On Visible Adversarial Perturbations & Digital Watermarking

Jamie Hayes
University College London
j.hayes@cs.ucl.ac.uk

Abstract

Given a machine learning model, adversarial perturbations transform images such that the model's output is classified as an attacker chosen class. Most research in this area has focused on adversarial perturbations that are imperceptible to the human eye. However, recent work has considered attacks that are perceptible but localized to a small region of the image. Under this threat model, we discuss both defenses that remove such adversarial perturbations, and attacks that can bypass these defenses.

1. Introduction

Neural networks are known to be vulnerable to adversarial examples: perturbations that when combined with data inputs, cause intentional misclassifications. Most research has focused on studying small ℓ_p perturbations that remain visually imperceptible when applied to an input. FGSM [13], L-BFGS [35], DeepFool [25], Carlini$_{\{\ell_2,\ell_\infty\}}$ [8], PGD [24] and EAD [9] are all examples of attacks that modify each pixel in an image by a small amount, while attacks such as JSMA [26] and Carlini$_{\ell_0}$ [8], perturb a small subset of pixels in an image.

Recent work by Brown et al. [5] and Karmon et al. [20] have studied adversarial examples under a new threat model - an attacker that crafts perturbations that *are not* bounded by an ϵ value but *are* bounded to a small region or location in the image. Clearly, this removes the visually imperceptible property that most other attacks possess. However, as noted by both Brown et al. [5] and Karmon et al. [20], attackers may not be concerned with this property; ML models are often not validated by humans, and thus visual indistinguishability between adversarial and clean examples may not a highly sought after property of the attack. Furthermore, humans may not recognize the adversarial examples as adversarial, viewing the small visual perturbations as natural corruption or noise within the image.

Brown et al. [5] introduced an attack, Adversarial Patch, that finds a "universal" adversarial perturbation that can be applied to any image and cause a misclassification of the adversaries choice. They show that it is possible to create a small adversarial "patch" that is invariant to location (on the image) and rotation. Furthermore, they show the attack transfers to the physical world - it is possible to construct a patch, print it out and retain its adversarial properties.

In parallel, Karmon et al. [20] introduced an attack, LaVAN, that creates localized and visible adversarial perturbations. Broadly, both attacks are equivalent; Adversarial Patch is trained under a pre-processing stage that is applied to the patch, which outputs a new rotated and scaled patch, which is then applied to the image at a randomly chosen location. LaVAN is similar, however the location, rotation and scale is randomly chosen once at the beginning of the attack and is then left unchanged. The authors also discuss transferability properties of their method, and show that applying a similar pre-processing stage at each iteration of the attack can create perturbations that are invariant to location or rotation.

In this work, we discuss the difficulties in defending against such attacks. Because both Adversarial Patch and LaVAN radically change small regions of an image, defending against such attacks is very closely aligned to the problem of inpainting and watermark removal. We may think of the perturbation as a watermark placed in the image by the attacker. The goal of a robust defense is then to remove this watermark, or at least render it as safe to the target model. Quiring et al. [28] have also drawn connections between the research fields of adversarial examples and digital watermarking, in the context of traditional ϵ-ball attacks.

We show that localized and visible adversarial perturbations can be defended against using simple principles from inpainting research. Our intuition is that the influence signal of pixels within an image containing localized and visible adversarial perturbations are dominated by the perturbations, and thus we can use this dominating factor to detect and defend. We show an attacker is unable to modify their attack to successfully bypass the defense if they are confined to perturb only a small subset of pixels. We then introduce a trivial attack that can break this defense, highlighting the

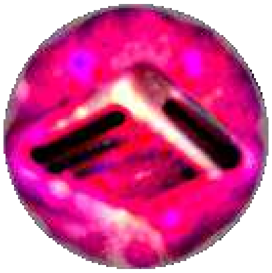


Figure 1: Toaster patch for Inception-V3.

need to consider a variety of potential attacks if deploying models to safety critical tasks.

2. Localized and Visible Adversarial Perturbations

Here we give an overview of two recently proposed localized and visible adversarial perturbation attacks, Adversarial Patch and LaVAN.

