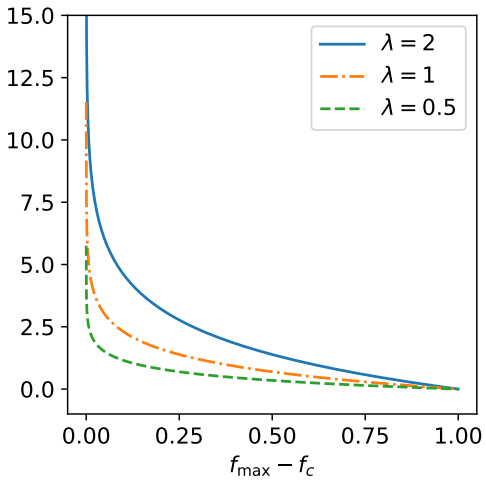


Figure 1: The logarithmic barrier function $\phi(\cdot) := -\lambda \log(\cdot)$ defined over $(0, 1)$. As λ decreases, the barrier becomes steeper, mimicking a hard constraint.

3.1. Algorithm description

We now give a precise description of our implementation of the log barrier method for generating adversarial images. The constraint $f_{\max} - f_c > 0$ can be viewed as a feasible set. Thus, the algorithm begins by finding an initial feasible image: the original image must be perturbed so that it is misclassified (not necessarily close to the original). There are several ways to find a misclassified image. A simple approach would be to take another natural image with a different label. However, we have found in practice that closer initial images are generated by randomly perturbing the original image with increasing levels of noise (*e.g.* Standard Normal or Bernoulli) until it is misclassified. After each random perturbation, the image is projected back onto the set of images in the $[0, 1]^M$ box, via the projection $\mathcal{P}$. This process is briefly described in Algorithm 1. Note that if the original image is already misclassified, no random perturbation is performed, since the original image is already adversarial.

Algorithm 1 LogBarrier: Initialization

Input: image $x \in \mathcal{X}$, model $f(\cdot; w)$, ρ, step-size $h > 0$, $k_{\max} \in \mathbb{N}$.
Initialize: $B \sim \text{Bernoulli}(\rho) \in \mathcal{X}$ or $B \sim \text{Normal}(0, 1)$
for $k = 0$ **to** $k_{\max}$ **do**
  if x misclassified **then**
    Exit **for-loop**
  else
    Sample b from B
    $x \leftarrow \mathcal{P}(x + h1.01^k b)$
  end if
end for

After an initial perturbation is found, we solve (6) for a fixed λ. Various optimization methods algorithms may be used to solve (6). For small- to medium-scale problems, variants of Newton's method are typically preferred. However, due to computational constraints, we chose to use gradient descent. After each gradient descent step, we check to ensure that the updated adversarial image remains in the $[0, 1]^M$ box. If not, it is projected back into the set of images with the projection $\mathcal{P}$.

It is possible that a gradient descent step moves the adversarial image so that the image is correctly classified by the model. If this occurs, we simply backtrack along the line between the current iterate and the previous iterate, until we regain feasibility. To illustrate the backtracking procedure, let $u^{(k)}$ be the previous iterate, and $\tilde{u}^{(k+1)}$ be a candidate adversarial image which is now correctly classified. We continue backtracking the next iterate via

$$\tilde{u}^{(k+1)} \leftarrow \gamma \tilde{u}^{(k+1)} + (1 - \gamma) u^{(k)}, \tag{7}$$

until the iterate is misclassified. The hyper-parameter $\gamma \in (0, 1)$ is a backtracking parameter. The accumulation point of the above sequence is $u^{(k)}$. As a result, this process is guaranteed to terminate, since the previous iterate is itself misclassified. This backtracking procedure is sometimes necessary when iterates are very close to the decision boundary. If the iterate is very close to the decision boundary, then the gradient of the log barrier term is very large, and dominates the update step. Since the constraint set $f_{\max} - f_y > 0$ is not necessarily convex or even fully connected, it is possible that the iterate could be sent far from the previous iterate without maintaining misclassification. We rarely experience this phenomenon in practice, but include the backtracking step as a safety. An alternate approach (which we did not implement), more aligned with traditional optimization techniques, would be instead to use a dynamic step size rule such as the Armijo-Goldstein condition [1].

