CERTIFIED DEFENSES FOR ADVERSARIAL PATCHES

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial patch attacks were recently recognized as the most practical threat model against real-world computer vision systems. Most published defenses against patch attacks are based on preprocessing input images to mitigate adversarial noise. The first contribution of this paper is a set of experiments demonstrating that such defense strategies can easily be broken by white-box adversaries. Motivated by this finding, we present an extension of certified defense algorithms and propose significantly faster variants for robust training against patch attacks. Finally, we experiment with different patch shapes for testing, and observe that robustness to such attacks transfers surprisingly well.

1 INTRODUCTION

Despite the great success of neural networks for vision problems, a range of adversarial attacks have been proposed to fool such models. An adversarial attack carefully perturbs the input to a machine learning model in order to manipulate its output, which poses serious security threats. One of the widely studied threat models is the norm-bounded attack (Madry et al., 2017; Tramèr & Boneh, 2019; Shafahi et al., 2019), where the adversary is allowed to perturb *all* pixels in an image provided that the ℓ_p-norm of the added noise is within given bounds. Other adversarial models were also proposed, such as functional (Laidlaw & Feizi, 2019), rotation/translation (Engstrom et al., 2017), and Wasserstein (Wong et al., 2019), all of which allow adversaries to modify any pixel of the image.

For most physical-world attacks, whole-image perturbations seem rather unrealistic. A physical adversary usually modifies an object using stickers or paint, which limits the number of pixels they can manipulate. As such, the more practical *patch attack* model was proposed (Brown et al., 2017). In a patch attack, the adversary may only change the pixels in a confined region, but is otherwise free to choose the values yielding the strongest attack. The threat to real-world computer vision systems is well-demonstrated in recent literature where carefully crafted patches can fool a classifier with high reliability (Brown et al., 2017; Karmon et al., 2018), or make objects invisible to an object detector (Lee & Kolter, 2019). Moreover, special glasses with an adversarial frame were shown to fool a face recognition system (Sharif et al., 2017). In light of such effective physical-world patch attacks, very few defenses are known to date.

In this paper, we study principled defenses against patch attacks. We begin by looking at existing defenses in the literature that claim to be effective against patch attacks, including Local Gradient Smoothing (LGS) (Naseer et al., 2019) and Digital Watermarking (DW) (Hayes, 2018). Similar to what has been observed for whole-image attacks by (Athalye et al., 2018), we show that these patch defenses can easily be broken by stronger adversaries. Concretely, we demonstrate successful white-box attacks, where the adversary optimizes against a given model including any preprocessing steps. To cope with such potentially stronger adversaries, we train a robust model that produces a lower-bound on adversarial accuracy. In particular, we propose the first *certifiable* defense against patch attacks by extending interval bound propagation (IBP) defenses (Gowal et al., 2018; Mirman et al., 2018) to rebuff patch attacks. We also propose modifications to IBP training in order to make it faster against patch attacks. Finally, we study the generalization of certified patch defenses to patches of different shapes, and observe that robustness transfers well across different patch types.

2 Problem Setup

We consider a white-box adversary that is allowed to choose the location of the patch (chosen from a set $\mathbb{L}$ of possible locations) and can modify pixels within the particular patch (chosen from the set $\mathbb{P}$) similar to (Karmon et al., 2018). An attack is successful if the adversary changes the classification of the network to a wrong label. Throughout this paper, we are interested in the patch attack robust accuracy (adversarial accuracy for short) as defined by

$$\mathop{\mathbb{E}}_{x \sim X} \min_{p \in \mathbb{P}, l \in \mathbb{L}} \mathcal{X}[f(A(x,p,l);\theta) = y], \tag{1}$$

where the operator A, places the adversarial patch p on a given image x at location l, f is a neural network with parameter θ, and X is a distribution of images, and $\mathcal{X}$ is a characteristic function that takes value 1 if its argument is true, and 0 otherwise.

In this model, the strength of the adversary can vary depending on the set of possible patches allowed, and the type of perturbation allowed within the patch. In what follows, we assume the standard setup in which the adversary is allowed any perturbation that maintains pixel intensities in the range $[0, 1]$. Unless otherwise noted, we also assume the patch is restricted to a square of prescribed size. We consider two different options for the set $\mathbb{L}$ of possible patch locations. First, we consider a weak adversary that can only place patches at the corner of an image. We find that even this weak model is enough to break existing patch defenses. Then, we consider a stronger adversary with no restrictions on patch location, and use this model to evaluate our proposed defenses. Note that an adversary, when restricted to modify only a square patch at location l in the image, has the freedom to modify any non-square subset of these pixels. In other words, a certified defense against square patch attacks also provably subverts any non-square patch attack that fits inside a small enough square.