2.1. Adversarial Patch

Adversarial Patch creates a universal "patch" that can be applied to any image x in a dataset X, and cause a targeted misclassification in a model f, regardless of the scale, orientation or location of the patch.

Given a patch p, an image $x \in X$, a target class $\hat{y}$, a sampled location in the location space of images $l \in L$, and a transformation over a set of transformations $t \in T$, an operator $A(p, x, l, t)$ is defined that re-scales and rotates a patch and is then applied to an image at a location l. The attacker updates the patch iteratively by optimizing the objective function:

$$\hat{p} = \arg\max_{p} \mathbb{E}_{x \sim X, t \sim T, l \sim L}[\log \Pr(\hat{y}|A(p, x, l, t)]$$

By optimizing over the Expectation over Transformation [2], a patch is found that remains adversarial regardless of scale, location or orientation.

Experimental results are reported on a patch crafted over an ensemble of ImageNet classifiers (Inception-V3, ResNet-50, Xception, VGG-16, and VGG-19), single models, and in black-box attacks - where the patch is trained on four models and results reported on the fifth. With a patch size of 10% of the image, over 90% of images were successfully misclassified as the patch target class. We re-implemented the authors work and were successfully in replicating these results, an example of an "Adversarial Patch" is shown in Figure 1.

2.2. LaVAN

LaVAN takes a different approach to Adversarial Patch, computing an adversarial perturbation that is dependent on the chosen location and the image under attack [1]. The LaVAN attack takes as input: a confidence threshold κ, a mask $m \in \{0, 1\}^n$, an image $x \in X$, a model f, and a target class $\hat{y}$. At each iteration, the attacker updates the perturbation by the following method:

1. Apply the current perturbation to the image:

$$(1 - m) \odot x + m \odot p,$$

where $\odot$ is element-wise multiplication.

2. Find the target class output, $f(x)|_{\hat{y}}$ and the clean image source classification $f(x)|_y$.

3. Update the perturbation by:

$$-\epsilon \cdot (\frac{\partial f(x)|_{\hat{y}}}{\partial x} - \frac{\partial f(x)|_y}{\partial x})$$

The attack is terminated when $f(x)|_{\hat{y}} \geq \kappa$.

The authors reported that when attacking the Inception-V3 model on 100 ImageNet images, and restricting perturbations to be 2% of the total image size, they were successful at causing a targeted misclassification in 79% of 110 configurations tested, when κ is equivalent to 90%. We re-implemented the authors work and were successfully in replicating these results. However, we found attacks were more successful when removing the ϵ multiplier in step (3) since gradient values are already orders of magnitude smaller than the acceptable input range of the classifier, thus in further experiments we update the perturbation omitting the ϵ coefficient:

$$-(\frac{\partial f(x)|_{\hat{y}}}{\partial x} - \frac{\partial f(x)|_y}{\partial x})$$

The authors also stop the attack after 10,000 iterations or when the confidence threshold is met. We introduce an early stopping criteria - terminating the attack if the objective function fails to decrease. We found this improved the speed of the attack by a factor of 10-20X without harming success.

Finally, LaVAN is not designed to create a universal "patch" and thus, we were able to compute both non-targeted and targeted attacks. Non-targeted attacks are equivalent to targeted attacks, with the target class chosen to be the second most likely class and with no specified confidence threshold.

3. Threat Model

We aim to defend against an attacker who has white-box access to a target model, f, and can manipulate a subset of pixels in an image x in a dataset X. We restrict the attacker

[1] The authors also discuss methods to make their approach transferable across images, however we omit a full analysis of that here. The method is almost identical to Adversarial Patch.

to only manipulate pixels within a small region of the image - the attacker may choose the location of the area, but may not perturb pixels outside of this location.

We consider two attack settings for our defense: (1) the attacker is not aware that a defense is being used (2) the attacker is aware of the defense and crafts adversarial perturbations to avoid detection. In (1), we report results from an attack that pre-constructs adversarial examples against an undefended model, and apply these inputs to the defended model, while in (2) the attack has access to the defense while constructing the perturbations. This is similar to Kerchoff's principle in Cryptography, which states that a scheme should be secure even if an attacker has full knowledge of the scheme in use, but is not aware of the key, or in our case, the hyperparameters of the defense.

