Improved Image Wasserstein Attacks and Defenses

J. Edward Hu*
Microsoft Research AI
1 Microsoft Way
Redmond, WA 98052
edwardhu@microsoft.com

Hadi Salman
Microsoft Research AI
1 Microsoft Way
Redmond, WA 98052
Hadi.Salman@microsoft.com

Adith Swaminathan
Microsoft Research AI
1 Microsoft Way
Redmond, WA 98052
adswamin@microsoft.com

Greg Yang
Microsoft Research AI
1 Microsoft Way
Redmond, WA 98052
gregyang@microsoft.com

Abstract

Robustness against image perturbations bounded by a ℓ_p ball have been well-studied in recent literature. Perturbations in the real-world, however, rarely exhibit the pixel independence that ℓ_p threat models assume. A recently proposed Wasserstein distance-bounded threat model is a promising alternative that limits the perturbation to pixel mass movements. We point out and rectify flaws in previous definition of the Wasserstein threat model and explore stronger attacks and defenses under our better-defined framework. Lastly, we discuss the inability of current Wasserstein-robust models in defending against perturbations seen in the real world.

1 Introduction

Deep learning approaches to computer vision tasks, such as image classification, are not robust. For example, a data point that is classified correctly can be modified in a nearly imperceptible way to cause the classifier to mis-classify it (Szegedy et al., 2013; Goodfellow et al., 2015). Projected Gradient Descent (PGD) is a well-studied method to find such small perturbations within a ℓ_p ball of a small radius (Madry et al., 2017). While both general and effective, the ℓ_p threat model perturbs each pixel independently, a property not seen in realistic perturbations, such as distortion, blurring, and spatial shifts.

Previously, Wasserstein distance has been proposed as a more perceptually-aligned metric for images (Peleg et al., 1989). Recently, Wong et al. (2019) proposed a Wasserstein distance-based threat model as an alternative to the ℓ_p threat model, and derived a computationally feasible approach to project onto the Wasserstein ball during PGD. Fig. 1 (Left) shows a perturbation found when attacking a ℓ_p robust model with our Wasserstein threat model. The perturbation looks different than that from a ℓ_p threat model.

The Wasserstein threat model of Wong et al. (2019) provided a great foundation, but only considered *normalized* images; their attack algorithm, when applied to real images, could produce a perturbed image that is *outside* the allowed Wasserstein ball. In this work, we define the Wasserstein threat model such that it applies to all images, and we provide a *safe* algorithm to find adversarial perturbations within a specified Wasserstein radius. Our algorithm uses a *constrained* Sinkhorn iteration to project images onto the intersection of a Wasserstein ball and a ℓ_∞ ball; the computational overhead from our new constraint is offset by our run-time optimizations and justified by our stronger attacks. Further,

*Work done as a part of the Microsoft AI Residency.

Preprint. Under review.

MNIST	1-Wasserstein $\epsilon \times n_{pixel}$	5	10	20	50	100	200	500	1000
	Wong et al. (2019) (%)	100	100	100	100	94	91	83	71
	Our Attack (%)	100	90	85	68	39	8	1	1
CIFAR-10	1-Wasserstein $\epsilon \times n_{pixel}$	5	10	20	50	100	200	500	1000
	Wong et al. (2019) (%)	82	82	82	82	82	81	80	77
	Our Attack (%)	76	68	58	35	20	9	1	0

Table 1: Empirical top-1 accuracies of the adversarially trained Wasserstein-robust classifier from Wong et al. (2019) under their and our attack at various Wasserstein radii (ϵ) multiplied by the per image pixel count n_{pixel}. Results are for MNIST and CIFAR-10. Lower accuracy at a given ϵ suggests a stronger attack.

we provide a significantly stronger attack than Wong et al. (2019)'s by exploring different PGD steps, which we incorporate in an adversarial training framework to obtain more robust models.

The main contributions of our work include:

- A definition of the Wasserstein adversarial threat model for all images of the same dimensionality, fixing a glitch in the previous formulation (Sec. 3);
- A constrained Sinkhorn iteration projection algorithm that produces adversarial examples under our new, better-formulated threat model (Sec. 4);
- A significantly stronger Wasserstein-bounded attack by taking a different PGD step; our new attack breaks adversarially trained Wasserstein-robust models in prior works (Sec. 5);
- An adversarially trained model that achieves state-of-the-art robustness against the new attacks, while still being robust against the attacks in prior works (Sec. 6).

2 Adversarial Robustness for Images

Deep neural networks (DNNs) can be fooled into making wrong predictions by deliberately changing the input in ways imperceptible to humans. In this section we summarize common approaches for such attacks and defenses against them.

Adversarial Attacks One well-studied example is the *additive ℓ_p-bounded attack* (Madry et al., 2017; Goodfellow et al., 2015; Carlini & Wagner, 2017; Wong & Kolter, 2018), where a perturbation designed to increase the loss of the correct label, usually found using first-order information, is added to the input. PGD performs this addition and projection jointly and iteratively (Madry et al., 2017), which gives an efficient empirical algorithm for finding adversarial examples. The ℓ_p norm of total perturbation is bounded to a small ϵ so that the change is imperceptible. Another example is the *pixel-wise functional attack* (Laidlaw & Feizi, 2019), where a function is applied to each pixel individually, with constraints on the function itself and/or on the output pixel, enabling attacks in the color space, for instance. *Image-wide attacks* can also be used to fool DNNs. These are defined by a function that applies to the image as a whole, such as translations and rotations, with ℓ_p constraints on the function parameters (Mohapatra et al., 2019). In this work, we focus on a different class of attacks, specifically adversarial examples with a bounded Wasserstein distance (Wong et al., 2019) and whose ℓ_1 norm is the same as the original inputs.

