Appendix - Shape Matters: Deformable Patch Attack

Zhaoyu Chen[1], Bo Li[2†], Shuang Wu[2], Jianghe Xu[2],
Shouhong Ding[2], and Wenqiang Zhang[1,3]

[1] Academy for Engineering and Technology, Fudan University
[2] Youtu Lab, Tencent
[3] Yiwu Research Institute of Fudan University

1 Overview

This document provides more details of our **DAPatch**, organized as follows:

- In Section 2, we describe the whole algorithm of Deformable Adversarial Patch, corresponding to Section 3.3 of the main body.
- In Section 3, we supplement the description of the experimental setup, corresponding to Section 4.1 of the main body.
- In Section 4, we show the complete data of white box attacks under different areas, corresponding to Section 4.2, 4.3 and 4.6 of the main body.
- In Section 5, we add physical attack examples at different angles and lighting, corresponding to Section 4.5 of the main body.
- In Section 6, we show the ablation study on some hyper-parameters, including the sparsity of activation function, the shape loss L_{shape}, the number of rays R, and shape ratio s.
- In Section 7, we provide more visual comparison.

2 Deformable Adversarial Patch

Our proposed Deformable Adversarial Patch is summarized as Algorithm 1.

3 Experimental Setup

3.1 Comparable Methods

We use circular and square shape initialization in both GAP [1] and LaVAN [5]. We use random noise as initialization for both untargeted and targeted attacks. For all experiments, we set the number of iterations to 100 and γ to 1. u is the value of 1 pixel value after normalization, α equals to $8u$ before perturbation tuning, and α equals to u after perturbation tuning. For PS-GAN [6], we use PS-GAN with weak constraints. The settings and constraints are the same as [6].

† indicates the corresponding author (libraboli@tencent.com).

Algorithm 1 Deformable Adversarial Patch (DAPatch)

Input: image $x \in [x_{\min}, x_{\max}]$, label y, the center O, the number of rays R, ray length array $r = \{r_1, r_2, ..., r_n\}$, shape ratio s, patch percent pc, perturbation step α, regular parameter β, ray step γ, the number of iteration T
Output: x_{adv}^T

Random sample δ^0 from $[x_{\min}, x_{\max}]$
$r^0 \leftarrow r$, $x_{adv}^0 \leftarrow x$
for $k \in [1,\ T]$ **do**
  if $k < \text{int}(s\ *\ T)$ **then**
    $M^k \leftarrow \text{DRP}(O, R, r^{k-1})$
    $x_{adv}^k \leftarrow (I - M^k) \odot x + M^k \odot \delta^{k-1}$
    $z^k \leftarrow f(x_{adv}^k)$
    $l \leftarrow L\left(z^k, y, pc, \beta\right)$
    $\delta^k \leftarrow \text{Clip}\left(\delta^{k-1} + \alpha \cdot \text{sign}(\nabla_{x_{adv}^k} l), x_{\min}, x_{\max}\right)$
    $r^k \leftarrow \text{Clip}\left(r^{k-1} + \gamma \cdot \text{sign}(\nabla_{r^{k-1}} l),\ 1,\ \infty\right)$
  else if $k == \text{int}(s\ *\ T)$ **then**
    $M \leftarrow \text{Sharpen}(M^{k-1})$
  else
    $x_{adv}^k \leftarrow (I - M) \odot x + M \odot \delta^{k-1}$
    $z^k \leftarrow f(x_{adv}^k)$
    $l \leftarrow L\left(z^k, y, pc, \beta\right)$
    $\delta^k \leftarrow \text{Clip}\left(\delta^{k-1} + \alpha \cdot \text{sign}(\nabla_{x_{adv}^k} l), x_{\min}, x_{\max}\right)$
  end if
end for
return x_{adv}^T

For a image classifier $f : x \rightarrow y$, we denote the clean image as $x \in R^{c\times h\times w}$ and the adversarial image as $x_{adv}^k \in R^{c\times h\times w}$ at the k-th iteration. We also denote the corresponding label as y and the predicted label as $\hat{y}$. In Algorithm 1, the loss function of GAP is expressed as:

$$L = \begin{cases} CE(x_{adv}^k, y), & \text{untargeted attack} \\ -CE(x_{adv}^k, \hat{y}), & \text{targeted attack} \end{cases}, \tag{1}$$

and the loss function of LaVAN is expressed as:

$$L = \begin{cases} CE(x_{adv}^k, y) - CE(x_{adv}^k, y_s), & \text{untargeted attack} \\ CE(x_{adv}^k, y) - CE(x_{adv}^k, y_t), & \text{targeted attack} \end{cases}, \tag{2}$$

where y_s is the highest class other than class y and y_t is the pre-set target class.

3.2 Adversarial Training

Adversarial training is currently the most mainstream and effective method in adversarial defense. We choose the most efficient and powerful adversarial training method as the threat model.