The gradient descent algorithm comprises a series of iterates in an inner loop. Recall that as $\lambda \to 0$, the log barrier problem approaches the original problem (5). Thus, we shrink λ by a certain factor and repeat the procedure again, iterating in a series of outer loops (of course, initializing now with previous iterate). As λ shrinks, the solutions to (6) approach the decision boundary. In each inner loop, if the iterates fail to move less than some threshold value ε, we move onto the next outer loop. The path taken by the iterates of the outer loop is called the *central path*, illustrated in Figure 2.

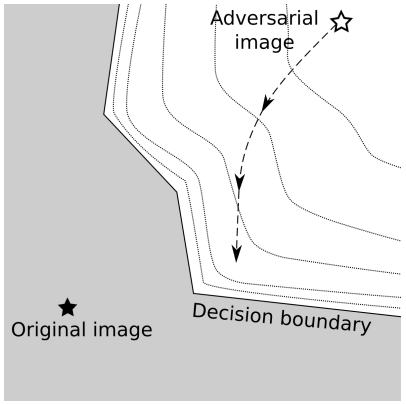


Figure 2: The central path taken by the LogBarrier attack. Dashed lines represent level sets of the logarithmic barrier function. As λ decreases iterates approach the decision boundary.

Algorithm 2 LogBarrier attack

Input: original image x, initial misclassified image $u^{(0)}$, model $f(\cdot; w)$, distance measure $m(\cdot)$
Hyperparameters: backtrack factor $\gamma \in (0, 1)$; initial penalty size λ_0; step size h; λ shrink factor $\beta \in (0, 1)$; termination threshold $\varepsilon > 0$; and maximum iterations $K_{\text{outer}}, J_{\text{inner}} \in \mathbb{N}$.
for $j = 0$ **to** $K_{\text{outer}} \cdot J_{\text{inner}}$ **do**
  If $j \mod K_{\text{outer}} = 0 : \lambda_k = \lambda_0 \beta^k$

$$\tilde{u}^{(j+1)} \leftarrow u^{(j)} - h\nabla \left(m(u^{(j)} - x) + \lambda_k \phi(u^{(j)}) \right)$$
$$u^{(j+1)} \leftarrow \mathcal{P}\left(\tilde{u}^{(j+1)} \right) \quad \text{project onto } [0, 1]^M$$

  while $u^{(j+1)}$ **not** misclassified **do**

$$u^{(j+1)} \leftarrow \gamma u^{(j+1)} + (1 - \gamma) u^{(j)}$$

  end while
  if $\|u^{(j+1)} - u^{(j)}\| \leq \varepsilon$ **then**
    break
  end if
end for

The LogBarrier attack pseudocode is presented in Algorithm 2. For brevity we write the log barrier $\phi(u) := -\log(\max_i f_i(u) - f_c(u))$. We remark that the LogBarrier attack can be improved by running the method several times, with different random initializations (although we do not implement this here).

The literature on adversarial perturbations primarily focuses on perturbations measured in the ℓ_2 and ℓ_∞ norms. For perturbations measured in the ℓ_2 norm, we set the distance measure to be the squared Euclidean norm, $m(\delta) = \|\delta\|_2^2$. When perturbations are measured in the ℓ_∞ norm, we do not use the max-norm directly as a measure, due to the fact that the ℓ_∞ norm is non-smooth with sparse subgradients. Instead, we use the following approximation of the ℓ_∞ norm [12],

$$\begin{aligned}\|\delta\|_\infty &= \max_{i=1,\dots,N} |\delta_i| \\ &\approx \frac{\sum_{i=1}^N |\delta_i| \exp(\alpha|\delta_i|)}{\sum_{i=1}^N \exp(\alpha|\delta_i|)},\end{aligned}$$

where $\alpha > 0$. As $\alpha \to \infty$ the ℓ_∞ norm is recovered.