4. Defenses

We wish to both detect and remove localized and visible adversarial perturbations, simultaneously. The problem of removing adversarial perturbations is closely aligned with the problem of watermark removal, or sometimes referred to as inpainting. In a classical inpainting problem, an image has been corrupted through scratches or random noise and the task is to restore the image and remove such noise. This is exactly the same problem as defending against adversarial perturbations - we have a corrupted copy of an image and wish to remove the noise and restore the image. Previous work [3, 4, 6, 10, 11, 12, 14, 15, 16, 17, 18, 19, 21, 22, 23, 27, 29, 30, 31, 33, 38, 40] has attempted to defend against classical adversarial examples, where the adversarial noise is distributed over the entire image and is designed to remain visually imperceptible. Unfortunately, these techniques have been shown to degrade under attacks with knowledge of the underlying defense [1, 7, 37]. However, our threat model allows us to revisit proposed noise reduction defenses such as inpainting since the adversarial perturbation is confined to a small region of the image and provides a larger signal than in non-localized attacks.

The inpainting problem can generally be classified into two categories:

> **Non-blind.** In non-blind image inpainting, the reconstruction process is given the location of the areas to be inpainted along with the corrupted image.
>
> **Blind.** In blind image inpainting, the reconstruction process is given only the noisy image. The areas to be inpainted must be discovered before inpainting can begin. Blind inpainting is a strictly more difficult problem than non-blind inpainting.

While we anticipate the blind setting to be the most realistic, we study defenses in both settings.

In both non-blind and blind settings, our aim is to construct a pre-processing function $h : \mathbb{R}^{w\times h\times c} \to \mathbb{R}^{w\times h\times c}$ such that $\arg\max_i f(h(x+p))_i = \arg\max_i f(x)_i$, where p is a localized and visible adversarial perturbation, $x \in X$ and f is the classifier under attack.

4.1. Non-blind

In the non-blind setting, the pre-processing step h, has access to a mask $m \in \{0,1\}^n$ that contains the location of noise to be inpainted in addition to the adversarial example $x+p$. There are many options for traditional non-blind watermark removal, however our primary criteria for this preprocessing step is that it is fast, and renders the adversarial perturbation ineffective. Although original image fidelity is certainly aligned with these criteria, it is not exactly the same. We thus choose an inpainting method, developed by Alexandru Telea [36], that is optimized for speed rather than accurate inpainting.

The method works as follows: Define the region to be inpainted as Ω and the boundary of the region as $\partial\Omega$. Given some point u on $\partial\Omega$, take the ϵ-ball of the known image around u, $B_\epsilon(u) = \{v \in x \mid \|u-v\|_p < \epsilon, v \notin \Omega\}$. For small ϵ, we consider a first-order approximation $I_v(u)$ of the image at point u, given the image $I(v)$ and gradient $\nabla I(v)$ at values of point v:

$$I_v(u) = I(v) + \nabla I(v)(u-v)$$

The point u is inpainted by summing all points in $B_\epsilon(u)$, weighted by a normalized function $w(u,v)$:

$$I(u) = \frac{\sum_{n=1}^{|B_\epsilon(u)|} w(u,n)(I(n) + \nabla I(n)(u-n))}{\sum_{n=1}^{|B_\epsilon(u)|} w(u,v)} \tag{✷}$$

To inpaint all of Ω, (✷) is iteratively applied to all unknown points on $\partial\Omega$ in increasing distance from the starting position, and then advances into Ω until $\Omega = \emptyset$.

4.2. Blind

If the defense does not have access to a mask giving the location of the perturbation, it must first find the areas that contain the adversarial perturbations, before sanitization can begin. To detect localized and visible adversarial perturbations, we note that for such images, their saliency maps generally have very dense clusters around the location of the perturbation. This is because the classification of the image is almost entirely influenced solely by this small area, while in natural images, pixels that influence classification are more sparsely distributed. We use this observation to