Fast-AT Fast-AT [11] shows that adversarial training with the fast gradient sign method (FGSM), when combined with random initialization, is as effective as PGD-based training but has significantly lower cost. In the experiment, we choose Fast-AT ($\epsilon = 4/255$) as the benchmark model of the attack.

Feature Denoising Feature Denoising [12] is the state-of-the-art defense against traditional perturbation-based adversarial attacks in a white-box setting, which contains blocks that denoise the features using non-local means or other filters. On ImageNet, under 10-iteration PGD white-box attacks, it achieves 55.7%. Even under extreme 2000-iteration PGD white-box attacks, it secures 42.6% accuracy. In the experiment, we choose its three open source models (Adv-ResNet-152, ResNet-152-Denoise and Resnext-101-Deniose) as the benchmark models for the attack.

4 More Experimental Results

We demonstrate the effectiveness of our proposed DAPatch on models of different architectures. We divide the model architecture into three categories: **Convolutional Neural Network** (VGG19 [9], Resnet-152 [3], DenseNet-161 [4] and MobileNet V2 [8]), **Vision Transformer** (ViT-B/16 [2] and Swin-B [7]), and **Neural Architecture Search** (EfficientNet-b7 [10]).

Table 2 illustrates that a particular shape can provide attack performance when textures are disabled. The area of the DAPatch shape is smaller than its convex hull, but it achieves a higher ASR. The experimental results in untargeted setting under 5 different patch areas on ILSVRC2012 and GTSRB are summarized in Table 3 and Table 4. For the more challenging targeted setting, the experimental results on ILSVRC2012 are reported in Table 5. The results of untargeted attacks on shape and texture bias are shown in Table 6. The results of untargeted attacks on adversarial training are shown in Table 7. All experiments show that when the patch area is small, DAPatch always obtains a higher ASR with a smaller area. Furthermore, under different patch areas, DAPatch can always obtain better attack performance within a smaller area compared with state-of-the-art methods.

5 Physical Attack

In this section, we provide more visual details about physical attacks. Figure 1 shows the different class example of DAPatch. Figure 2 shows the examples of DAPatch under different angles and lightning. So please **zoom** Figure 1 and Figure 2 to get more clearer shape details.

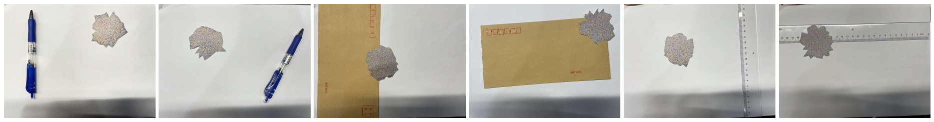
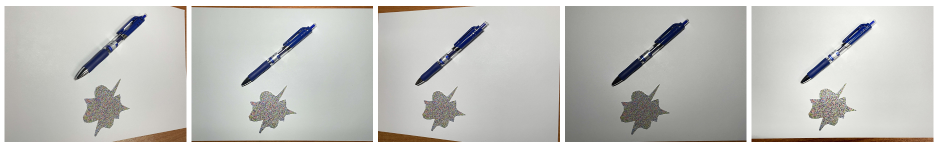


Fig. 1. More examples of physical attacks of DAPatch in untargeted setting.



(a) Middle, $-30°$ (b) Middle, $0°$ (c) Middle, $30°$ (d) Low, $0°$ (e) High, $0°$

Fig. 2. More examples of DAPatch with different angles and lightning.

6 Ablation Study

6.1 Ablation Study on λ

We review the special activation function Φ, which is expressed as:

$$\Phi(x) = \frac{\tanh\left(\lambda(x-1)\right)+1}{2}. \tag{3}$$

The λ controls the sparsity of activation function. Here, we study the effect of λ on attack performance in 2% area under the untargeted setting, as shown in Table 1. Experiments show that when $\lambda = -100$, the attack performance is better. tanh can better make the mask close to binarization, so it is selected as the activation function.

Table 1. Ablation study on λ.

λ	-10	-50	-100	-300	-500
MoblieNet v2	89.9	94.9	**97.6**	93.6	92.6
Vit-B/16-224	75.5	88.9	**95.0**	86.1	84.8
ResNet-152	77.1	86.8	**93.1**	84.0	83.1

6.2 Ablation Study on Shape Loss L_{shape}

The area of patches needs to be controlled by L_{shape}. We compare the area variation wrt L_{shape} in Figure 3. We find that, if there is no L_{shape}, the area of the patch will increase indefinitely, but this does not meet the experimental settings. According to Figure 3, When $\beta = 10$, L_{shape} cannot control the area well. When the β is large, the patch area can be better constrained to be within the specified percentage. Note that when β is large, ASR and area are not sensitive to beta, so in the experiment, to better control the area, we set $\beta = 200$.