In general, calculating the adversarial accuracy (1) is intractable due to non-convexity. Common approaches try to approximate it by solving the inner minimization using a gradient-based method. However, in Section 3, we show that depending on how the minimization is solved, the upper bound could be very loose: a model may appear to be very robust, but fails when faced with a stronger attack. To side-step the arms race between attacks and defenses, in Section 4, we extend the work of (Gowal et al., 2018) and (Mirman et al., 2018) to train a network that produces a lower bound on adversarial accuracy. We will refer to approximations of the upper bound as *empirical adversarial accuracy* and the lower bound as *certified accuracy*.

3 Vulnerability of Existing Defenses

We start by examining existing defense strategies that claim to be effective against patch attacks. Similar to what has been observed for whole-image attacks by Athalye et al. (2018), we show that these patch defenses can easily be broken by white-box attacks, where the adversary optimizes against a given model including any preprocessing steps.

3.1 Existing Defenses

Under our threat model, two defenses have been proposed that each use input transformations to detect and remove adversarial patches.

The first defense is based on the observation that the gradient of the loss with respect to the input image often exhibits large values near the perturbed pixels. In (Hayes, 2018), the proposed digital watermarking (DW) approach exploits this behavior to detect unusually dense regions of large gradient entries using saliency maps, before masking them out in the image. Despite a 12% drop in accuracy on clean (non-adversarial) images, this defense method supposedly achieves an empirical adversarial accuracy of 63% for non-targeted patch attacks of size 42×42 (2% of the image pixels), using 400 randomly picked images from ImageNet (Deng et al., 2009) on VGG19 (Simonyan & Zisserman, 2014).

The second defense, Local Gradient Smoothing (LGS) by Naseer et al. (2019) is based on the empirical observation that pixel values tend to change sharply within these adversarial patches. In other words, the image gradients tend to be large within these adversarial patches. Note that the image gradient here differs from the gradient in Hayes (2018), the former is with respect the changes

Table 1: Empirical adversarial accuracy of ImageNet classifiers defended with Local Gradient Smoothing and Digital Watermarking. We consider two types of adversaries, one that takes the defense into account during backpropagation and one that does not

		Patch Size		
Attack	Defense	42×42	52×52	60×60
IFGSM	LGS	78%	75%	71%
IFGSM + LGS	LGS	14%	5%	3%
IFGSM	DW	56%	49%	45%
IFGSM + DW	DW	13%	8%	5%

of adjacent pixel values and the later is with respect to the classification loss. Naseer et al. (2019) propose suppressing these adversarial noise by multiplying each pixel with one minus its image gradient as in (2). The λ here is a smoothing hyper-parameter. To make their methods more effective, Naseer et al. (2019) also preprocess the image gradient with a normalization and a thresholding step.

$$\hat{x} = x \odot (1 - \lambda g(x)) \tag{2}$$

Naseer et al. (2019) claim the best adversarial accuracy on ImageNet with respect to patch attacks among all of the defenses we studied. They also claim that their defense is resilient to Backward Pass Differential Approximation (BPDA) from Athalye et al. (2018), one of the most effective methods to attack models that includes a non-differentiable operator as a preprocessing step.

3.2 Breaking Existing Defenses

Using a similar setup as in (Hayes, 2018; Naseer et al., 2019), we are able to mostly replicate the reported empirical adversarial accuracy for Iterative Fast Gradient Sign Method (IFGSM), a common gradient based attack, but we show that when the preprocessing step is taken into account, the empirical adversarial accuracy on ImageNet quickly drops from $\tilde{7}0\%(\tilde{5}0\%)$ for LGS(DW) to levels that are almost the same as random guessing $(\tilde{1}0\%)$ as shown in Table 1.

Specifically, we break DW (Hayes, 2018) by applying BPDA, in which the non-differentiable operator is approximated with an identity mapping during the backward pass. We break LGS (Naseer et al., 2019) by directly incorporating the smoothing step during backpropagation. Even though the windowing and thresholding steps are non-differentiable, the smoothing operator provides enough gradient information for the attack to be effective.

To make sure that our evaluation is fair, we used the exact same models as Hayes (2018) (VGG19) and Szegedy et al. (2016) (Inception V3). We also consider a weaker set of attackers that can only attack the corners, the same as their setting. Further, we ensure that we were able to replicate their reported result under similar setting.

4 Certified Defenses

Given the ease with which these supposedly strong defenses are broken, it is natural to seek methods that can rigorously guarantee robustness of a given model to patch attacks. With such *certifiable* guarantees in hand, we need not worry about an adversary with a stronger optimizer, or a more clever algorithm for choosing patch locations.