Adversarial Defenses In order to defend a model against a given adversary, one can emply empirical or certified defenses. Empirical defenses are basically heuristics that are used to train robust models. One of the most effective empirical defenses is *adversarial training* (Goodfellow et al., 2015; Madry et al., 2017; Wong et al., 2019), in which a given predictive model is trained on adversarial examples generated by a given threat model. This defense, although empirical, has proven to be one of the strongest defenses in practice. Certified defenses on the other hand are those that provide guarantees under a specific threat model (Wong & Kolter, 2018; Cohen et al., 2019; Salman et al., 2019a; Levine & Feizi, 2019; Salman et al., 2019b; Weng et al., 2018; Raghunathan et al., 2018), but these often yield weaker empirical performance. In this work, we focus on empirical defenses, specifically adversarial training.

3 Wasserstein Distance-based Threat Model

The p-Wasserstein distance between two probability distributions measures the "minimal effort" needed to rearrange the probability mass in one distribution so it matches the other one. More formally, given two distributions A and B over a metric space X with metric d, the p-Wasserstein distance is defined as $W_p(A, B) = \left[\inf_\Gamma \mathbb{E}_{(x,y)\sim\Gamma} d(x, y)^p\right]^{\frac{1}{p}}$ where Γ is a distribution over the product space $X \times X$ such that its marginals are A and B. Given two images represented as 3-dimensional tensors, $x, x' \in \mathbb{R}_+^{m,n,c}$ where m, n are the dimensions of the image and c is the number of channels; we measure the p-Wasserstein distance between them by normalizing both tensors into probability distributions. Intuitively, the p-Wasserstein distance measures the cost of transporting pixel mass to turn one image into the other[2], with the transport costing pixel distance (measured in d) to the p^{th} power per unit mass. We define the p-Wasserstein distance of images x and x' of the same dimensionality with non-zero ℓ_1-norms as:

$$W_p(x, x') = \sum_{i\in\{R,G,B\}} W_p\left(\frac{x_i}{||x_i||_1}, \frac{x_i'}{||x_i'||_1}\right) \tag{1}$$

Note that x and x' can have different ℓ_1 norms. According to Wong et al. (2019), an adversarial example x' of radius ϵ under the p-Wasserstein threat model is one that causes the neural network to give a different prediction than that of x and also satisfies: $W_p(x, x') \leq \epsilon$.

An Obvious Flaw This threat model only considers the probability distribution represented by the image but not the total pixel mass (i.e. the ℓ_1-norm of the image, reflected as brightness). We devise a trivial attack to break the Wasserstein-robust model from Wong et al. (2019) by simply dimming the brightness of images from the MNIST test set. By dividing the image tensor by 30 ($x \mapsto x/30$), we cause the model to misclassify 86% of the test set, even though all the dimmed data points have a Wasserstein distance of 0 to the original undimmed data points, as illustrated in Fig. 1 (Right).

Intuitively, a threat model that allows pixel mass movements should preserve the total pixel mass. Therefore, we further require our Wasserstein threat model to preserve the ℓ_1 norm, or the total pixel mass, after the perturbation. This is a commonly acknowledged constraint to make the Wasserstein distance a true metric (Rubner et al., 2000) and was implicit in the attack-finding PGD algorithm of Wong et al. (2019), where the original image's ℓ_1 norm is used to re-scale the their Sinkhorn projected probability distributions.

Definition 3.1. For a radius ϵ and an image x, we define our *constrained Wasserstein ball* to be

$$\mathcal{B}(x, \epsilon) = \{x' : W_p(x, x') \leq \epsilon, \|x\|_1 = \|x'\|_1, \overbrace{0 \leq x' \leq 1}^{\text{pixel range } [0,\,1]}\} \tag{2}$$

An adversarial example of f in an ϵ neighborhood of x is any $x' \in \mathcal{B}(x, \epsilon)$ such that $f(x') \neq f(x)$.

4 Constrained Sinkhorn Iteration

PGD Iteration During untargeted[3] PGD attacks, one updates the input iteratively following:

$$x^{(t+1)} = \underset{\mathcal{B}(x,\epsilon)}{\text{proj}}\left(x^{(t)} + \underset{\alpha}{\text{step}}(\nabla\ell(f(x^{(t)}), y))\right) \tag{3}$$

where $\text{proj}_{\mathcal{B}(x,\epsilon)} = \arg\min_{x'\in\mathcal{B}(x,\epsilon)} \|x - x'\|_2^2$ where $\mathcal{B}(x, \epsilon)$ is determined by the threat model. $step$ is a function that takes in the gradient of the model f with respect to $x^{(t)}$ and a step size α, and outputs a step which is added to $x^{(t)}$.

[2]We only allow pixel mass movement within channels to limit our problem to finding the 2D Wasserstein distance, following Wong et al. (2019).

[3]We maximize the loss of the correct label during an untargeted attack, and minimize the loss of a particular incorrect label during a targeted attack.