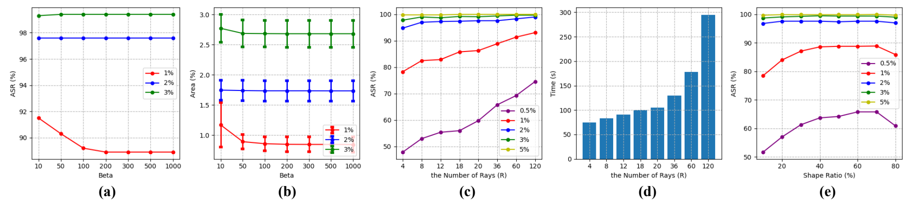


Fig. 3. Ablation study. (a) shows the relationship between β and ASR. Error bars in (b) represent the standard deviation of the area of DAPatch. We can find when β is large, ASR and area are not sensitive to β. (c) and (d) are the ASR upper bound analysis. More rays can model patches with higher ASR and the result is saturated with 120 rays. The time cost increases as R becomes larger. (e) is the ablation study on the shape ratio s. It has the best attack performance at about $s = 70$.

6.3 Ablation Study on the Number of Rays R

It plays a fundamental role in the DAPatch and explicitly affects the shape modeling ability of the patch. From Figure 3 (c), more rays show higher upper bound and better ASR. For example, 36 rays improve by 6.0% ASR compared to 20 rays in 0.5% patch percentage. The 120 rays also saturate the performance since it depicts the patches well already and the rays are not the only constraint. Note that according to Figure 3 (d), the performance of 120 rays is not much improved compared to 36 rays, which is higher than other baseline methods, but it brings more than double the training time. In practice, considering for the efficiency, we set $R = 36$.

6.4 Ablation Study on Shape Ratio s

The shape ratio s is an important parameter to perturbation tune in DAPatch. Therefore, we evaluate the untargeted attack performance concerning different shape ratios s on MobileNet v2 in Figure 3 (e). When the patch area is small, s will greatly affect the attack effect. When the patch area is large, the effect of s is not very obvious. In practice, we choose $s = 70$.

7 More Visual Comparison

In this section, we provide more visual details. Figure 4 shows the visualization of DAPatch and other patch attacks. **Single** represents the single-anchor deformable patch representation and **Multi** represents the multi-anchor deformable patch representation. Figure 5 shows the deformation process of DA-Patch in untargeted attacks under 5% area. **Disabling texture** means we just deform the shape and keep perturbations as white. The patches are generated on Mobilenet v2. The patches generated by multi-anchor deformable patch representation have more complex shapes, and there are cases where rays and contours intersect multiple times and the interior is hollowed out. So please **zoom** Figure 4 and Figure 5 to get more clearer shape details.

Table 2. The area of the convex hull is larger than DAPtach, but the attack performance is not as good as it, which shows that having a specific shape can improve the attack performance.

Network	Shape	0.5%		1%		2%		3%		5%	
		ASR	Area	ASR	Area	ASR	Area	ASR	Area	ASR	Area
MoblieNet v2	Circle	1.5	0.510	2.2	0.964	4.5	2.040	6.8	3.031	10.2	4.982
	Square	1.4	0.504	1.7	1.054	3.3	2.010	4.4	3.023	5.9	4.888
	Ours	**8.9**	**0.377**	**13.4**	**0.790**	**21.0**	**1.648**	**25.8**	**2.496**	**35.7**	**4.340**
	Convex hull	2.4	0.902	4.0	2.143	6.4	4.090	9.1	5.676	12.4	8.296
Vit-B/16-224	Circle	0.9	0.510	1.4	0.964	2.2	2.040	2.2	3.031	2.7	4.982
	Square	0.5	0.504	0.7	1.054	1.1	2.010	1.6	3.023	2.0	4.888
	Ours	**8.6**	**0.355**	**12.0**	**0.789**	**16.3**	**1.563**	**20.7**	**2.507**	**27.2**	**4.247**
	Convex hull	1.3	0.720	2.4	1.874	3.4	3.644	4.4	5.149	5.5	7.643
ResNet-152	Circle	0.9	0.510	1.2	0.964	2.6	2.040	3.3	3.031	4.6	4.982
	Square	0.5	0.504	0.6	1.054	0.8	2.010	1.3	3.023	2.0	4.888
	Ours	**5.8**	**0.371**	**10.3**	**0.776**	**18.4**	**1.618**	**23.6**	**2.449**	**27.2**	**4.251**
	Convex hull	1.0	0.880	1.5	2.102	3.6	4.110	5.0	5.745	6.7	8.271

Table 3. Untargeted attacks of various network architectures on ILSVRC2012.