Algorithm hyper-parameters

Like many optimization routines, the logarithmic barrier method has several hyper-parameters. However, because our implementation is parallelized, we have found that the tuning process is relatively quick. For the ℓ_∞ attack, our default parameters are $\varepsilon = 10^{-6}, h = 0.1, \beta = 0.75, \gamma = 0.5, \lambda_0 = 0.1, K_{\text{outer}} = 25$, and $J_{\text{inner}} = 1000$. For ℓ_2, we set $h = 5 \cdot 10^{-3}$ with $K_{\text{outer}} = 15$ and $J_{\text{inner}} = 200$; the rest are the same as in the ℓ_∞ case.

For the initialization procedure, we have $k_{\max} = 10^3$ and $h = 5 \cdot 10^{-4}$. If attacking in ℓ_2, we initialize using the Standard Normal distribution. Else, for ℓ_∞, we use the Bernoulli initialization with $\rho = 0.01$.

Top5 misclassification

The LogBarrier attack may be generalized to enforce Top5 misclassification as well. In this case, the misclassification constraint is that $f_{(k)}(x+\delta) - f_c(x+\delta) > 0$, $k = 1, \dots, 5$, where now (k) is the index of sorted model outputs. (In other words, $f_{(1)} = \max_i f_i$, and $f_{(2)}$ is the second-largest model output, and so forth.) We then set the barrier function to be $-\sum_{k=1}^5 \log(f_{(k)} - f_c)$. In this scenario, the LogBarrier attack is initialized with an image that is not classified in the Top5.

4. Experimental results

We compare the LogBarrier attack with current state-of-the-art adversarial attacks on three benchmark datasets: MNIST [13], CIFAR10 [10], and ImageNet-1K [6]. On MNIST and CIFAR10, we attack 1000 randomly chosen images; on ImageNet-1K we attack 500 randomly selected images, due to computational constraints. On ImageNet-1K, we use the Top5 version of the LogBarrier attack.

Table 1: Percent misclassification of the networks at a specified perturbation size, for attacks measured in ℓ_2. Because we are measuring the strength of adversarial attacks, at a given adversarial distance, a higher percentage misclassified is better.

	MNIST	CIFAR10		Imagenet-1K
		AllCNN	ResNeXt34	
$\|\delta\|_2$	2.3	120/255	120/255	1
LogBarrier	**99.10**	**98.70**	**99.90**	**98.40**
CW	98.50	97.30	90.40	74.86
PGD	52.58	86.60	59.80	90.00
BA	97.20	**98.70**	99.60	48.80

Table 2: Percent misclassification of the networks at a specified perturbation size, for attacks measured in ℓ_∞. Higher percentage misclassified is better.

	MNIST	CIFAR10		Imagenet-1K
		AllCNN	ResNeXt34	
$\|\delta\|_\infty$	0.3	8/255	8/255	8/255
LogBarrier	**94.80**	**100**	**98.70**	95.20
IFGSM	73.40	93.1	75.80	**99.60**

All other attack methods are implemented using the adversarial attack library Foolbox [18]. For adversarial attacks measured in ℓ_2, we compare the LogBarrier attack against Projected Gradient Descent (PGD) [14], the Carlini-Wagner attack (CW) [5], and the Boundary attack (BA) [4]. These three attacks all very strong, and consistently perform well in adversarial attack competitions. When measured in ℓ_∞, we compare against IFGSM [11], the current state-of-the-art. We leave Foolbox hyper-parameters to their defaults, except the number of iterations in the Boundary attack, which we set to a maximum of 5000 iterations.

4.1. Undefended networks

We first study the LogBarrier attack on networks that have not been trained to be adversarially robust. For MNIST, we use the network described in [5, 16]. On CIFAR10, we consider two networks: AllCNN [20], a shallow convolutional network; and a ResNeXt34 (2x32) [23], a much deeper network residual network. Finally, for ImageNet-1K, we use a pre-trained ResNet50 [8] available for download on the PyTorch website.