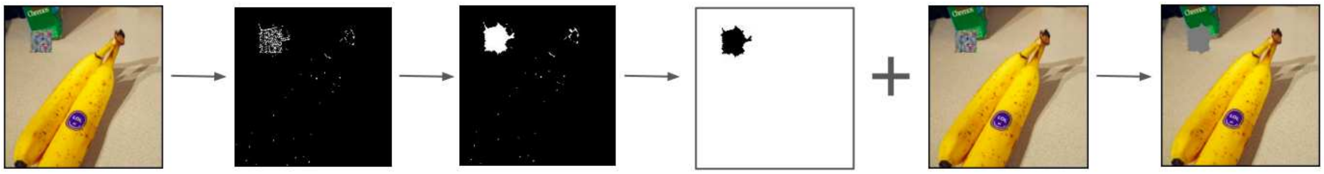


Figure 2: A flow diagram of the steps to defend against a localized and visible adversarial perturbation in a blind setting. We first find the saliency map of the image. The following two steps constructs a mask that is applied to the adversarial image, blocking the adversarial perturbation.

Algorithm 1: Pseudocode for blind defense.

```
Input  : image $x$,
         classifier $f$,
         predicted label $\hat{y} = \arg\max_{y} f(x)$,
         pixel threshold $\mu \in \mathbb{R}$,
         contour area threshold $\phi \in \mathbb{R}_{>0}$
Output : image $x'$
$sal \leftarrow$ Get saliency map of $x$ with respect to $\hat{y}$;
for $p \in sal$ do
    $p = \begin{cases} 0, & \text{if } p > \mu \\ 1, & \text{otherwise} \end{cases}$
end
$sal' \leftarrow ((sal \oplus B) \ominus B)^n$, where $n \in \mathbb{Z}^+$,
    $B := \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$
Find contiguous contours in $sal'$. If contour area
    $< \phi$, zero out.
$m \leftarrow ((sal' \oplus B) \ominus B)^n$
for $p \in m$ do
    $p = \begin{cases} 0, & \text{if } p = 1 \\ 1, & \text{otherwise} \end{cases}$
end
$x' \leftarrow x \odot m$
return $x'$
```

detect unnaturally dense regions that contribute to classification, constructing a mask that covers these regions and so remove their influence on classification.

To create an overview of which areas of the image are influencing classification, we construct a saliency map of the image using the guided backpropagation method [34]. Simonyan et al. [32] use the gradient of the output class with respect to the input image to construct a saliency map. When backpropagating the influence signal through ReLU activation units, the signal is zero'd if the input in the forward pass was below zero. In parallel, Zeiler & Fergus [39] construct a similar saliency map but zero out the signal if it is negative in the backwards pass, ignoring any information through the ReLU unit in the forward pass. Guided backpropagation combines both of these approaches, zeroing the influence signal if either the forward or backward pass through a ReLU unit is negative.

Once a saliency map for the input has been found, we use a combination of erosion and dilation to remove small "holes" [2]. Finally, we find the contour area of positive regions within the updated saliency map, and if the contour area is below a threshold, we zero out this area. Finally, we use the remaining positive regions of the saliency map as locations to mask the adversarial image. The pseudo-code of the defense is given in Algorithm 1, and a flow diagram from adversarial example to benign example is shown in Figure 2.

5. Experimental Results

We compare our non-blind and blind defenses across a number of ImageNet models: VGG-19, ResNet-101 and Inception-V3, and the two proposed attacks: Adversarial Patch and LaVAN. We re-implemented both attacks and verified we could replicate their results. All results are reported on 400 randomly chosen images in the ImageNet validation set.

For the LaVAN attack, we craft both non-targeted and targeted adversarial examples. For a non-targeted attack, we terminate when any misclassification is found, however for targeted attacks we specify a target class and a target threshold confidence score (κ) that must be met before the attack is stopped. Since Adversarial Patch finds a universal adversarial perturbation, we only study targeted attacks. For each image under a LaVAN targeted attack, we randomly chose a target class, whereas for Adversarial Patch we chose a target of the toaster class and the golf ball class. A robust defense must both remove adversarial perturbations from adversarial images and not affect the classification of clean

[2] Erosion is denoted by $\oplus$ and is defined by:

$$U \oplus V := \bigcup_{v \in V} U_v$$

Dilation is denoted by $\ominus$ and is defined by:

$$U \ominus V := \bigcap_{v \in V} U_{-v}$$

Where $U, V \in \mathbb{Z}^{\{0,1\}}$ and U_v denotes the translation of U by v.