Network	Method	≈0.5%		≈1%		≈2%		≈3%		≈5%	
		ASR	Area	ASR	Area	ASR	Area	ASR	Area	ASR	Area
VGG-19	GAP_s	73.4	0.510	92.6	0.964	98.4	2.040	99.3	3.031	99.7	4.982
	GAP_c	72.4	0.504	94.5	1.054	98.7	2.010	99.3	3.023	99.8	4.888
	LaVAN_s	76.9	0.510	92.6	0.964	98.9	2.040	99.5	3.031	100.0	4.982
	LaVAN_c	78.5	0.504	95.1	1.054	99.0	2.010	99.5	3.023	100.0	4.888
	PS-GAN	74.5	0.510	94.2	0.964	97.4	2.040	99.2	3.031	100.0	4.982
	Ours	**78.6**	**0.449**	**95.6**	**0.868**	**99.1**	**1.744**	**99.5**	**2.759**	**100.0**	**4.598**
ResNet-152	GAP_s	44.3	0.510	71.0	0.964	89.5	2.040	96.5	3.031	99.5	4.982
	GAP_c	44.8	0.504	74.4	1.054	91.2	2.010	97.8	3.023	99.7	4.888
	LaVAN_s	43.7	0.510	67.5	0.964	88.3	2.040	95.9	3.031	99.6	4.982
	LaVAN_c	43.5	0.504	71.2	1.054	90.4	2.010	96.8	3.023	99.8	4.888
	PS-GAN	44.5	0.510	68.9	0.964	91.3	2.040	97.4	3.031	99.7	4.982
	Ours	**52.2**	**0.409**	**78.8**	**0.845**	**93.1**	**1.699**	**97.9**	**2.623**	**99.8**	**4.546**
DenseNet-161	GAP_s	48.6	0.510	74.4	0.964	94.6	2.040	97.9	3.031	99.8	4.982
	GAP_c	49.6	0.504	79.8	1.054	94.5	2.010	98.6	3.023	99.6	4.888
	LaVAN_s	46.3	0.510	73.6	0.964	93.5	2.040	97.6	3.031	99.8	4.982
	LaVAN_c	48.0	0.504	78.1	1.054	93.5	2.010	98.0	3.023	100.0	4.888
	PS-GAN	47.5	0.510	77.7	0.964	93.7	2.040	98.5	3.031	99.9	4.982
	Ours	**55.5**	**0.417**	**83.2**	**0.851**	**96.4**	**1.718**	**99.0**	**2.656**	**100.0**	**4.546**
MoblieNet v2	GAP_s	57.9	0.510	83.6	0.964	96.2	2.040	98.6	3.031	99.9	4.982
	GAP_c	57.8	0.504	86.7	1.054	97.2	2.010	99.2	3.023	100.0	4.888
	LaVAN_s	56.6	0.510	81.8	0.964	95.6	2.040	98.9	3.031	99.9	4.982
	LaVAN_c	58.0	0.504	84.3	1.054	96.6	2.010	98.7	3.023	99.8	4.888
	PS-GAN	54.5	0.510	84.2	0.964	95.5	2.040	99.3	3.031	99.9	4.982
	Ours	**65.8**	**0.423**	**88.9**	**0.847**	**97.6**	**1.735**	**99.4**	**2.684**	**100.0**	**4.578**
Efficientnet-b7	GAP_s	43.3	0.510	63.5	0.964	85.5	2.040	91.5	3.031	97.2	4.982
	GAP_c	42.5	0.504	68.8	1.054	88.0	2.010	94.4	3.023	97.5	4.888
	LaVAN_s	42.3	0.510	64.7	0.964	89.5	2.040	95.9	3.031	98.3	4.982
	LaVAN_c	41.2	0.504	69.2	1.054	89.2	2.010	95.9	3.023	98.1	4.888
	PS-GAN	40.9	0.510	65.8	0.964	89.3	2.040	95.2	3.031	97.9	4.982
	Ours	**45.7**	**0.442**	**71.1**	**0.956**	**89.6**	**2.003**	**95.9**	**3.014**	**98.3**	**4.824**
Vit-B/16-224	GAP_s	47.0	0.510	72.0	0.964	92.4	2.040	97.2	3.031	99.8	4.982
	GAP_c	45.6	0.504	77.2	1.054	93.0	2.010	97.8	3.023	99.7	4.888
	LaVAN_s	44.8	0.510	71.8	0.964	93.5	2.040	98.3	3.031	99.9	4.982
	LaVAN_c	46.9	0.504	74.9	1.054	93.5	2.010	98.3	3.023	99.9	4.888
	PS-GAN	45.9	0.510	71.9	0.964	90.2	2.040	97.4	3.031	99.8	4.982
	Ours	**56.9**	**0.417**	**80.9**	**0.849**	**95.0**	**1.717**	**98.3**	**2.676**	99.9	**4.528**
Swin-B-224	GAP_s	32.4	0.510	68.2	0.964	91.8	2.040	97.6	3.031	99.7	4.982
	GAP_c	34.4	0.504	77.7	1.054	94.5	2.010	98.6	3.023	99.6	4.888
	LaVAN_s	35.3	0.510	68.6	0.964	95.8	2.040	99.0	3.031	99.9	4.982
	LaVAN_c	37.2	0.504	75.8	1.054	96.6	2.010	99.4	3.023	100.0	4.888
	PS-GAN	31.2	0.510	66.4	0.964	91.2	2.040	97.6	3.031	99.7	4.982
	Ours	**39.7**	**0.395**	**79.6**	**0.818**	**97.1**	**1.668**	**99.5**	**2.545**	**100.0**	**4.354**

Table 4. Untargeted attacks of various network architectures on GTSRB.