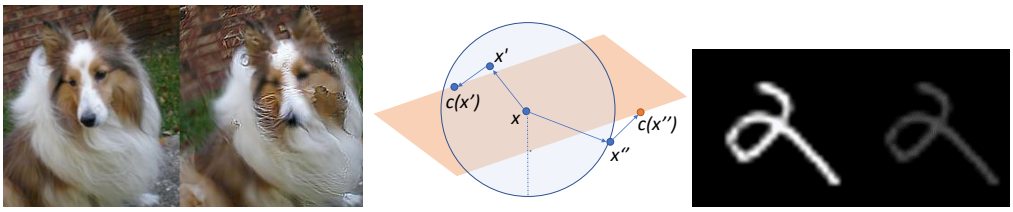

Figure 1: **Left**: A Wasserstein perturbation found by our attack that causes the ℓ_2 robust model from Engstrom et al. (2019) to misclassify a image from ImageNet (Russakovsky et al., 2015). **Middle**: Clamping the perturbed image (x', x'') can increase the Wasserstein distance to the original image (x), especially it is if near the boundary; the blue circle represents a Wasserstein ball of radius ϵ; the orange parallelogram represents an ℓ_∞ ball; $c(\cdot)$ is the clamping function. **Right**: We break the adversarially trained Wasserstein-robust model from Wong et al. (2019) by dimming the image brightness, which does not change the normalized image. The example here has its brightness reduced by 3x.

For example, ℓ_∞ PGD has

$$\underset{\alpha}{\text{step}} = \text{sign}(\nabla \ell(f(x^{(t)}), y)),\ \mathcal{B}(x, \epsilon) = \{x' : \|x - x'\|_\infty \leq \epsilon, 0 \leq x' \leq 1\} \tag{4}$$

and ℓ_2 PGD has

$$\underset{\alpha}{\text{step}} = \frac{\nabla \ell(f(x^{(t)}), y)}{\|\nabla \ell(f(x^{(t)}), y)\|_2},\ \mathcal{B}(x, \epsilon) = \{x' : \|x - x'\|_2 \leq \epsilon, 0 \leq x' \leq 1\} \tag{5}$$

In this section, we detail our changes to the function $\text{proj}_{\mathcal{B}(x,\epsilon)}$ where $\mathcal{B}(x, \epsilon)$ describes our threat model as in Eq. 2. We investigate the effect of the *step* function in Sec. 5.

To summarize our change to the $\text{proj}_{\mathcal{B}(x,\epsilon)}$ function, we

1. add a ℓ_∞ constraint to eliminate the need for clamping;
2. improve run-time by re-using the dual variables across PGD steps;
3. modify termination conditions to allow an explicit trade-off between safety and efficiency.

Critically, this algorithm allows us to safely explore stronger empirical attacks that find perturbations close to the allowed Wasserstein ball boundary around the original image.

The Original Algorithm in Wong et al. (2019) In the sequel, assume that all images are represented as vectors in $\mathbb{R}^n_+$ and bounded in $[0, 1]$. Projecting a perturbed image onto a Wasserstein ball around the original image requires solving the following optimal transport problem:

$$\underset{z \in \mathbb{R}^n_+, \Pi \in \mathbb{R}^{n \times n}_+}{\text{minimize}} \quad \frac{1}{2}\|w - z\|_2^2 \quad \text{s.t. } \Pi 1 = x,\ \ \Pi^T 1 = z,\ \ \langle \Pi, C \rangle \leq \epsilon \tag{6}$$

w is the image after taking a gradient step; x is the original image; Π is the transport plan; C is the cost matrix; and z is the projected image. Wong et al. (2019) made this projection efficient by approximating with an entropy-regularized optimization problem, which is solved using Lagrange multipliers and coordinate-descent on the dual variables.

Clamping: the flaw of the algorithm in Wong et al. (2019) As shown in Eq. 2, Eq. 4, and Eq. 5, we need to project the perturbed image into the valid pixel range. ℓ_p-based PGD attacks achieve this by clamping every pixel to $[0, 1]$ after projecting onto the ℓ_p ball. This ad-hoc clamping is also used in many works studying PGD attacks using the Wasserstein threat model (Wong et al., 2019; Levine & Feizi, 2019). However, clamping does not guarantee that the clamped image is still within the allowed Wasserstein radius, as illustrated in Fig. 1 (Middle). Under the attack of Wong et al. (2019), the perturbed image often uses less than 50% of the Wasserstein budget, and thus rarely goes outside of the Wasserstein ball despite this unsafe clamping. However, as we derive stronger attacks that project points onto the boundary of the Wasserstein ball, ad-hoc clamping causes the resultant images

to be over-budget by 50% on average, and in some cases by over 200% when using the original Sinhorn iteration projection. This shows that the need to clamp an un-normalized image introduces a critical flaw in Wong et al. (2019)'s attack-finding algorithm. We bypass this need for clamping by deriving a constrained Sinkhorn iteration to project directly onto the intersecton of the Wasserstein ball and an ℓ_∞ ball, making this ad-hoc clamping unnecessary.

Our Algorithm As discussed in Sec. 3, a perturbed distribution should not only be within the Wasserstein ball of the original image after normalization but also be elementwise within $[0, 1]$ after un-normalization by multiplying the original ℓ_1 norm $||x||_1$. This can be expressed as an additional ℓ_∞ constraint on z: $0 \leq z \leq r, r = \frac{1}{||w||_1}$, which can be simplified to $z_j \leq r$ for $j = 1, .., n$ since z is already constrained to be non-negative.