Table 1: Results for non-blind defense.

				VGG-19			ResNet-101			Inception-V3		
	Attack Type	κ	**Perturbation (%)**	**Advesarial Accuracy**	**Reconstructed Accuracy**	**Time (s)**	**Advesarial Accuracy**	**Reconstructed Accuracy**	**Time (s)**	**Advesarial Accuracy**	**Reconstructed Accuracy**	**Time (s)**
LaVAN	*Non-targeted*	-	2	0.970	0.990	0.006	0.875	0.986	0.006	0.818	0.985	0.011
			5	0.995	0.935	0.010	0.988	0.947	0.010	0.983	0.959	0.016
			10	1.000	0.880	0.015	1.000	0.902	0.015	0.998	0.910	0.025
			25	1.000	0.685	0.030	1.000	0.690	0.030	1.000	0.715	0.053
	Targeted	0.90	2	0.714	0.992	0.007	0.675	0.981	0.006	0.575	0.978	0.011
			5	0.882	0.934	0.010	0.940	0.944	0.010	0.907	0.956	0.016
			10	0.998	0.880	0.015	0.988	0.901	0.015	0.990	0.909	0.025
			25	1.000	0.685	0.031	1.000	0.690	0.031	1.000	0.715	0.053
		0.99	2	0.543	0.992	0.007	0.675	0.981	0.007	0.573	0.983	0.010
			5	0.889	0.934	0.010	0.938	0.947	0.010	0.900	0.956	0.016
			10	0.998	0.880	0.015	0.993	0.902	0.015	0.985	0.909	0.026
			25	1.000	0.685	0.030	1.000	0.690	0.030	1.000	0.715	0.053
Adversarial Patch (Toaster)	*Targeted*	0.90	2	0.551	0.971	0.005	0.296	0.923	0.006	0.198	1.000	0.009
			5	0.684	0.921	0.009	0.556	0.910	0.013	0.678	0.953	0.016
			10	0.910	0.887	0.017	0.882	0.751	0.018	0.804	0.896	0.020
			25	1.000	0.662	0.033	1.000	0.589	0.029	1.000	0.606	0.040
		0.99	2	0.210	0.971	0.007	0.138	0.875	0.006	0.099	1.000	0.010
			5	0.652	0.926	0.010	0.436	0.923	0.015	0.662	0.950	0.019
			10	0.765	0.893	0.020	0.690	0.907	0.022	0.872	0.893	0.031
			25	0.934	0.701	0.041	0.901	0.454	0.045	0.918	0.664	0.060
Adversarial Patch (Golf Ball)	*Targeted*	0.90	2	0.384	0.970	0.005	0.226	0.923	0.006	0.298	0.978	0.010
			5	0.614	0.881	0.009	0.446	0.853	0.053	0.568	0.900	0.018
			10	0.933	0.809	0.021	0.910	0.781	0.019	0.844	0.922	0.028
			25	1.000	0.600	0.032	1.000	0.442	0.031	1.000	0.589	0.045
		0.99	2	0.301	0.932	0.009	0.111	0.772	0.006	0.109	1.000	0.012
			5	0.714	0.902	0.018	0.331	0.771	0.014	0.699	0.901	0.019
			10	0.785	0.838	0.026	0.678	0.619	0.029	0.888	0.800	0.031
			25	0.935	0.709	0.044	0.943	0.578	0.045	0.957	0.445	0.060

Table 2: Results for blind defense on VGG-19.

	Attack Type	κ	**Perturbation (%)**	**Adversarial Accuracy**	**Reconstructed Accuracy**	**Time (s)**
LaVAN	*Non-targeted*	-	2	0.995	0.630	0.314
			5	1.000	0.403	0.313
			10	1.000	0.301	0.311
			25	1.000	0.136	0.309
	Targeted	0.90	2	0.711	0.980	0.190
			5	0.827	0.934	0.248
			10	0.968	0.672	0.340
			25	0.995	0.153	0.338
		0.99	2	0.453	0.935	0.347
			5	0.805	0.884	0.327
			10	0.968	0.639	0.244
			25	0.995	0.129	0.195
Adversarial Patch (Toaster)	*Targeted*	0.90	2	0.551	0.959	0.382
			5	0.684	0.761	0.371
			10	0.910	0.538	0.389
			25	1.000	0.090	0.299
		0.99	2	0.210	0.957	0.297
			5	0.652	0.881	0.365
			10	0.765	0.422	0.320
			25	0.934	0.062	0.041