Network	Method	≈0.5%		≈1%		≈2%		≈3%		≈5%	
		ASR	Area	ASR	Area	ASR	Area	ASR	Area	ASR	Area
ResNet-152	GAP_s	13.6	0.510	21.4	0.964	37.8	2.040	56.0	3.031	76.0	4.982
	GAP_c	13.0	0.504	22.6	1.054	38.0	2.010	57.6	3.023	80.0	4.888
	LaVAN_s	14.6	0.510	22.8	0.964	41.6	2.040	60.4	3.031	83.8	4.982
	LaVAN_c	14.8	0.504	26.0	1.054	41.8	2.010	58.4	3.023	83.2	4.888
	PS-GAN	13.7	0.510	23.4	0.964	39.4	2.040	59.5	3.031	82.3	4.982
	Ours	**15.0**	**0.477**	**27.1**	**0.831**	**42.3**	**1.932**	**61.5**	**2.873**	**85.6**	**4.722**
Efficientnet-b7	GAP_s	20.4	0.510	45.0	0.964	68.0	2.040	84.0	3.031	94.4	4.982
	GAP_c	20.6	0.504	39.4	1.054	66.6	2.010	82.6	3.023	94.4	4.888
	LaVAN_s	22.2	0.510	46.2	0.964	74.0	2.040	89.2	3.031	95.6	4.982
	LaVAN_c	22.8	0.504	51.4	1.054	74.0	2.010	89.0	3.023	97.2	4.888
	PS-GAN	21.5	0.510	46.2	0.964	71.2	2.040	85.6	3.031	96.2	4.982
	Ours	**23.6**	**0.469**	**53.1**	**0.893**	**75.2**	**1.873**	**89.5**	**2.934**	**98.7**	**4.825**
Vit-B/16-224	GAP_s	28.6	0.510	61.2	0.964	90.0	2.040	97.6	3.031	99.8	4.982
	GAP_c	28.4	0.504	68.0	1.054	90.2	2.010	98.2	3.023	99.0	4.888
	LaVAN_s	28.2	0.510	61.6	0.964	91.8	2.040	98.2	3.031	99.8	4.982
	LaVAN_c	26.2	0.504	65.4	1.054	92.0	2.010	97.4	3.023	99.6	4.888
	PS-GAN	27.4	0.510	64.2	0.964	90.2	2.040	98.1	3.031	99.7	4.982
	Ours	**30.1**	**0.483**	**68.1**	**0.896**	**93.5**	**1.783**	**98.9**	**2.892**	**100.0**	**4.732**

Table 5. Targeted attacks of various network architectures on ILSVRC2012.