Tables 1 and 2 report the percentage misclassified, for each attack at a fixed perturbation size. A strong attack should have a high misclassification rate. In the tables, the perturbation size is chosen to agree with attack thresholds commonly reported in the adversarial literature. Measured

Table 3: Adversarial attacks perturbation statistics in the ℓ_2 norm. We report the mean and variance of the adversarial distance on a subsample of the test dataset. Lower values are better.

	MNIST		CIFAR10 AllCNN		CIFAR10 ResNeXt34		ImageNet-1K	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
LogBarrier	1.29	1.98e−1	**1.63e−1**	**1.12e−2**	1.21e−1	**6.68e−3**	**3.82e−1**	**6.87e−2**
CW	**1.27**	**1.96e−1**	1.72e−1	8.57e−2	2.39e−1	1.87e−1	8.86e−1	1.59
PGD	2.54	2.53	3.18e−1	3.49e−1	6.88e−1	1.15	4.21e−1	3.16e−1
BA	1.41	2.11e−1	**1.63e−1**	1.36e−2	**1.11e−1**	7.396e−3	1.55	3.31

Table 4: Adversarial attacks perturbation statistics in the ℓ_∞ norm. We report the mean and variance of the adversarial attack distance for each method on a subsample of the test dataset. Lower values are better.

	MNIST		CIFAR10 AllCNN		CIFAR10 ResNeXt34		ImageNet-1K	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
LogBarrier	**1.57e−1**	**7.43e−3**	**6.16e−3**	**1.3e−5**	**5.14e−3**	**3.20e−5**	1.27e−2	1.46e−3
IFGSM	2.49e−1	3.4e−2	1.14e−2	6.93e−4	2.70e−2	2.07e−3	**2.38e−3**	**1.30e−5**

in Euclidean norm, we see that the LogBarrier attack is the strongest on all datasets and models. Measured in the max-norm, the LogBarrier outperforms IFGSM on all datasets and models, except on ImageNet-1K where the difference is slight.

We also report the mean and variance of the adversarial attack distances, measured in ℓ_2 and ℓ_∞, in Tables 3 and 4 respectively. A strong adversarial attack should have a small mean adversarial distance, and a small variance. Small variance is necessary to ensure precision of the attack method. A strong attack method should be able to consistently find close adversarial examples. Table 3 demonstrates that, measured in ℓ_2, the LogBarrier attack is either the first ranked attack, or a close second. When measured in ℓ_∞, the LogBarrier attack significantly outperforms IFGSM on all datasets and models, except ImageNet-1K.

For illustration, we show examples of adversarial images from the IFGSM and LogBarrier attacks in Figure 3. On images where IFGSM requires a large distance to adversarially perturb, the LogBarrier attack produces visibly less distorted images.

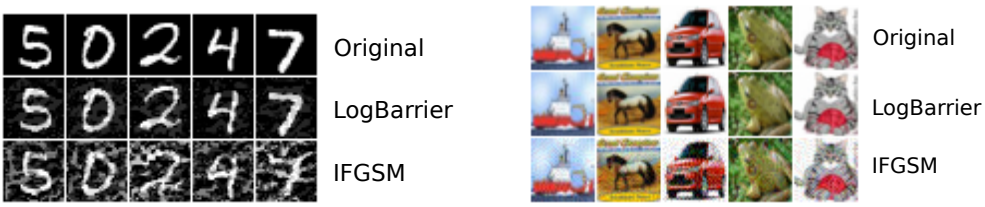
4.2. Defended networks

In this section we turn to attacking adversarially defended networks. We first consider two defence strategies: gradient obfuscation [2], and multi-step adversarial training as described in Madry et al [14]. We study these two strategies on the MNIST and ResNeXt34 networks used in Section 4.1. We limit ourselves to studying defence methods for attacks in the ℓ_∞ norm. Attacks are performed on the same 1000 randomly selected images as the previous section. Finally, we also test our attack on a MNIST model trained with Convex Adversarial Polytope [22] training, the current state-of-the-art defence method on MNIST.