images. Our non-blind defense leaves clean images unaffected since the defense has access to a mask with locations of perturbations, which will be empty in this case. Our blind defense, currently misclassifies 12% of clean images. For example, this reduces VGG-19 top-1 accuracy from 74.2% to 65.4%, which is approximately equivalent to VGG-11 top-1 accuracy. Our blind defense may also be combined with other detection mechanisms in the future, a problem that is strictly easier than simultaneous detection and removal.

In both non-blind and blind experiments we report: *attack type* - targeted or non-targeted, κ, *adversarial accuracy* - the fraction of adversarial images that were successful in fooling the classifier, *reconstructed accuracy* - the fraction of successfully defended adversarial images, and *time* - the average time to remove the adversarial perturbation from the image.

5.1. Non-blind

Table 1 gives the adversarial success of both Adversarial Patch and LaVAN, and the reconstruction success of our non-blind defense against the attacks. For both attacks we vary the area that the localized adversarial perturbation can cover from 2% to 25% of the image.

Both attacks are overwhelmingly successful even for very small areas, however this also results in an easier reconstruction task since fewer pixels need to be inpainted. However, we can still successfully remove the adversarial perturbations in over 68% of images when the perturbation covers a quarter of the image. The average time taken to reconstruct the image is also negligible, e.g. 6ms for a perturbation size of 2% on VGG-19.

5.2. Blind

Table 2 gives the adversarial success of both LaVAN and Adversarial Patch, and the reconstruction success of our blind defense against the attacks. For both attacks we vary the size of the localized adversarial perturbation from

Table 3: Results for improved LaVAN attack on blind defense on VGG-19.

κ	Bounding Area (%)	Perturbation (%)	Advesarial Accuracy	Reconstructed Accuracy	Time (s)
0.90	2	2	0.711	0.980	0.190
	5	2	0.708	0.912	0.188
	10	2	0.723	0.733	0.192
	25	2	0.741	0.299	0.199
	100	2	0.805	0.046	0.173

2%-25% of the image.

Compared to the non-blind setting, the defense is weaker and slower, however we still achieve strong results for small perturbations; successfully removing the adversarial perturbations in 95% of targeted LaVAN and Adversarial Patch attacks when the perturbation covers 2% of the image. For the LaVAN non-targeted attack we successfully remove 63% of adversarial examples for a 2% perturbation. Interestingly, since our defense relies upon strong gradient signals, it is more difficult to defend against weaker attacks such as the non-targeted attacks that terminate as soon as a misclassification is found, while our defense is resilient against strong attacks. As reported by Karmon et al. [20], we found Adversarial Patch generally performs worse than LaVAN in terms of adversarial example success. This is because Brown et al. [5] focused on constructing perturbations that succeed against black-box models and that can be physically printed out.

5.3. Bypassing the defense

The evaluated defenses have so far assumed that the attacker did not have direct access to the defense when constructing the adversarial perturbations. However, recent work [1, 7, 37] have shown many defenses can be trivially bypassed if incorporated into the attack pipeline. Thus, we now evaluate an attack that constructs perturbations against a model using the blind defense. However, we do not reveal the hyperparameters used in the defense, such as the contour threshold area. Instead, they are empirically estimated by the attacker during the attack. We show that an attacker can estimate sensitive defense attributes and thus bypass the defense.

An attacker who simply applies the same targeted attack to the defended model will not be successful in crafting adversarial perturbations, given some confidence threshold κ. As soon as the perturbation triggers the contour area threshold, the perturbation will be masked before being input into the classifier, and so no further information will be provided from which to optimize the attack. Experimentally, we found this triggering to often occur well before the image is classified as the adversaries target class. Indeed, as we can see from Table 2, 63% of non-targeted adversarial examples are successfully defended against for an area of 2% of the image, while the difference between most confident and second most confident class is very small.