Network	Method	≈1%		≈3%		≈5%		≈7%		≈10%	
		ASR	Area	ASR	Area	ASR	Area	ASR	Area	ASR	Area
VGG-19	GAP_s	16.60	0.964	59.1	3.031	81.0	4.982	93.9	6.938	96.7	10.046
	GAP_c	21.40	1.054	58.5	3.023	77.3	4.888	90.0	6.794	98.1	10.002
	LaVAN_s	5.60	0.964	24.3	3.031	37.4	4.982	52.0	6.938	63.0	10.046
	LaVAN_c	5.20	1.054	22.8	3.023	36.3	4.888	47.0	6.794	58.4	10.002
	PS-GAN	20.60	0.964	59.4	3.031	81.4	4.982	94.2	6.938	97.2	10.046
	Ours	**21.40**	**0.864**	**61.7**	**2.676**	**82.8**	**4.560**	**94.6**	**6.521**	**98.3**	**9.225**
ResNet-152	GAP_s	5.30	0.964	41.1	3.031	68.8	4.982	87.3	6.938	95.0	10.046
	GAP_c	8.80	1.054	44.3	3.023	72.8	4.888	88.3	6.794	97.4	10.002
	LaVAN_s	3.50	0.964	22.6	3.031	45.1	4.982	61.8	6.938	78.0	10.046
	LaVAN_c	6.00	1.054	23.9	3.023	47.7	4.888	64.7	6.794	81.1	10.002
	PS-GAN	8.90	0.964	46.2	3.031	73.8	4.982	87.4	6.938	96.4	10.046
	Ours	**9.10**	**0.849**	**48.7**	**2.668**	**78.1**	**4.553**	**90.2**	**6.439**	**97.6**	**9.232**
DenseNet-161	GAP_s	9.20	0.964	45.0	3.031	73.3	4.982	92.2	6.938	96.9	10.046
	GAP_c	11.50	1.054	44.9	3.023	73.1	4.888	90.4	6.794	98.3	10.002
	LaVAN_s	4.60	0.964	21.6	3.031	35.9	4.982	51.0	6.938	67.9	10.046
	LaVAN_c	5.90	1.054	24.1	3.023	38.2	4.888	54.0	6.794	72.4	10.002
	PS-GAN	10.30	0.964	49.2	3.031	74.5	4.982	92.3	6.938	97.5	10.046
	Ours	**13.70**	**0.858**	**55.3**	**2.682**	**77.8**	**4.569**	**92.6**	**6.472**	**98.6**	**9.315**
MoblieNet v2	GAP_s	4.50	0.964	51.3	3.031	81.1	4.982	93.7	6.938	97.3	10.046
	GAP_c	5.80	1.054	51.6	3.023	81.7	4.888	94.8	6.794	99.3	10.002
	LaVAN_s	1.60	0.964	30.0	3.031	51.7	4.982	71.8	6.938	86.2	10.046
	LaVAN_c	2.60	1.054	31.8	3.023	52.8	4.888	72.5	6.794	88.1	10.002
	PS-GAN	5.90	0.964	52.4	3.031	82.3	4.982	93.9	6.938	99.5	10.046
	Ours	**7.60**	**0.869**	**54.8**	**2.271**	**84.9**	**4.593**	**94.8**	**6.481**	**99.7**	**9.433**
Efficientnet-b7	GAP_s	4.40	0.964	52.1	3.031	81.9	4.982	93.7	6.938	97.2	10.046
	GAP_c	6.20	1.054	53.6	3.023	81.4	4.888	93.4	6.794	99.0	10.002
	LaVAN_s	1.50	0.964	33.1	3.031	65.0	4.982	82.2	6.938	91.6	10.046
	LaVAN_c	2.30	1.054	34.0	3.023	62.1	4.888	83.6	6.794	95.1	10.002
	PS-GAN	4.90	0.964	53.6	3.031	81.5	4.982	93.2	6.938	99.1	10.046
	Ours	**7.60**	**0.869**	**53.6**	**2.953**	**82.0**	**4.851**	**93.7**	**6.713**	**99.3**	**10.000**
Vit-B/16-224	GAP_s	6.20	0.964	48.8	3.031	85.4	4.982	97.3	6.938	97.9	10.046
	GAP_c	7.90	1.054	50.6	3.023	85.7	4.888	97.2	6.794	100.0	10.002
	LaVAN_s	3.10	0.964	25.4	3.031	52.8	4.982	78.3	6.938	93.4	10.046
	LaVAN_c	4.20	1.054	24.7	3.023	54.9	4.888	78.4	6.794	96.6	10.002
	PS-GAN	5.30	0.964	49.3	3.031	85.8	4.982	97.1	6.938	98.1	10.046
	Ours	**9.60**	**0.850**	**50.7**	**2.697**	**86.2**	**4.688**	**97.4**	**6.727**	**100.0**	**9.322**
Swin-B-224	GAP_s	18.50	0.964	91.2	3.031	99.5	4.982	100.0	6.938	100.0	10.046
	GAP_c	27.80	1.054	94.4	3.023	99.5	4.888	99.9	6.794	100.0	10.002
	LaVAN_s	19.00	0.964	85.1	3.031	97.0	4.982	99.4	6.938	98.0	10.046
	LaVAN_c	25.20	1.054	86.5	3.023	96.9	4.888	99.3	6.794	100.0	10.002
	PS-GAN	29.60	0.964	95.1	3.031	98.9	4.982	99.4	6.938	100.0	10.046
	Ours	**38.20**	**0.848**	**98.5**	**2.587**	**99.7**	**4.371**	**100.0**	**6.216**	**100.0**	**8.912**

Table 6. Untargeted attacks of shape and texture bias on ILSVRC2012.