4.1 Background on certified defenses

Certified defenses have been intensely studied with respect to norm-bounded attacks (Cohen et al., 2019; Wong & Kolter, 2017; Gowal et al., 2018; Mirman et al., 2018; Zhang et al., 2019; Weng et al., 2018). In all of these methods, in addition to the prediction model, there is also a verifier. Given a model and an input, the verifier outputs a certificate if it is guaranteed that the image can not be adversarially perturbed. This is done by checking whether there exists any nearby image (within a prescribed ℓ_p distance) with a different label than the image being classified. Alternatively, the verifier may output a lower bound on the distance to the nearest image of a different label. This latter distance is referred to as the *certifiable radius.* Most of these verifiers provide a rather loose bound on the certifiable radius. However, if the verifier is differentiable, then the network can be trained with a loss that promotes tightness of this bound. We use the term *certificate training* to refer to the process of training with a loss that promotes strong certificates.

Interval bound propagation (IBP) (Mirman et al., 2018; Gowal et al., 2018) is a very simple verifier that uses layer-wise interval arithmetic to produce a certificate. Even though the IBP certificate is generally loose, after certificate training, it yields state-of-the-art certifiably-robust models for l_∞-norm bounded attacks (Gowal et al., 2018; Zhang et al., 2019). In this paper, we extend IBP to train certifiably-robust networks resilient to patch attacks. We first introduce some notation and basic algorithms for IBP training.

Notation We represent a neural network with a series of transformations $h^{(k)}$ for each of its k layers. We use $z^{(k)} \in \mathbb{R}^{n_k}$ to denote the output of layer k, where n_k is the number of units in the k^{th} layer and $z^{(0)}$ stands for the input. Specifically, the network computes

$$z^{(k)} = h^{(k-1)}(z^{(k-1)}) \qquad \forall k = 1, \dots, K.$$

Certification Problem To produce a certificate for an input x_0, we want to verify that the following condition is true with respect to all possible labels y:

$$(e_{y_{true}} - e_y)^T z^{(K)} = \mathbf{m}_y \geq 0 \qquad \forall z^{(0)} \in \mathbb{B}(x_0) \qquad \forall y. \tag{3}$$

Here, e_i is the i^{th} basis vector, and $\mathbf{m}_y$ is called the margin following Wong & Kolter (2017). Note that $\mathbf{m}_{y_{true}}$ is always equal to 0. The vector $\mathbf{m}$ contains all margins corresponding to all labels. $\mathbb{B}(x_0)$ is the constraint set over which the adversarial input image may range. In a conventional setting, this is an ℓ_∞ ball around x_0. In the case of patch attack, the constraint set contains all images formed by applying a patch to x_0;

$$\mathbb{B}(x_0) = \{A(x_0, p, l) | p \in \mathbb{P} \text{ and } l \in \mathbb{L}\}. \tag{4}$$

The Basics of Interval Bound Propagation (IBP) We now describe how to produce certificates using interval bound propagation as in (Gowal et al., 2018). Suppose that for each component in $z^{(k-1)}$ we have an interval containing all the values which this component reaches as $z^{(0)}$ ranges over the ball $\mathbb{B}(x_0)$. If $z^{(k)} = h^{(k)}(z^{(k-1)})$ is a linear (or convolutional) layer of the form $z^{(k)} = W^{(k)} z^{(k-1)} + b^{(k)}$, then we can get an *outer approximation* of the reachable interval range of activations by the next layer $z^{(k)}$ using the formulas below

$$\overline{z}^{(k)} = W^{(k)} \frac{\underline{z}^{(k-1)} + \overline{z}^{(k-1)}}{2} + |W^{(k)}| \frac{\underline{z}^{(k-1)} + \overline{z}^{(k-1)}}{2}, \tag{5}$$

$$\underline{z}^{(k)} = W^{(k)} \frac{\underline{z}^{(k-1)} + \overline{z}^{(k-1)}}{2} - |W^{(k)}| \frac{\underline{z}^{(k-1)} + \overline{z}^{(k-1)}}{2}. \tag{6}$$

Here $\overline{z}^{(k-1)}$ contains the upper bound of each interval, and $\underline{z}^{(k-1)}$ denotes the lower bound. Alternatively, if $h^{(k)}(z^{(k-1)})$ is an element-wise monotonic activation (e.g., a ReLU), then we can calculate the *outer approximation* of the reachable interval range of the next layer using the formulas below

$$\overline{z}^{(k)} = h^{(k)}(\overline{z}^{(k-1)}) \tag{7}$$

$$\underline{z}^{(k)} = h^{(k)}(\underline{z}^{(k-1)}). \tag{8}$$

When the feasible set $\mathbb{B}(x_0)$ represents a simple ℓ_∞ attack, the range of possible $z^{(0)}$ values is trivially characterized by an interval bound $\overline{z}^{(0)}$ and $\underline{z}^{(0)}$. Then, by iteratively applying the above rules, we can propagate intervals through the network and eventually get $\overline{z}^{(K)}$ and $\underline{z}^{(K)}$. A certificate can then be given if we can show that (3) is always true for outputs in the range $\overline{z}^{(K)}$ and $\underline{z}^{(K)}$ with respect to all possible labels. More specifically, we can check that the following holds for all y

$$\underline{\mathbf{m}}_y = e_{y_{true}}^T \underline{z}^{(K)} - e_y^T \overline{z}^{(K)} = \underline{z}_{y_{true}}^{(K)} - \overline{z}_y^{(K)} \geq 0 \quad \forall y \tag{9}$$