To bypass the defense, the locations of adversarial pixels must be sparse enough to bypass the erosion and dilation processes, and so fall below the contour area threshold. We define a pre-processing function: $g : \mathbb{R}^{w\times h\times c} \rightarrow \mathbb{R}^{w\times h\times c}$ that removes dense areas of adversarial pixels in order to bypass the defense. Essentially the attacker aims to construct the inverse of the function h, such that when erosion, dilation and contour filling is applied by h, the threshold is not reached and will not be successfully masked.

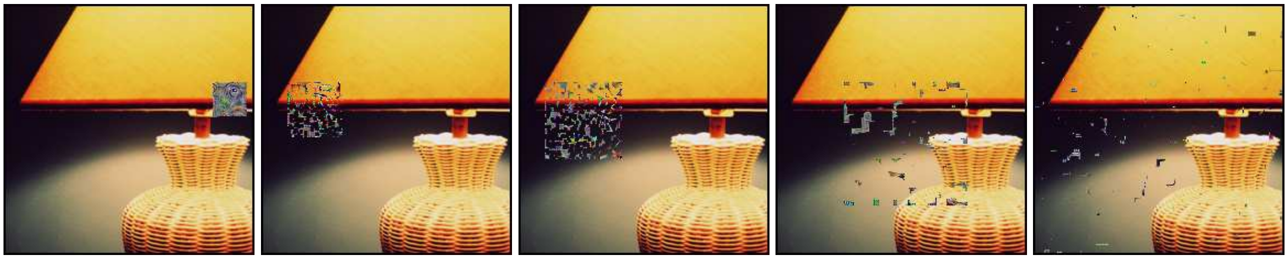
Ideally, the attacker would like to maximize the minimum neighbour distance between adversarial pixels. There are many options for choosing pixels according to this criteria such as Poisson-disk sampling or Halton sequences. However, we observed that choosing a set of pixels uniformly at random and performing the inverse of the erosion and dilation operations (commonly referred to as morphological opening) to separate adversarial pixels, provided a good approximation.

To construct this new attack, we include another hyperparameter, the bounding area the attacker can manipulate and additionally, the percentage of pixels the attacker chooses to manipulate. By removing clusters of pixels within this area we can sparsify the adversarial pixels, as shown in Figure 3.

Table 3 shows the success of LaVAN modified under this new approach; varying bounding areas and number of adversarial pixels. For small bounding areas, we found that the defense was able to resist this modified attack in general. For example, with a bounding area of 5% of the image, and modification of 2% of image pixels (restricted to be contained in the bounding area) the adversarial success if 70.8%, which is almost identical to 71.1% in the unmodified attack, while the reconstructed accuracy drops to only 91.2%. However, if we increase the bounding area, and so increase potential sparsity of adversarial pixels, the defense suffers dramatically. Manipulating 2% of image pixels within a bounding area of 25% reduces reconstruction success from 98.0% to 29.9%. In the most extreme case, if we allow the attacker to modify 2% of pixels anywhere in the image, both adversarial success improves (71.1% to 80.5%) and reconstruction success deteriorates (98.0% to 4.6%).

6. Discussion, Limitations & Conclusion

We can trivially defeat the defense if we no longer restrict adversarial pixels to be contained in a small area of an image. The attacker can distribute the location of adversarial pixels, removing the dense saliency regions that the defense relies upon. This threat model is not altogether unrealistic, it is unlikely that an attacker would be restricted to a bounding area in a real attack. For example, an attacker modifying an image to be sent to a Machine Learning as a Service model will invariably have access to the entire image and so not



(a) 2% (b) 5% (c) 10% (d) 25% (e) 100%

Figure 3: Each image has 2% adversarial pixels while we vary the bounding area. All were classified as the "Hare" class with 90% confidence by VGG-19, using the LaVAN attack.

limit themselves to a small area if they suspect a defense is in use.

The defense is sufficient to defend against localized and visible attacks in their current form, however we have shown that granting some latitude to the attacker results in bypassable defenses. Motivating the need for further research on how best to defend against such attacks. One may again consider exploiting the natural structures of images versus the unnatural structure of adversarial perturbations. For example, empirically we observed that nearly all salient figures for ImageNet samples contained thin but continuous regions that defined the most influential parts of the image, while our modified attack produces a sparse noisy structure. However, further research is needed to show this kind of reasoning can be extrapolated to defend against localized and visible adversarial perturbations in other domains.