Network	Method	≈0.5%		≈1%		≈2%		≈3%		≈5%	
		ASR	Area	ASR	Area	ASR	Area	ASR	Area	ASR	Area
ResNet50-SIN	GAP_s	70.4	0.510	87.3	0.964	96.9	2.040	99.1	3.031	99.8	4.982
	GAP_c	70.1	0.504	88.5	1.054	96.9	2.010	99.6	3.023	99.8	4.888
	LaVAN_s	66.2	0.510	82.2	0.964	95.1	2.040	98.1	3.031	99.8	4.982
	LaVAN_c	65.6	0.504	84.2	1.054	96.0	2.010	98.7	3.023	99.7	4.888
	PS-GAN	70.0	0.510	85.3	0.964	96.8	2.040	99.5	3.031	99.8	4.982
	Ours	**74.1**	**0.446**	**90.3**	**0.893**	**98.7**	**1.764**	**99.6**	**2.724**	**99.9**	**4.620**
ResNet50-SIN+IN	GAP_s	44.3	0.510	68.3	0.964	90.6	2.040	95.7	3.031	98.5	4.982
	GAP_c	44.7	0.504	72.4	1.054	90.9	2.010	96.3	3.023	99.1	4.888
	LaVAN_s	41.2	0.510	64.9	0.964	88.8	2.040	94.6	3.031	98.7	4.982
	LaVAN_c	42.4	0.504	69.4	1.054	89.4	2.010	96.1	3.023	98.9	4.888
	PS-GAN	44.5	0.510	68.9	0.964	90.6	2.040	95.2	3.031	98.2	4.982
	Ours	**48.1**	**0.426**	**75.6**	**0.860**	**91.5**	**1.750**	**96.3**	**2.669**	**99.1**	**4.587**
ResNet50-SIN+IN-IN	GAP_s	41.6	0.510	62.2	0.964	85.6	2.040	93.0	3.031	98.0	4.982
	GAP_c	44.3	0.504	68.6	1.054	87.4	2.010	94.2	3.023	98.3	4.888
	LaVAN_s	38.7	0.510	58.2	0.964	83.1	2.040	92.8	3.031	98.2	4.982
	LaVAN_c	39.6	0.504	65.3	1.054	84.7	2.010	93.4	3.023	98.2	4.888
	PS-GAN	39.2	0.510	62.7	0.964	85.2	2.040	94.7	3.031	98.6	4.982
	Ours	**44.3**	**0.420**	**70.1**	**0.845**	**88.9**	**1.729**	**95.1**	**2.651**	**99.0**	**4.562**
ResNet50-Debiased	GAP_s	42.5	0.510	64.2	0.964	85.8	2.040	93.1	3.031	98.4	4.982
	GAP_c	44.0	0.504	68.6	1.054	87.9	2.010	94.3	3.023	98.4	4.888
	LaVAN_s	38.2	0.510	59.7	0.964	83.3	2.040	90.9	3.031	97.9	4.982
	LaVAN_c	38.8	0.504	64.2	1.054	84.3	2.010	93.2	3.023	98.6	4.888
	PS-GAN	43.2	0.510	67.6	0.964	88.0	2.040	93.4	3.031	98.0	4.982
	Ours	**46.6**	**0.438**	**68.6**	**0.852**	**88.1**	**1.744**	**94.7**	**2.665**	**98.6**	**4.593**
ResNet152-Debiased	GAP_s	33.3	0.510	53.0	0.964	81.4	2.040	91.0	3.031	98.3	4.982
	GAP_c	34.1	0.504	58.9	1.054	84.1	2.010	93.0	3.023	98.4	4.888
	LaVAN_s	30.6	0.510	49.8	0.964	77.0	2.040	88.8	3.031	98.1	4.982
	LaVAN_c	29.8	0.504	52.1	1.054	77.5	2.010	90.1	3.023	98.4	4.888
	PS-GAN	32.3	0.510	53.4	0.964	83.0	2.040	93.2	3.031	98.3	4.982
	Ours	**37.4**	**0.422**	**61.8**	**0.850**	**85.3**	**1.735**	**94.5**	**2.626**	**98.5**	**4.532**

Table 7. Untargeted attacks on networks with adversarial training.