Gradient Obfuscation

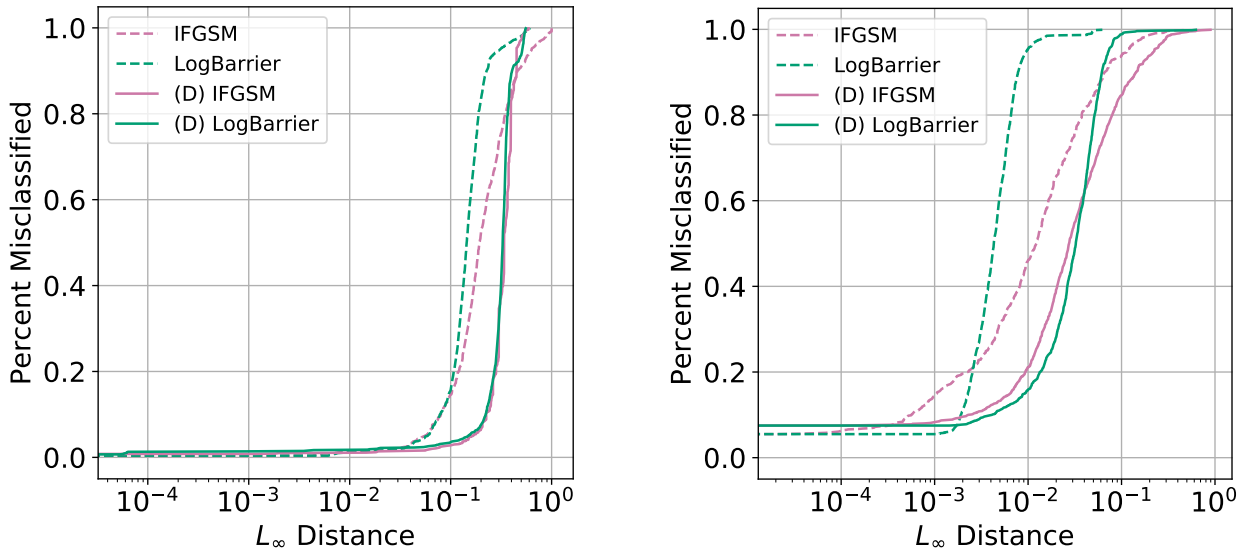
Although discredited as a defence method [2], gradient obfuscation is a hurdle any newly proposed adversarial attack method must be able to surmount. We implement gradient obfuscation by increasing the temperature on the softmax function computing model probabilities from model logits. As the softmax temperature increases, the size of the gradients of the model probabilities approaches zero, because the model probabilities approach one-hot vectors. Although the decision boundary of the model does not change, many adversarial attack algorithms have difficulty generating adversarial examples when model gradients are small.

In Tables 5 and 6 we show that the LogBarrier attack easily overcomes gradient obfuscation, on both CIFAR10 and MNIST models. The reason that the LogBarrier method is able to overcome gradient obfuscation is simple: away from the decision boundary, the logarithmic barrier term is not active (indeed, it is nearly zero). Thus the LogBarrier algorithm focuses on minimizing the adversarial distance, until it is very close to the decision boundary, at which point the barrier term activates. In contrast, because IFGSM is a local method, if model gradients are small, it has a difficult time climbing the loss landscape, and is not able to generate adversarial images.



(a) ℓ_∞ attacks on MNIST (b) ℓ_∞ attacks on CIFAR10

Figure 3: Adversarial images for ℓ_∞ perturbations, generated by the LogBarrier and IFGSM adversarial attacks, compared against the original clean image. Where IFGSM has difficulties finding adversarial images, the LogBarrier method succeeds: LogBarrier adversarial images are visibly less distorted than IFGSM adversarial images.



(a) MNIST (b) CIFAR10

Figure 4: Overlay of attack curves, measured in ℓ_∞, on (a) MNIST and (b) CIFAR10 networks. Two types of networks are compared: an undefended network, and a defended network (denoted (D)), trained using the same architecture as the undefended network with adversarial training. The LogBarrier attack requires a smaller adversarial distance to attack all images, compared to IFGSM.