The new entropy-regularized optimized problem with the new constraint is as follows:

$$\underset{z\in\mathbb{R}_+^n, \Pi\in\mathbb{R}_+^{n\times n}}{\text{minimize}} \quad \frac{\lambda}{2}\|w - z\|_2^2 + \sum_{ij} \Pi_{ij} \log(\Pi_{ij}) \quad \text{s.t. } \Pi 1 = x, \Pi^T 1 = z, \langle \Pi, C\rangle \leq \epsilon, z_j \leq r \quad (7)$$

for $j = 1, ..., n$. We introduce dual variables $(\alpha, \beta, \psi, \phi)$ where $\psi, \phi_j \geq 0$, for $j = 0, ..., n$. The dual of the problem is:

$$\underset{\alpha,\beta\in\mathbb{R}_+^n, \psi\in\mathbb{R}_+, \phi\in\mathbb{R}_+^n}{\text{maximize}} \; g(\alpha, \beta, \psi, \phi), \quad \text{where}$$

$$g(\alpha, \beta, \psi, \phi) = \begin{cases} -\frac{1}{2\lambda}\|\beta + \phi\|_2^2 - \psi\epsilon + \alpha^T x + \beta^T w + \phi^T w \\ -r\sum_j \phi_j - \sum_{ij} \exp(\alpha_i) \exp(-\psi C_{ij} - 1) \exp(\beta_j). \end{cases}$$

We leave the details of the derivation to subsection A.2. By maximizing g w.r.t individual dual variables, we obtain the solution to the dual variables:

$$\arg\max_{\alpha_i} g(\alpha, \beta, \psi, \phi) = \log(x_i) - \log\left(\sum_j \exp(-\psi C_{ij} - 1) \exp(\beta_j)\right) \quad (8)$$

$$\arg\max_{\beta_j} g(\alpha, \beta, \psi, \phi) = \lambda w_j - \phi_j - W\left(\lambda \exp(-\phi_j + \lambda w_j) \sum_i \exp(\alpha_i - \psi C_{ij} - 1)\right) \quad (9)$$

Note that Eq. 8 is identical to the one in Wong et al. (2019); Eq. 9 has an additional variable ϕ_j. Since L is quadratic in ϕ with a negative coefficient, the maximization w.r.t. ϕ yields:

$$\arg\max_{\phi_j} g(\alpha, \beta, \psi, \phi) = \max(0, \lambda(w_j - r) - \beta_j)$$

We also perform Newton steps following Wong et al. (2019) to find the solution to ψ iteratively. The primal solutions to our optimal transport problem can be recovered as:

$$\Pi_{ij} = \exp(\alpha_i) \exp(-\psi C_{ij} - 1) \exp(\beta_j), \quad z = -\lambda^{-1}(\beta + \phi) + w. \quad (10)$$

4.1 Run-time Optimization

We initialize the dual variables to the stopping condition in the previous PGD step, instead of the general starting condition. Empirically, this allows the algorithm to converge faster, which is especially helpful since the additional constraint require more iterations to converge. The full description of our algorithm (1) can be found in subsection A.3.

Below we benchmark the attack run-time on MNIST test (Table 2) set using the original Sinkhorn iteration (Wong et al., 2019), our constrained version with and without this optimization, and pairing the optimized constrained Sinkhorn with our stronger attack, which is described in Sec. 5.

4.2 Termination Conditions

To ensure that the perturbed image is sufficiently compliant with our Wasserstein threat model, we incorporate both the Wasserstein and ℓ_1 norm constraints into the termination condition for Sinkhorn iterations. Specifically, we calculate the Wasserstein radius over-budget W_{over} and $\Delta\ell_1$, the deviation

Attack	Wong et al. (2019)	Constrained Proj.	+Optim.	Our Attack+Optim.
Run-time	34 mins	78 mins	38 mins	38 mins
Avg. ϵ	0.443	0.460	0.460	0.135

Table 2: Run-time and effectiveness for different attacks on a standard MNIST model using a single NVIDIA Tesla P100 GPU. Average ϵ is the mean Wasserstein radius needed to break the entire MNIST test set (the lower the better). The overhead from the additional constraint is offset by our run-time optimization. Our stronger attack does not have significant overhead.

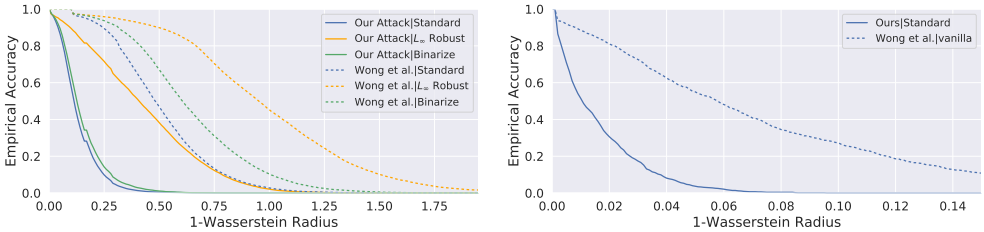

Figure 2: **Left**: Our new attack against the one in Wong et al. (2019) on MNIST (lower is better). Given the same Wasserstein budget, we reduce the classifier accuracy significantly more. **Right**: the same result on CIFAR-10.

between the sum of the output distribution and 1, once the algorithm has converged or run for at least a set number of iterations.

$$W_{over} = \sum_{ij} C_{ij} \exp(\alpha_i) \exp(-\psi C_{ij} - 1) \exp(\beta_j) - \epsilon \tag{11}$$

$$\Delta \ell_1 = abs(1 - \sum_j z_j) \tag{12}$$

We terminate the algorithm only when both quantities are sufficiently small. For attacks, we set the threshold for d_{W-over} to be $0.01 \times \epsilon$ and for $\Delta \ell_1$ to be 0.01. Note that for adversarial training, where such strict compliance of the constraints might not be necessary, one can use more lenient thresholds to speed-up the projection at the expense of strict compliance with the threat model.