References

[1] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018. 3, 6

[2] A. Athalye and I. Sutskever. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017. 2

[3] O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, and A. Criminisi. Measuring neural net robustness with constraints. In *Advances in neural information processing systems*, pages 2613–2621, 2016. 3

[4] A. N. Bhagoji, D. Cullina, and P. Mittal. Dimensionality reduction as a defense against evasion attacks on machine learning classifiers. *arXiv preprint arXiv:1704.02654*, 2017. 3

[5] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017. 1, 6

[6] J. Buckman, A. Roy, C. Raffel, and I. Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *International Conference on Learning Representations*, 2018. 3

[7] N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM, 2017. 3, 6

[8] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017. 1

[9] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. *arXiv preprint arXiv:1709.04114*, 2017. 1

[10] G. S. Dhillon, K. Azizzadenesheli, J. D. Bernstein, J. Kossaifi, A. Khanna, Z. C. Lipton, and A. Anandkumar. Stochastic activation pruning for robust adversarial defense. In *International Conference on Learning Representations*, 2018. 3

[11] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017. 3

[12] Z. Gong, W. Wang, and W.-S. Ku. Adversarial and clean data are not twins. *arXiv preprint arXiv:1704.04960*, 2017. 3

[13] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1

[14] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017. 3

[15] S. Gu and L. Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014. 3

[16] C. Guo, M. Rana, M. Cissé, and L. van der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017. 3

[17] D. Hendrycks and K. Gimpel. Early methods for detecting adversarial images. 2017. 3

[18] R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári. Learning with a strong adversary. *arXiv preprint arXiv:1511.03034*, 2015. 3

[19] J. Jin, A. Dundar, and E. Culurciello. Robust convolutional neural networks under adversarial noise. *arXiv preprint arXiv:1511.06306*, 2015. 3

[20] D. Karmon, D. Zoran, and Y. Goldberg. Lavan: Localized and visible adversarial noise. *arXiv preprint arXiv:1801.02608*, 2018. 1, 6

[21] X. Li and F. Li. Adversarial examples detection in deep networks with convolutional filter statistics. *CoRR, abs/1612.07767*, 7, 2016. 3

[22] B. Liang, H. Li, M. Su, X. Li, W. Shi, and X. Wang. Detecting adversarial examples in deep networks with adaptive noise reduction. *arXiv preprint arXiv:1705.08378*, 2017. 3

[23] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, M. E. Houle, D. Song, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In *International Conference on Learning Representations*, 2018. 3

[24] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 1

[25] S. M. Moosavi Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, number EPFL-CONF-218057, 2016. 1

[26] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 372–387. IEEE, 2016. 1

[27] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pages 582–597. IEEE, 2016. 3

[28] E. Quiring, D. Arp, and K. Rieck. Fraternal twins: Unifying attacks on machine learning and digital watermarking. *arXiv preprint arXiv:1703.05561*, 2017. 1

[29] A. Rozsa, E. M. Rudd, and T. E. Boult. Adversarial diversity and hard positive generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 25–32, 2016. 3

[30] P. Samangouei, M. Kabkab, and R. Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations*, 2018. 3

[31] U. Shaham, Y. Yamada, and S. Negahban. Understanding adversarial training: Increasing local stability of neural nets through robust optimization. *arXiv preprint arXiv:1511.05432*, 2015. 3

[32] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 4

[33] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *International Conference on Learning Representations*, 2018. 3

[34] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 4

[35] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[36] A. Telea. An image inpainting technique based on the fast marching method. *Journal of graphics tools*, 9(1):23–34, 2004. 3

[37] J. Uesato, B. O'Donoghue, A. v. d. Oord, and P. Kohli. Adversarial risk and the dangers of evaluating against weak attacks. *arXiv preprint arXiv:1802.05666*, 2018. 3, 6

[38] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018. 3

[39] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014. 4

[40] S. Zheng, Y. Song, T. Leung, and I. Goodfellow. Improving the robustness of deep neural networks via stability training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4480–4488, 2016. 3