Adversarial Training

Adversarial training is a popular method for defending against adversarial attacks. We test the LogBarrier attack on networks trained with multi-step adversarial training in the ℓ_∞ norm, as presented in Madry et al [14]. Our results are shown in Tables 5 and 6. We also plot defence curves of the LogBarrier and IFGSM attacks on defended and undefended models in Figures 4a and 4b, for respectively MNIST and CIFAR10.

On MNIST, we did not observe a reduction in test accuracy on clean images with adversarially trained models compared to undefended models. As expected, adversarial training hinders both LogBarrier and IFGSM from finding adversarial images at very small distances. However, we see that the LogBarrier attack is able to attack all images with nearly the same distance in both the defended and undefended models. In contrast, IFGSM requires a very large adversarial distance to attack all images on the defended model, as shown in Figure 4a. That is, adversarial training *does not significantly reduce the empirical distance required to perturb all images*, when the LogBarrier attack is used. The point is illustrated in Table 5, where we report the distance required to perturb 90% of all images. The LogBarrier attack requires an adversarial distance of 0.22 on the undefended MNIST model, and 0.29 on the defended MNIST model, to perturb 90% of all images. In contrast, IFGSM requires a distance of 0.46 on the undefended model, but 0.65 on the defended model.

On CIFAR10, we observe the same behaviour, although

Table 5: Defence strategies on MNIST. We report the percentage misclassified at ℓ_∞ adversarial magnitudes $\|\delta\|_\infty = 0.1$ and 0.3; higher is better. We also report the attack magnitude needed to perturb 90% of the images (the 90% quantile of attacks, written $q(90\%)$). 'NA' indicates that the attack failed.

	Undefended			Obfuscated ($T = 2$)			Adversarial training[1]		
	$\|\delta\|_\infty = 0.1$	$\|\delta\|_\infty = 0.3$	$q(90\%)$	$\|\delta\|_\infty = 0.1$	$\|\delta\|_\infty = 0.3$	$q(90\%)$	$\|\delta\|_\infty = 0.1$	$\|\delta\|_\infty = 0.3$	$q(90\%)$
LogBarrier	**15.70**	**94.80**	**2.27e−1**	**18.30**	**99.80**	**1.95e−1**	**3.59** (2.90)	**31.50** (95.40)	**4.06e−1** (2.85e−1)
IFGSM	12.40	62.5	4.60e−1	8.60	32.90	NA	2.80 (3.00)	23.59 (53.80)	4.49e−1 (6.51e−1)

Table 6: Defence strategies on ResNeXt34 on the CIFAR10 dataset. We report the percentage misclassified at ℓ_∞ adversarial magnitudes of $\|\delta\|_\infty = 4/255$ and $8/255$, and the magnitude required to perturb 90% of test images. If the adversarial attack was unsuccessful, we report NA.

	Undefended			Obfuscated ($T = 20$)			Adversarial training		
	$\|\delta\|_\infty = \frac{4}{255}$	$\|\delta\|_\infty = \frac{8}{255}$	$q(90\%)$	$\|\delta\|_\infty = \frac{4}{255}$	$\|\delta\|_\infty = \frac{8}{255}$	$q(90\%)$	$\|\delta\|_\infty = \frac{4}{255}$	$\|\delta\|_\infty = \frac{8}{255}$	$q(90\%)$
LogBarrier	**98.40**	**98.70**	**7.79e−3**	**47.60**	**54.40**	**1.53e−1**	23.40	48.10	**9.58e−2**
IFGSM	58.30	75.80	6.56e−2	36.90	43.90	NA	**31.60**	**54.90**	1.38e−1

the phenomenon is less pronounced. As shown in Table 6 and Figure 4b, the LogBarrier attack requires a smaller adversarial distance to perturb all images than IFGSM. Notably, the LogBarrier attack on the defended network is able to attack all images with a smaller adversarial distance than even IFGSM on the undefended network.

Against the Convex Adversarial Polytope

Finally, we use the LogBarrier attack on a provable defence strategy, the Convex Adversarial Polytope [22]. The Convex Adversarial Polytope is a method for training a model to guarantee that no more than a certain percentage of images may be attacked at a given adversarial distance. We chose to attack the defended MNIST network in [22], which is guaranteed to have no more than 5.82% misclassification at perturbation size $\|\delta\|_\infty = 0.1$. We validated this theoretical guarantee with both the LogBarrier attack and IFGSM, and found that both methods were unable to perturb more than 3% of test images at distance 0.1.