Training for Interval Bound Propagation To train a network to produce accurate interval bounds, we simply replace standard logits with the $-\underline{\mathbf{m}}$ vector in (3). Note that all elements of $\underline{\mathbf{m}}$ need to be larger than zero to satisfy the conditions in (3), and $\underline{\mathbf{m}}_{y_{true}}$ is always equal to zero. Put simply, we would like $\underline{\mathbf{m}}_{y_{true}} = 0$ to be the least of all margins. We can promote this condition by training with the loss function

$$\text{Certificate Loss} = \text{Cross Entropy Loss}(-\underline{\mathbf{m}}, y). \tag{10}$$

Unlike regular neural network training, stochastic gradient descent for minimizing equation 10 is unstable, and a range of tricks are necessary to stabilize IBP training (Gowal et al., 2018). The first trick is merging the last linear weight matrix with $(e_y - e_{y_{true}})$ before calculating $-\underline{\mathbf{m}}_y$. This allows a tighter characterization of the interval bound that noticeably improves results. The second trick uses an "epsilon schedule" in which training begins with a perturbation radius of zero, and this radius is slowly increased over time until a sentinel value is reached. Finally, a mixed loss function containing both a standard natural loss and an IBP loss is used.

In all of our experiments, we use the merging technique and the epsilon schedule, but we do not use a mixed loss function containing a natural loss as it does not increase our certificate performance.

4.2 Certifying against patch attacks

We can now describe the extension of IBP to patches. If we specify the patch location, one can represent the feasible set of images with a simple interval bound: for pixels within the patch, the upper and lower bound is equal to 1 and 0. For pixels outside of the patch, the upper and lower bounds are both equal to the original pixel value. By passing this bound through the network, we would be able to get $\underline{\mathbf{m}}^{\text{single location}}$ and verify that they satisfy the conditions in (3).

However, we have to consider not just a single location, but all possible locations $\mathbb{L}$ to give a certificate. To adapt the bound to all possible location, we pass each of the possible patches through the network, and take the worst case margin. More specifically,

$$\underline{\mathbf{m}}^{\text{es}}(\mathbb{L})_y = \min_{l \in \mathbb{L}} \underline{\mathbf{m}}^{\text{single patch}}(l)_y \quad \forall y \tag{11}$$

Similar to regular IBP training, we simply use $\underline{\mathbf{m}}^{\text{es}}(\mathbb{L})$ to calculate the cross entropy loss for training and backpropagation,

$$\text{Certificate Loss} = \text{Cross Entropy Loss}(-\underline{\mathbf{m}}^{\text{es}}(\mathbb{L}), y). \tag{12}$$

Unfortunately, the cost of producing this naïve certificate increases quadratically with image size. Consider that a CIFAR-10 image is of size 32×32, requiring over a thousand interval bounds, one for each possible patch location. To alleviate this problem, we propose two certificate training methods: *Random Patch* and *Guided Patch*, so that the number of forward passes does not scale with the dimension of the inputs.

Random Patch Certificate Training In this method, we simply select a random set of patches out of the possible patches and pass them forward. A certain level of robustness is achieved even though a very small number of random patches are selected compared to the total number of possible patches.

$$\underline{\mathbf{m}}^{\text{random patches}}(\mathbb{L})_y = \underline{\mathbf{m}}^{\text{es}}(S)_y \tag{13}$$

where S is a random subset of $\mathbb{L}$

$$\text{Random Patch Certificate Loss} = \text{Cross Entropy Loss}(-\underline{\mathbf{m}}^{\text{random patches}}(\mathbb{L}), y) \tag{14}$$

Guided Patch Certificate Training In this method, we propose using a U-net (Ronneberger et al., 2015) to predict $\underline{\mathbf{m}}^{\text{single patch}}$, and then randomly select a couple of locations based on the predicted $\underline{\mathbf{m}}^{\text{single patch}}$ so that fewer patches need to be passed forward.

Note that very few patches contribute to the worst case bound $\underline{\mathbf{m}}^{\text{es}}$ in (11). In fact, the number of patches that yield the worst case margins will be no more than the number of labels. If we know the worst-case patches beforehand, then we can simply select the few worst-case patches during training.