5 Stronger Empirical Attacks

Now we take a look at the choice of the step function during PGD. The empirical attack proposed in Wong et al. (2019) uses the steepest descent w.r.t. the ℓ_∞ norm as their PGD step, with a step size tied to absolute pixel values.

Step Size The PGD step size for ℓ_p threat models is usually defined in the pixel space. On the other hand, Wasserstein threat models manipulate the underlying distribution of images, thus, it is more natural to define step sizes in the normalized distribution space. We find a trade-off between effectiveness and run-time - a larger step size can result in a stronger attack up to a point but takes longer for Sinkhorn iteration to converge. Empirically, we pick a step size of 0.06 for our experiments, which limits the change to a single pixel to 6% of the total pixel mass and is much larger compared to Wong et al. (2019). We quantified the effect of step size in subsection A.1, which shows that it does not affect the effectiveness of the attack once it is sufficiently large.

Gradient Step The additive perturbation before projection during PGD is a function of the gradient Eq. 3, and Wong et al. (2019) used the steepest descent w.r.t. the ℓ_∞ norm, which is effective the sign of the gradient. This is similar to the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015) from the ℓ_p robustness literature, which is known to yield weaker attacks than the steepest descent w.r.t. the ℓ_2 norm. Motivated by this, we replace the step function with the steepest descent w.r.t. the ℓ_2 norm, and match the ℓ_∞ norm of our gradient step and the original gradient step for a fair comparison. The empirical accuracy of both our attack and the original one is shown in Fig. 2.

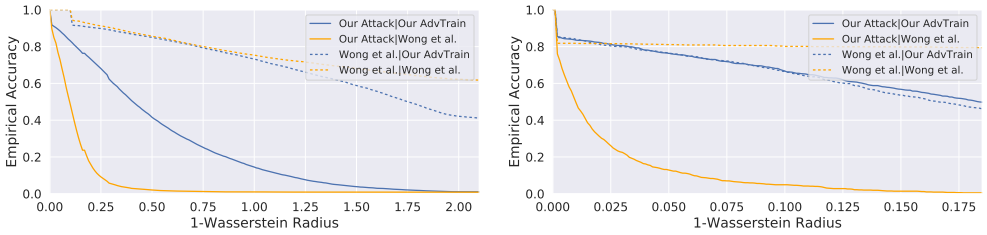

Figure 3: **Left**: Adversarial training under our attack and that of Wong et al. (2019) on MNIST (higher is better). Our defended model is more robust to our attack, while also robust to the attack from Wong et al. (2019). **Right**: the same result on CIFAR-10

Ablation Study We ablate the effect of increasing the step size and using the steepest descent w.r.t. the ℓ_2 norm (Table 3). Our result shows that the improvement comes from both changes, and significant gain comes from using a different gradient step that considers not just the sign.

MNIST	1-Wasserstein $\epsilon \times n_{pixel}$	5	10	20	50	100	200	500	1000
	Wong et al. (2019) (%)	100	100	100	100	94	91	83	71
	+large α (%)	100	96	96	94	88	69	12	0
	+large α + ℓ_2 descent (%)	100	90	85	68	39	8	1	1
CIFAR-10	1-Wasserstein $\epsilon \times n_{pixel}$	5	10	20	50	100	200	500	1000
	Wong et al. (2019) (%)	82	82	82	82	82	81	80	77
	+large α (%)	81	80	78	70	55	23	1	0
	+large α + ℓ_2 descent (%)	76	68	58	35	20	9	1	0

Table 3: Ablation study on the effect of larger step size (α) and using steepest descent w.r.t. the ℓ_2 norm. The model under attack is the adversarially trained Wasserstein-robust model from (Wong et al., 2019).

6 Defending against a Wasserstein Adversary

Following Wong et al. (2019), we use adversarial training to defend against our Wasserstein attack. We train our MNIST model for 100 epochs and CIFAR-10 model for 200 epochs with a terminition condition of $W_{over} = 0.1$ and $\Delta\ell_1 = 0.1$ (introduced in subsection 4.2) for efficiency. The model is trained on perturbed inputs with ϵ growing from 0.1 to 10 on an exponential schedule.

We report our result in Fig. 3. Our defended model is more robust against our stronger attack compared to the previously defended model from Wong et al. (2019), while still being generally robust against the attack in Wong et al. (2019). Note that for large ϵ, our defended model performs worse than the previously defended model under the weaker attack. We believe this could be that the adversarially trained model from Wong et al. (2019) overfits the weaker attack, similar to the "catastrophic overfitting" phenomenon (Wong et al., 2020) associated with FGSM. We are not able to close this gap by increasing model capacity.

7 Not Ready to Defend Against Common Natural Perturbations

The Wasserstein threat model is motivated by ℓ_p threat model's failure to bound common natural perturbations, e.g., translation, rotation, and blurring. Such perturbations, when their magnitude is small, are barely perceptible to human, yet they incur a large change in the ℓ_p distance. One reason behind this is the ℓ_p threat model perturbs each pixel independently, while the Wasserstein threat model does not. We investigate empirically if a Wasserstein-robust model is more robust against translations, rotations, and blurring. For some intuition regarding the ℓ_p and Wasserstein distance incurred by such perturbations, please see subsection A.4.