5. Discussion

We have presented a new adversarial attack that uses a traditional method from the optimization literature, namely the logarithmic barrier method. The LogBarrier attack is effective in both the ℓ_∞ and ℓ_2 norms. The LogBarrier attack *directly* solves the optimization problem posed by the very definition of adversarial images; i.e., find an image close to an original image, while being misclassified by a network. This is in contrast to many other adversarial attack problems (such as PGD or IFGSM), which attempt to maximize a loss function as a proxy to the true adversarial optimization problem. Whereas loss-based adversarial attacks start locally at or near the original image, the LogBarrier attack begins far from the original image. In this sense, the LogBarrier attack is similar in spirit to the Boundary attack [4]: both the LogBarrier attack and the Boundary attack begin with a misclassified image, and iteratively move the image closer to the original image, while maintaining misclassification. The LogBarrier attack is a gradient-based attack: to enforce misclassification, gradients of the logarithmic barrier are required. In contrast, the Boundary attack is gradient-free, and uses rejection sampling to enforce misclassification. Although the LogBarrier attack uses gradients, we have shown that it is not impeded by gradient obfuscation, a common drawback to other gradient-based attacks. Because the LogBarrier attack is able to use gradients, it is typically faster than the Boundary attack.

The LogBarrier attack may be used as an effective tool to validate claims of adversarial robustness. We have shown that one strength of the LogBarrier attack is its ability to attack all images in a test set, using a fairly small maximum adversarial distance compared to other attacks. In other words, the LogBarrier attack estimates the mean adversarial distance with high precision. Using the LogBarrier attack, we have raised questions about the robustness of multi-step adversarial training [14]. For instance, on MNIST, we showed that multi-step adversarial training did not significantly improve the necessary distance required to perturb all test images, relative to an undefended model. For adversarially trained models on CIFAR10, we showed that the necessary distance to perturb all images is significantly smaller than the estimate provided by IFGSM. This is further motivation for the development of rigorous, theoretical guarantees of model robustness.

[1] In an earlier manuscript, the defended MNIST model was not as robust as it could have been. Here we report attacks on a new robust model as well as the older less robust model; the results for the old model is in parenthesis.

References

[1] Larry Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16(1):1–3, 1966. 4

[2] Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 274–283, 2018. 1, 6

[3] Andrea Braides. Γ-*convergence for Beginners*. Oxford University Press, 2002. 3

[4] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations*, 2018. 1, 2, 3, 5, 8

[5] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 39–57, 2017. 1, 2, 5

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255, 2009. 2, 5

[7] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1625–1634, 2018. 1

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. 5

[9] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, pages 97–117, 2017. 1

[10] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 2, 5

[11] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016. 1, 2, 5

[12] Mandy Lange, Dietlind Zühlke, Olaf Holz, Thomas Villmann, and Saxonia-Germany Mittweida. Applications of lp-norms and their smooth approximations for gradient based learning vector quantization. In *ESANN*, 2014. 5

[13] Yann LeCun and Corinna Cortes. The MNIST database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*. 2, 5

[14] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *CoRR*, abs/1706.06083, 2017. 1, 2, 5, 6, 7, 8

[15] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006. 2, 3

[16] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, pages 372–387, 2016. 1, 5

[17] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *CoRR*, abs/1801.09344, 2018. 1

[18] Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox v0.8.0: A python toolbox to benchmark the robustness of machine learning models. *CoRR*, abs/1707.04131, 2017. 2, 5

[19] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 1528–1540, 2016. 1

[20] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, abs/1412.6806, 2014. 5

[21] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. 1, 2

[22] Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 5283–5292, 2018. 1, 6, 8

[23] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5987–5995, 2017. 5

[24] Tianhang Zheng, Changyou Chen, and Kui Ren. Distributionally adversarial attack. *CoRR*, abs/1808.05537, 2018. 1