We propose to use U-net as the number of locations and margins is very large. For a square patch of size $n \times n$ and an image of size $m \times m$, the total number of possible locations is $(m - n + 1)^2$, and for each location the number of margins is equal to the number of possible labels.

$$\underline{\mathbf{m}}^{\text{pred}} = \text{U-net(image)} \tag{15}$$

$$\dim(\underline{\mathbf{m}}^{\text{pred}}) = (m - n + 1, m - n + 1, \text{\# of labels}). \tag{16}$$

Given the U-net prediction of $\underline{\mathbf{m}}^{\text{pred}}$, we then randomly select a single patch for each label based on the softmax of the predicted $\underline{\mathbf{m}}^{\text{pred}}$. The number of selected patches is equal to the number of labels. After these patches are passed forward, the U-net is then updated with a mean-squared-error loss between the predicted margins $\underline{\mathbf{m}}^{\text{pred}}$ and the actual margins $\underline{\mathbf{m}}^{\text{actual}}$. Note that only a few patches are selected at a time, so that the mean-squared-error only passes through the selected patches.

$$\text{U-net Loss} = \text{MSE}(\underline{\mathbf{m}}^{\text{pred}}, \underline{\mathbf{m}}^{\text{actual}}). \tag{17}$$

The network is trained with the following loss:

$$\text{Guided Patch Certificate Loss} = \text{Cross Entropy Loss}(-\underline{\mathbf{m}}^{\text{guided patches}}(\mathbb{L}), y). \tag{18}$$

5 EXPERIMENTS

In this section, we compare our certified defenses with exiting algorithms on two datasets and three model architectures of varying complexity. We consider a strong attack setting in which adversarial patches can appear anywhere in the image. Different training strategies for the certified defense are also compared, which shows a trade-off between performance and training efficiency. Finally, we evaluate the transferability of a model trained using square patches to other adversarial shapes, including shapes that cannot be bounded by the original square. The training and architectural details can be found in Appendix A.1.

5.1 COMPARISON AGAINST EXISTING DEFENSES

In this section, we study the effectiveness of our proposed IBP certified models against an adversary that is allowed to place patches anywhere in the image, even on top of the salient object. We think that if the patch is sufficiently small, and does not cover a large portion of the salient object, then the model should still classify correctly.

In the best case, our IBP certified model is able to achieve 91% certified (Table 2) with respect to a 2×2 patch ($\sim$ .5% of image pixels) adversary on MNIST. For more challenging cases, such as a 5×5 ($\sim$ 2.5% of image pixels) patch adversary on CIFAR-10, the certified adversarial accuracy is only 29.2% (Table 2). Even though these existing defenses appear to achieve better or comparable adversarial accuracy as our IBP certified model when faced with a weak adversary, when faced with a stronger adversary their adversarial accuracy dropped to levels below our certified accuracy for all cases that we analyzed.

When evaluating existing defenses, we only report cases where the adversarial accuracy against a weaker adversary is a least reasonably good. We do not report some experiments as LGS and DW perform so poorly for the comparison to be meaningful. LGS and DW are highly dependant on hyperparameters to work effectively against naive attacks, and yet neither Naseer et al. (2019) nor Hayes (2018) proposed a way to learn these hyperparameters. By trial and error, we were able to increase the adversarial accuracy against a weaker adversary for some settings, but not all. In addition, we also notice a peculiar feature of DW: in order to increase the adversarial accuracy, the clean accuracy degrades so much that it is even lower than the empirical adversarial accuracy. That is because DW always removes a patch from the prediction. When an adversarial patch is detected, it is likely to be removed enabling correct prediction. On the other hand, when there are no adversarial patches, DW ends up removing actual salient information resulting in lower clean accuracy.

Here we did not compare our results with adversarial training, because even though it produces some of the most adversarially robust models, it is still empirical robust accuracy, and could still be decreased further with stronger attacks. For example, Wang et al. (2019) proposed a stronger attack that could find 47% more adversarial examples compared to gradient based method. Further, adversarial training on all possible patches would be even more expensive compared to certificate training, and is slightly beyond our computational budget.

5.2 COMPARISON OF TRAINING STRATEGIES

We find that all-patch certificate training performs the best with respect to certified accuracy, but takes the longest. In contrast, random-patch certificate training is fast, but trades off robustness for

Table 2: Comparison of our IBP certified patch defense against existing defenses. Empirical adversarial accuracy is calculated for random 400 images in both datasets. All results are averaged over three different models.