		Clean	Translation			Rotation			Gaussian Blur		
		/	5%	10%	20%	5°	10°	20°	3	5	7
MNIST	Standard (%)	98.9	97.8	85.8	48.3	98.5	97.7	91.9	98.6	93.1	62.4
	Our Robust (%)	92.8	91.2	82.3	49.3	91.0	88.7	79.1	88.0	75.1	58.4
CIFAR-10	Standard (%)	94.8	94.5	94.5	93.6	91.6	86.6	64.1	35.1	17.8	15.4
	Our Robust (%)	84.4	84.0	84.0	81.7	82.8	81.3	70.6	69.5	35.1	24.7

Table 4: The top-1 classification accuracy of a standard and our Wasserstein-robust model under fixed common perturbations. Gaussian blur is parameterized by the width of the kernel in pixels.

As it turns out, we can rarely defend against such common perturbations by adversarially training against a Wasserstein adversary, as shown in Table 4. The only effective case is when defending against Gaussian blurs on CIFAR-10. We reflect on this result from two perspectives.

Better Defenses The Wasserstein threat model is much less studied than that of ℓ_p, and the naive defense we use, adversarial training, does not defense against a large enough radius. Once we improve our ability to defend against a larger Wasserstein radius, we ought to gain robustness against the natural perturbations we explored in this section. This motivates more future works on defending against the Wasserstein threat model.

Constraints on the Threat Model Furthermore, we believe that perturbations exist on a spectrum of semantic specificity. On one end, we have a narrow set of semantic perturbations such as translations and rotations. We can defend against them effectively through data augmentation. On the other end, we have highly expressive perturbation classes such as those bounded by a Wasserstein ball, which are harder to defend against, but do not require knowing which perturbations are likely to occur. Orthogonal to finding stronger defenses, we can constrain the expressiveness of our threat model to make existing defenses more specific to the type of perturbations of interest.

For example, the common perturbations we study exhibit highly “smooth" movements with a directional (for translations and rotations) or a radial (for Gaussian blurs) pattern. This “smoothness" constraint is also exhibited in perturbations such as lens distortions and refraction, where one would reasonably assume that the Wasserstein distance is more meaningful than ℓ_p due to the dependency on neighboring pixels.

We leave the rigorous formulation of such a “smoothness" constraint to future works, and note a few ideas and challenges in subsection A.5

8 Conclusion and Future Work

We introduced a better-defined threat model based on Wasserstein distance for all images of the same dimensionality. A procedure naively brought over from the ℓ_p threat model could lead to adversarial examples that are not compliant with the Wasserstein threat model. We fixed this by proposing a constrained version of the original Sinkhorn iteration algorithm, which projects onto the intersection of a Wasserstein ball and an ℓ_∞ ball directly. This allowed us to safely explore attacks that come close to the allowed Wasserstein boundary. By increasing the attack step size and using the steepest descent w.r.t. the ℓ_2 norm of the gradient, we obtained a significantly stronger empirical Wasserstein attack that breaks previously defended models easily. We defended against our attack with adversarial training, and showed that our defended model was generally robust against prior attacks.

Nonetheless, our defended model remained inadequate to defend against large, real-world perturbations, such as translations, rotations, and blurring. This highlighted the need to design better defenses against the Wasserstein threat model, as current approaches did not yield a large enough defense radius. On the other hand, we noted that these perturbations exhibited smoothness, and could potentially be better modeled if we introduce smoothness constraints to our threat model. However, key difficulties must be overcome before such constraints can be efficiently optimized, and we left such explorations to future works.

Acknowledgments and Disclosure of Funding

We thank Tony Duan, Eric Wong, Ashish Kapoor, Jerry Li, Ilya Razenshteyn, Jeremy Cohen and the anonymous reviewers for their helpful comments.

References

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

Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. *arXiv preprint arXiv:1902.02918*, 2019.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial Robustness as a Prior for Learned Representations. *arXiv:1906.00945 [cs, stat]*, September 2019. URL `http://arxiv.org/abs/1906.00945`. arXiv: 1906.00945.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *ICLR*, 2015.

Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks. In *Advances in Neural Information Processing Systems*, pp. 10408–10418, 2019.

Alexander Levine and Soheil Feizi. Wasserstein Smoothing: Certified Robustness against Wasserstein Adversarial Attacks. *arXiv:1910.10783 [cs, stat]*, October 2019. URL `http://arxiv.org/abs/1910.10783`. arXiv: 1910.10783.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Jeet Mohapatra, Tsui-Wei, Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. Towards Verifying Robustness of Neural Networks Against Semantic Perturbations. *arXiv:1912.09533 [cs, stat]*, December 2019. URL `http://arxiv.org/abs/1912.09533`. arXiv: 1912.09533.

S. Peleg, M. Werman, and H. Rom. A unified approach to the change of resolution: space and gray-level. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):739–742, July 1989. ISSN 1939-3539. doi: 10.1109/34.192468.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *International Conference on Learning Representations (ICLR), arXiv preprint arXiv:1801.09344*, 2018.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115 (3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Hadi Salman, Greg Yang, Jerry Li, Pengchuan Zhang, Huan Zhang, Ilya Razenshteyn, and Sebastien Bubeck. Provably Robust Deep Learning via Adversarially Trained Smoothed Classifiers. *arXiv:1906.04584 [cs, stat]*, November 2019a. URL `http://arxiv.org/abs/1906.04584`. arXiv: 1906.04584.

Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. A convex relaxation barrier to tight robustness verification of neural networks. In *Advances in Neural Information Processing Systems*, pp. 9832–9842, 2019b.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel. Towards fast computation of certified robustness for ReLU networks. In *International Conference on Machine Learning*, 2018.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning (ICML)*, pp. 5283–5292, 2018.