Dataset	Patch Size	Adversary	Defense	Clean Accuracy	Empirical Adversarial Accuracy	Certified Accuracy
MNIST	2×2	IFGSM	None	98.4%	80.1%	-
	2×2	IFGSM	LGS	97.4%	90.0%	-
	2×2	IFGSM + LGS	LGS	97.4%	60.7%	-
	2×2	-	IBP	98.5%	-	91.3%
	5×5	IFGSM	None	98.5%	3.3%	-
	5×5	-	IBP	92.9%	-	60.7%
CIFAR	2×2	IFGSM	None	66.3%	25.4%	-
	2×2	IFGSM	LGS	64.9%	31.3%	-
	2×2	IFGSM + LGS	LGS	64.9%	24.2%	-
	2×2	IFGSM	DW	47.1%	43.3%	-
	2×2	IFGSM + DW	DW	47.1%	20.2%	-
	2×2	-	IBP	47.2%	-	36.7%
	5×5	IFGSM	None	66.5%	0.4%	-
	5×5	IFGSM	LGS	51.2%	22.11%	-
	5×5	IFGSM + LGS	LGS	51.2%	0.5%	-
	5×5	IFGSM	DW	45.3%	59.3%	-
	5×5	IFGSM + DW	DW	45.3%	15.6%	-
	5×5	-	IBP	35.7%	-	29.2%

speed. Finally, guided-patch certificate training outperforms random-patch certificate training by only a slim margin, but tends to do better as the task becomes harder while being moderately expensive.

In Table 3, we see that all-patch certificate training significantly outperforms both random-patch certificate training and guided-patch certificate training in terms of certified accuracy, outperforming the second best certified defenses in each task by .9% (MNIST, 2×2), 6.5% (MNIST, 5×5), 2.2% (CIFAR-10, 2×2), and 3.5% (CIFAR-10, 5×5). However, all-patch certificate training is very expensive, taking on average 2 to 4 times longer than guided-patch certificate training and over 10 times longer than random-patch certificate training.

Random-patch certificate training works surprisingly well, achieving almost the same certified accuracy as all-patch certificate training on (MNIST, 2×2) when only 5 random patches are used during training (Table 3). However, the gap remains, especially on harder tasks. With the exception of the (MNIST, 2×2) experiment, using more patches for training gives better results up to a point before we observe diminished returns.

Guided-patch certificate training is computationally expensive, in large part because the U-net architecture used is way too big compared to the trained model. However, given the 10 patches picked, guided-patch certificate training consistently outperforms random-patch certificate training, indicating that the U-net is learning how to predict the minimum margins $\underline{\mathbf{m}}$. Even though guided-patch certificate training does not outperform random-patch certificate training by much, the gap increases as the task becomes harder, giving potentially more merit to guided-patch certificate training in such cases.

5.3 TRANSFERABILITY TO PATCHES OF DIFFERENT SHAPES

Since real-world adversarial patches may not always be square, the robust transferability of the model to shapes other than the square is important.

Therefore, we evaluate the robustness of the square-patch-trained model to adversarial patches of different shapes while fixing the number of pixels. In all these experiments, we evaluate the certified accuracy for our largest model, on both MNIST and CIFAR datasets. We evaluate the transferability to various shapes including rectangle, line, parallelogram, and diamond. With the exception of rectangles, all the shapes have the exact same pixel count as the patches used for training. For

Table 3: Trade-off between certified accuracy and training time for different strategies. The numbers next to training strategies indicate the number of patches used for estimating the lower bound during training. Most training times are measured on a single 2080Ti GPU, with the exception of all-patch training which is run on four 2080Ti GPUs. For that specific case, the training time is multiplied by 4 for fair comparison. See Appendix A.3 for more detailed statistics.

		2×2			5×5		
Dataset	Training Strategy	Clean Accuracy	Certified Accuracy	Training Time(h)	Clean Accuracy	Certified Accuracy	Training Time(h)
MNIST	All Patch	98.5%	91.3%	14.2	92.9%	60.8%	11.2
	Random(1)	98.5%	82.6%	0.1	97.1%	27.8%	0.4
	Random(5)	98.3%	90.4%	0.7	93.6%	47.6%	0.7
	Random(10)	98.6%	88.1%	0.8	95.5%	51.9%	0.8
	Guided(10)	98.6%	88.7%	6.3	94.9%	54.2%	6.3
CIFAR	All Patch	47.2%	40.2%	50	35.7%	29.2%	41.3
	Random(1)	54.1%	22.8%	1.5	43.4%	7.6%	1.2
	Random(5)	52.9%	32.9%	3.7	39.9%	18.9%	3.6
	Random(10)	52.1%	36.7%	4.1	38.4%	23.8%	3.7
	Guided(10)	52.7%	38.0%	10.8	37.8%	25.7%	3.7

rectangles, we use multiple choices of width and length, obtaining some combinations with slightly more pixels, and the worst accuracy is reported in Table 4. The exact shapes used can be found in Appendix A.2.

The certified accuracy of our models generalize surprisingly well to other shapes, losing no more than than 1% in most cases for MNIST and no more than 8% for CIFAR-10 (Table 4). The largest degradation of accuracy happens for rectangles, and it is mostly because the rectangle considered has more pixels compared to the square. Surprisingly, the certificate even generalizes to a straight line, even though the model was never trained to be robust to lines. In the case of MNIST, the certified accuracy even improves when transferred to lines across all pixel counts.

Table 4: Certified accuracy for square-patch trained model for different shapes

Dataset	Pixel Count	Square	Rectangle	Line	Diamond	Parallelogram
MNIST	4	91.5%	-	91.8%	91.4%	91.8%
MNIST	16	73.8%	67.5%	78.5%	74.7%	75.4%
MNIST	25	61.4%	55.9%	63.7%	61.3%	62.1%
CIFAR	4	43.2%	-	36.1%	36.2%	42.9%
CIFAR	16	34.1%	31.0%	26.2%	32.1%	34.0%
CIFAR	25	26.6%	22.1%	25.6%	24.3%	25.9%

6 Conclusion and Future Work

In this paper, we established the vulnerability of known defenses against adversarial patch attacks. To remedy that, we proposed the first certified defense against patch attacks based on interval bound propagation. We demonstrated the effectiveness of our certified defense on two datasets, and proposed two training strategies to speed it up while trading-off efficiency and robustness. Finally, we considered the robust transferability of our trained certified models to different shapes obtaining surprisingly good generalization. In its current form, the proposed certified defense is unlikely to scale to ImageNet, and we hope the presented experiments will encourage further work along this direction.

References

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *arXiv preprint arXiv:1712.02779*, 2017.

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

Jamie Hayes. On visible adversarial perturbations & digital watermarking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1597–1604, 2018.

Danny Karmon, Daniel Zoran, and Yoav Goldberg. Lavan: Localized and visible adversarial noise. *arXiv preprint arXiv:1801.02608*, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Cassidy Laidlaw and Soheil Feizi. Functional adversarial attacks, 2019.

Mark Lee and Zico Kolter. On physical adversarial patches for object detection, 2019.

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.

Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pp. 3575–3583, 2018.

Muzammal Naseer, Salman Khan, and Fatih Porikli. Local gradients smoothing: Defense against localized adversarial attacks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1300–1307. IEEE, 2019.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Adversarial generative nets: Neural network attacks on state-of-the-art face recognition. *arXiv preprint arXiv:1801.00349*, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations. *arXiv preprint arXiv:1904.13000*, 2019.

Shiqi Wang, Yizheng Chen, Ahmed Abdou, and Suman Jana. Enhancing gradient-based attacks with symbolic intervals, 2019.

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, 2018.

Eric Wong and J Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. *arXiv preprint arXiv:1711.00851*, 2017.

Eric Wong, Frank R Schmidt, and J Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. *arXiv preprint arXiv:1902.07906*, 2019.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. *arXiv preprint arXiv:1906.06316*, 2019.

A Appendix

A.1 Experimental Settings and Network Structure

We evaluate the proposed certified patch defense on three neural networks: a multilayer perceptron (MLP) with one 255-neuron hidden layer, and two convolutional neural networks (CNN) with different depths. The small CNN has two convolutional layers (kernel size 4, stride 2) of 4 and 8 output channels each, and a fully connected layer with 256 neurons. The large CNN has four convolutional layers with kernel size (3, 4, 3, 4), stride (1, 2, 1, 2), output channels (4, 4, 8 ,8), and two fully connected layer with 256 neurons. We run experiments on two datasets, MNIST and CIFAR10, with two different patch sizes 2×2 and 5×5. For all experiments, we are using Adam (Kingma & Ba, 2014) with a learning rate of $5e-4$ for MNIST and $1e-3$ for CIFAR10, and with no weight decay. We also adopt a warm-up schedule in all experiments like (Zhang et al., 2019), where the input interval bounds start at zero and grow to [-1,1] after 61/121 epochs for MNIST/CIFAR10 respectively. We train the models for a total of 100/200 epochs for MNIST/CIFAR10, where in the first 61/121 epochs the learning rate is fixed and in the following epochs, we reduce the learning rate by one half every 10 epochs.

A.2 Sample shapes for generalization experiments

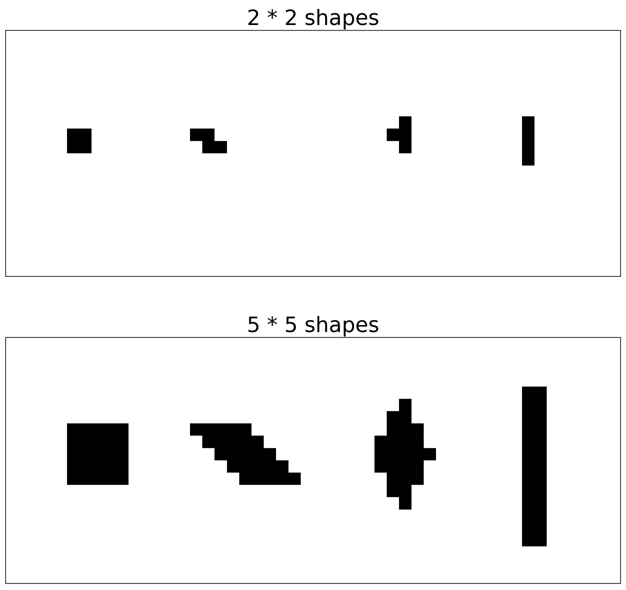

Figure 1: Examples of shapes with pixels number 4 and 25. From left to right are square, parallelogram, diamond and rectangle (line) respectively.