Eric Wong, Frank R. Schmidt, and J. Zico Kolter. Wasserstein Adversarial Examples via Projected Sinkhorn Iterations. *arXiv:1902.07906 [cs, stat]*, February 2019. URL `http://arxiv.org/abs/1902.07906`. arXiv: 1902.07906.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training, 2020.

A Appendix

A.1 Attacking with larger step sizes

We show in Table 5 the effect of attacking our adversarially trained model with different step sizes. The model under attack is the same as the one in Sec. 6, which is trained against an adversary with a step size of 0.06. We notice the trade-off between run-time and attack effectiveness as we increase the attack step size. Notably, increasing the step size beyond what the model was trained against does not break the model. Empirically, Sinkhorn iterations as currently implemented run into numerical issues with step sizes larger than 0.08. We hope future works can address this numerical stability issue.

When attacking within a small Wasserstein radius (e.g., < 0.06), we also find it helpful to decrease the PGD step size accordingly. Intuitively, if only less than 6% of the total pixel mass is allowed to move by 1 pixel, a step size of 0.06 (as we have chosen for our experiments), which allows a single pixel to move by 6% of the total pixel mass, only makes convergence slow without making the attack more effective. We empirically set the step size to $\min(\frac{\epsilon}{2}, \alpha)$, where ϵ is the Wasserstein radius and α is the step size.

MNIST	Step Size	Run-time	Acc. (%)@100	200	400	800	1600
	0.02	1.8 hrs	84	72	49	21	2
	0.04	2.1 hrs	81	68	42	15	1
	0.06	2.5 hrs	81	67	41	14	1
	0.08	3.0 hrs	81	67	41	14	1
CIFAR-10	Step Size	Run-time	Acc. (%)@100	200	400	800	1600
	0.02	3.7 hrs	52	44	44	44	44
	0.04	4.6 hrs	52	43	43	43	43
	0.06	4.9 hrs	52	43	43	43	43
	0.08	5.0 hrs	51	42	42	42	42

Table 5: The trade-off between run-time and attack effectiveness for MNIST and CIFAR-10. The accuracy is evaluated on our adversarially trained model (against an attack step size of 0.06) at various Wasserstein radii multiplied by the per image pixel count n_{pixel}. Run-time is recorded on a single NVIDIA Tesla P100 GPU.

A.2 Derivation of the Lagrangian

Proof. For convenience, we multiply the objective by λ, expand the L_∞ norm to individual pixels, and solve this problem instead:

$$\begin{aligned} \underset{z\in\mathbb{R}_+^n, \Pi\in\mathbb{R}_+^{n\times n}}{\text{minimize}} \quad & \frac{\lambda}{2}\|w-z\|_2^2 + \sum_{ij}\Pi_{ij}\log(\Pi_{ij}) \\ \text{subject to} \quad & \Pi 1 = x \\ & \Pi^T 1 = z \\ & \langle \Pi, C\rangle \le \epsilon \\ & z_j \le r, j = 1, ..., n. \end{aligned} \tag{13}$$

Here r is the maximal pixel value. Introducing dual variables $(\alpha, \beta, \psi, \vec{\phi})$ where $\psi, \phi_j \ge 0$, for j=0, ..., n, the Lagrangian is

$$\begin{aligned} & L(z, \Pi, \alpha, \beta, \psi, \vec{\phi}) \\ =& \frac{\lambda}{2}\|w-z\|_2^2 + \sum_{ij}\Pi_{ij}\log(\Pi_{ij}) + \psi(\langle \Pi, C\rangle - \epsilon) \\ & + \sum_j \phi_j(z_j - r) + \alpha^T(x - \Pi 1) + \beta^T(z - \Pi^T 1). \end{aligned} \tag{14}$$

The KKT optimality conditions are now

$$\begin{aligned}\frac{\partial L}{\partial \Pi_{ij}} &= \psi C_{ij} + (1 + \log(\Pi_{ij})) - \alpha_i - \beta_j = 0 \\ \frac{\partial L}{\partial z_j} &= \lambda(z_j - w_j) + \beta_j + \phi_j = 0\end{aligned} \tag{15}$$

so at optimality, we must have

$$\begin{aligned}\Pi_{ij} &= \exp(\alpha_i)\exp(-\psi C_{ij} - 1)\exp(\beta_j) \\ z &= -\frac{\beta + \phi}{\lambda} + w\end{aligned} \tag{16}$$

Plugging in the optimality conditions, we get

$$\begin{aligned}&L(z^*, \Pi^*, \alpha, \beta, \psi, \phi) \\ =&\frac{\lambda}{2}\|w - z\|_2^2 + \sum_{ij}\Pi_{ij}\log(\Pi_{ij}) + \psi(\langle \Pi, C\rangle - \epsilon) \\ &+ \sum_j \phi_j(z_j - r) + \alpha^T(x - \Pi 1) + \beta^T(z - \Pi^T 1) \\ =&\frac{-\|\beta\|_2^2 - \|\phi\|_2^2}{2\lambda} - \psi\epsilon + \alpha^T x + \beta^T w + \phi^T w - r\sum_j \phi_j \\ &- \frac{\beta^T\phi}{\lambda} - \sum_{ij}\exp(\alpha_i)\exp(-\psi C_{ij} - 1)\exp(\beta_j) \\ =&g(\alpha, \beta, \psi, \phi)\end{aligned} \tag{17}$$

so the dual problem is to maximize g over $\alpha, \beta, \psi, \phi \geq 0, for j = 0, ..., n$. □