A.3 Detailed Statistics on Training Strategies

Dataset	Training Strategies	Patch Sizes	Model Architecture	Clean Accuracy	Certified Accuracy
MNIST	All Patch	2*2	fully connected (255,10)	98.47%	91.34%
		2*2	2 layer convolution	98.50%	91.54%
		2*2	4 layer convolution	98.59%	91.06%
	Random (1)	2*2	fully connected (255,10)	98.55%	84.03%
		2*2	2 layer convolution	98.56%	82.87%
		2*2	4 layer convolution	98.50%	80.84%
	Random (5)	2*2	fully connected (255,10)	98.51%	90.80%
		2*2	2 layer convolution	98.39%	89.01%
		2*2	4 layer convolution	98.08%	91.34%
	Random (10)	2*2	fully connected (255,10)	98.70%	88.66%
		2*2	2 layer convolution	98.61%	87.49%
		2*2	4 layer convolution	98.46%	88.01%
	Guided Patch (10)	2*2	fully connected (255,10)	98.59%	89.03%
		2*2	2 layer convolution	98.49%	88.36%
		2*2	4 layer convolution	98.65%	88.66%
CIFAR	All Patch	2*2	fully connected (255,10)	49.32%	36.57%
		2*2	2 layer convolution	41.15%	41.15%
		2*2	4 layer convolution	51.23%	42.91%
	Random (1)	2*2	fully connected (255,10)	52.70%	6.94%
		2*2	2 layer convolution	55.30%	29.17%
		2*2	4 layer convolution	54.43%	32.35%
	Random (5)	2*2	fully connected (255,10)	51.65%	23.85%
		2*2	2 layer convolution	54.25%	36.87%
		2*2	4 layer convolution	52.71%	37.95%
	Random (10)	2*2	fully connected (255,10)	50.61%	31.62%
		2*2	2 layer convolution	53.09%	39.23%
		2*2	4 layer convolution	52.46%	39.26%
	Guided Patch (10)	2*2	fully connected (255,10)	51.20%	32.24%
		2*2	2 layer convolution	53.71%	40.81%
		2*2	Model Architecture	53.26%	40.96%

Table 5: Detailed statistics for the comparison of training strategies - 2*2

Dataset	Training Strategies	Patch Sizes	Model Architecture	Clean Accuracy	Certified Accuracy
MNIST	All Patch	5*5	fully connected (255,10)	94.08%	62.30%
		5*5	2 layer convolution	92.52%	58.31%
		5*5	4 layer convolution	92.30%	61.44%
	Random (1)	5*5	fully connected (255,10)	97.56%	29.48%
		5*5	2 layer convolution	97.33%	25.39%
		5*5	4 layer convolution	96.53%	28.49%
	Random (5)	5*5	fully connected (255,10)	94.78%	51.62%
		5*5	2 layer convolution	93.48%	45.42%
		5*5	4 layer convolution	92.38%	45.94%
	Random (10)	5*5	fully connected (255,10)	96.17%	53.56%
		5*5	2 layer convolution	95.58%	51.08%
		5*5	4 layer convolution	94.69%	50.92%
	Guided Patch (10)	5*5	fully connected (255,10)	95.84%	56.48%
		5*5	2 layer convolution	94.95%	54.28%
		5*5	4 layer convolution	93.93%	54.08%
CIFAR	All Patch	5*5	fully connected (255,10)	33.81%	30.93%
		5*5	2 layer convolution	35.46%	30.09%
		5*5	4 layer convolution	37.79%	26.58%
	Random (1)	5*5	fully connected (255,10)	41.45%	0.69%
		5*5	2 layer convolution	45.88%	8.74%
		5*5	4 layer convolution	42.71%	13.24%
	Random (5)	5*5	fully connected (255,10)	37.33%	13.25%
		5*5	2 layer convolution	41.47%	23.12%
		5*5	4 layer convolution	40.74%	20.36%
	Random (10)	5*5	fully connected (255,10)	35.20%	21.99%
		5*5	2 layer convolution	41.00%	27.76%
		5*5	4 layer convolution	38.87%	21.60%
	Guided Patch (10)	5*5	fully connected (255,10)	35.35%	24.63%
		5*5	2 layer convolution	40.84%	27.83%
		5*5	4 layer convolution	37.34%	24.63%

Table 6: Detailed statistics for the comparison of training strategies - 5*5