A.3 Changes to the Sinkhorn iteration algorithm

We further make a few changes to the Sinkhorn iterations proposed by Wong et al. (2019) to make the algorithm more efficient. First, note that the argmax

$$\phi_j^*, \beta_j^* := \arg\max_{\phi_j, \beta_j} g(\alpha, \beta, \psi, \phi)$$

can be obtained in one pass. First suppose the argmax ϕ_j^* is nonnegative. Then by solving $\partial g/\partial\beta_j = \partial g/\partial\phi_j = 0$ we have the solutions

$$\begin{aligned}\beta_j^* &= \log r - \log\sum_i \exp(\alpha_i)\exp(-\psi C_{ij} - 1) \\ \phi_j^* &= \max(\lambda(w_j - r) - \beta_j^*, 0)\end{aligned}$$

If $\lambda(w_j - r) - \beta_j$ above is negative, then ϕ_j^* has to be 0, and therefore

$$\beta_j^* = \lambda w_j - W\left(\lambda\exp(\lambda w_j)\sum_i \exp(\alpha_i)\exp(-\psi C_{ij} - 1)\right)$$

Our final algorithm can be described in the following pseudo-code (1).

A.4 Common Real-world Perturbations under ℓ_p and Wasserstein distance

We first measure the distance change under ℓ_p and 1-Wasserstein when we apply such common perturbations; the result is in Table 6. The raw distances are not comparable between the two metrics, since the ℓ_2 distance is on the pixel space, while 1-Wasserstein is on the underlying distribution space. We note that the 1-Wasserstein distance incurred by such perturbations scales roughly linearly with the magnitude, while the ℓ_2 distance does not.

Algorithm 1 Projected Sinkhorn iteration to project x onto the ϵ Wasserstein ball around y. We use $\cdot$ to denote element-wise multiplication. The $\log$ and $\exp$ operators also apply element-wise.

input: $x, w \in \mathbb{R}^n, C \in \mathbb{C}^{n \times n}, \lambda \in \mathbb{R}$
Initialize $\alpha_i, \beta_i \coloneqq \log(1/n)$ for $i = 1, \ldots, n$ and $\psi, \phi \coloneqq 1$
$u, v \coloneqq \exp(\alpha), \exp(\beta)$
while $\alpha, \beta, \psi, \phi$ not converged **do**
  // update K
  $K_\psi \coloneqq \exp(-\psi C - 1)$

  // block coordinate descent iterates
  $\alpha \coloneqq \log(x) - \log(K_\psi v)$
  $u \coloneqq \exp(\alpha)$
  $\beta_1 \coloneqq \lambda w - \phi - W\left(u^T K_\psi \cdot \lambda \exp(\lambda w - \phi)\right)$
  $\beta_2 \coloneqq \log r - \log \sum_i \exp(\alpha_i) \exp(-\psi C_{ij} - 1)$
  $\phi \coloneqq \max(\lambda(w - r) - \beta_1, 0)$
  $\beta \coloneqq where(\phi < 0, \beta_2, \beta_1)$
  $v \coloneqq \exp(\beta)$

  // Newton step
  $g \coloneqq -\epsilon + u^T (C \cdot K_\psi) v$
  $h \coloneqq -u^T (C \cdot C \cdot K_\psi) v$

  // ensure $\psi \geq 0$
  $\psi \coloneqq \max(\psi - g/h, 0)$
end while
return: $w - (\beta + \phi)/\lambda$

		Translation			Rotation			Gaussian Blur		
		5%	10%	20%	5°	10°	20°	3	5	7
MNIST	ℓ_2	5.2	8.1	10.3	3.7	5.4	7.8	3.2	5.5	6.5
	1-Wass.	1.0	2.2	4.2	0.5	0.8	1.5	0.3	1.0	1.6
CIFAR-10	ℓ_2	41.0	61.6	82.6	38.4	54.8	71.4	14.8	25.4	31.3
	1-Wass.	1.0	2.0	3.7	0.8	1.6	2.8	0.8	1.6	2.2

Table 6: The change under ℓ_p or 1-Wasserstein after applying fixed common perturbations. Results are averaged over the entire test set of either MNIST or CIFAR-10. Gaussian blur is parameterized by the width of the kernel in pixels. The numerical values of the two distance metrics are not directly comparable. Translation is based on the width of the image, which is 28 pixels for MNIST and 32 pixels for CIFAR-10.

A.5 Potential smoothness constraints on the threat model

We can directly constrain the total variance of the transport plan Π:

$$\sum_{\vec{i},\vec{j} \in \Pi} \Pi_{\vec{i} \to \vec{j}} \|\vec{j} - \vec{i}\|^2 - \left(\sum_{\vec{i},\vec{j} \in \Pi} \Pi_{\vec{i} \to \vec{j}} (\vec{j} - \vec{i}) \right)^2$$

Here, $\vec{i}$ and $\vec{j}$ are the position of pixels. The challenge for this approach is that this total variance constraint is not convex, and therefore cannot be easily integrated into the existing optimization framework.

A more ad-hoc approach is to use only the low-frequency gradient information when taking a PGD step. This can be done through applying a low-pass filter, e.g., a Gaussian filter, to the gradient matrix. Low-frequency perturbations have been studied under the ℓ_p threat model (CITE); the

dynamics between the perturbation frequency under the Wasserstein threat model and how close it is to modeling natural perturbations such as translations, rotations, and blurring is an open problem.

Another potential challenge is the use of local transport plans in Wong et al. (2019) for efficiency. This approximation can fundamentally limit the magnitude of the perturbations we can model; for example, using a 5-by-5 local transport plan limits the translations we can model to at most 2 pixels.