Network	Method	≈0.5%		≈1%		≈2%		≈3%		≈5%	
		ASR	Area	ASR	Area	ASR	Area	ASR	Area	ASR	Area
Adv-ResNet-152	GAP_s	61.0	0.510	74.5	0.964	86.6	2.040	90.3	3.031	94.4	4.982
	GAP_c	60.6	0.504	77.4	1.054	87.2	2.010	91.7	3.023	95.1	4.888
	LaVAN_s	58.4	0.510	71.1	0.964	83.9	2.040	88.8	3.031	93.8	4.982
	LaVAN_c	57.2	0.504	72.6	1.054	83.9	2.010	89.3	3.023	94.8	4.888
	PS-GAN	59.2	0.510	77.7	0.964	84.3	2.040	89.7	3.031	95.2	4.982
	Ours	**62.5**	**0.472**	**78.4**	**0.948**	**88.2**	**1.921**	**92.4**	**2.791**	**96.6**	**4.703**
ResNet-152-Denoise	GAP_s	59.3	0.510	74.5	0.964	86.5	2.040	92.6	3.031	96.4	4.982
	GAP_c	59.0	0.504	77.3	1.054	87.8	2.010	92.9	3.023	96.4	4.888
	LaVAN_s	59.6	0.510	72.6	0.964	84.7	2.040	91.8	3.031	96.9	4.982
	LaVAN_c	60.7	0.504	75.1	1.054	85.5	2.010	92.7	3.023	96.5	4.888
	PS-GAN	61.7	0.510	75.1	0.964	86.2	2.040	92.2	3.031	96.3	4.982
	Ours	**62.3**	**0.464**	**77.4**	**0.959**	**88.0**	**1.835**	**92.9**	**2.853**	**98.3**	**4.619**
Resnext-101-Deniose	GAP_s	50.4	0.510	66.3	0.964	83.8	2.040	90.2	3.031	95.0	4.982
	GAP_c	51.1	0.504	70.5	1.054	84.2	2.010	89.9	3.023	95.7	4.888
	LaVAN_s	49.7	0.510	65.0	0.964	80.6	2.040	87.6	3.031	94.9	4.982
	LaVAN_c	49.5	0.504	67.9	1.054	81.2	2.010	88.4	3.023	95.2	4.888
	PS-GAN	51.2	0.510	68.1	0.964	80.9	2.040	89.9	3.031	94.7	4.982
	Ours	**52.9**	**0.471**	**68.9**	**0.949**	**85.5**	**1.928**	**90.2**	**2.814**	**95.7**	**4.706**
Fast_AT	GAP_s	50.4	0.510	62.3	0.964	80.4	2.040	88.5	3.031	95.0	4.982
	GAP_c	50.6	0.504	65.5	1.054	80.3	2.010	88.8	3.023	94.9	4.888
	LaVAN_s	48.7	0.510	60.2	0.964	77.5	2.040	84.7	3.031	92.9	4.982
	LaVAN_c	48.7	0.504	62.6	1.054	78.0	2.010	85.3	3.023	93.1	4.888
	PS-GAN	48.9	0.510	63.4	0.964	79.1	2.040	85.2	3.031	93.2	4.982
	Ours	**51.3**	**0.473**	**65.6**	**0.944**	**82.0**	**1.890**	**90.0**	**2.963**	**95.4**	**4.772**

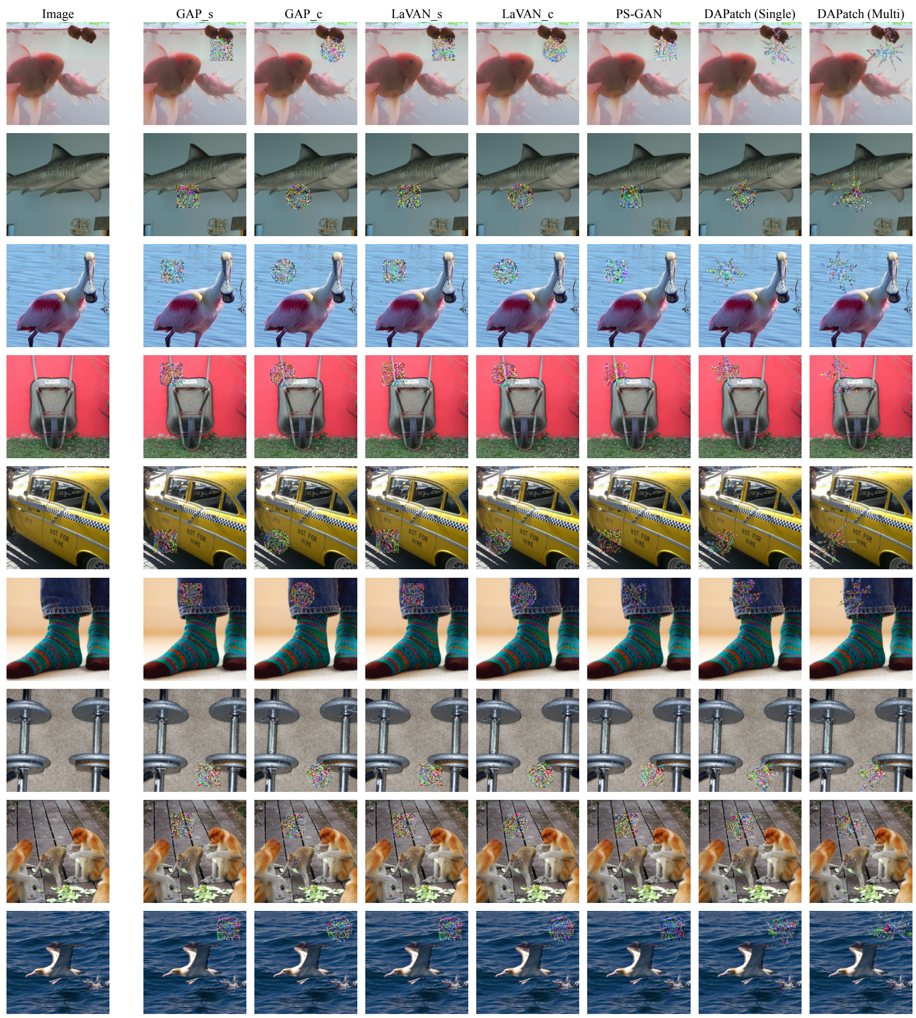


Fig. 4. Visualization of DAPatch and other patch attacks under 5% area. Please **zoom** images for better shape details.

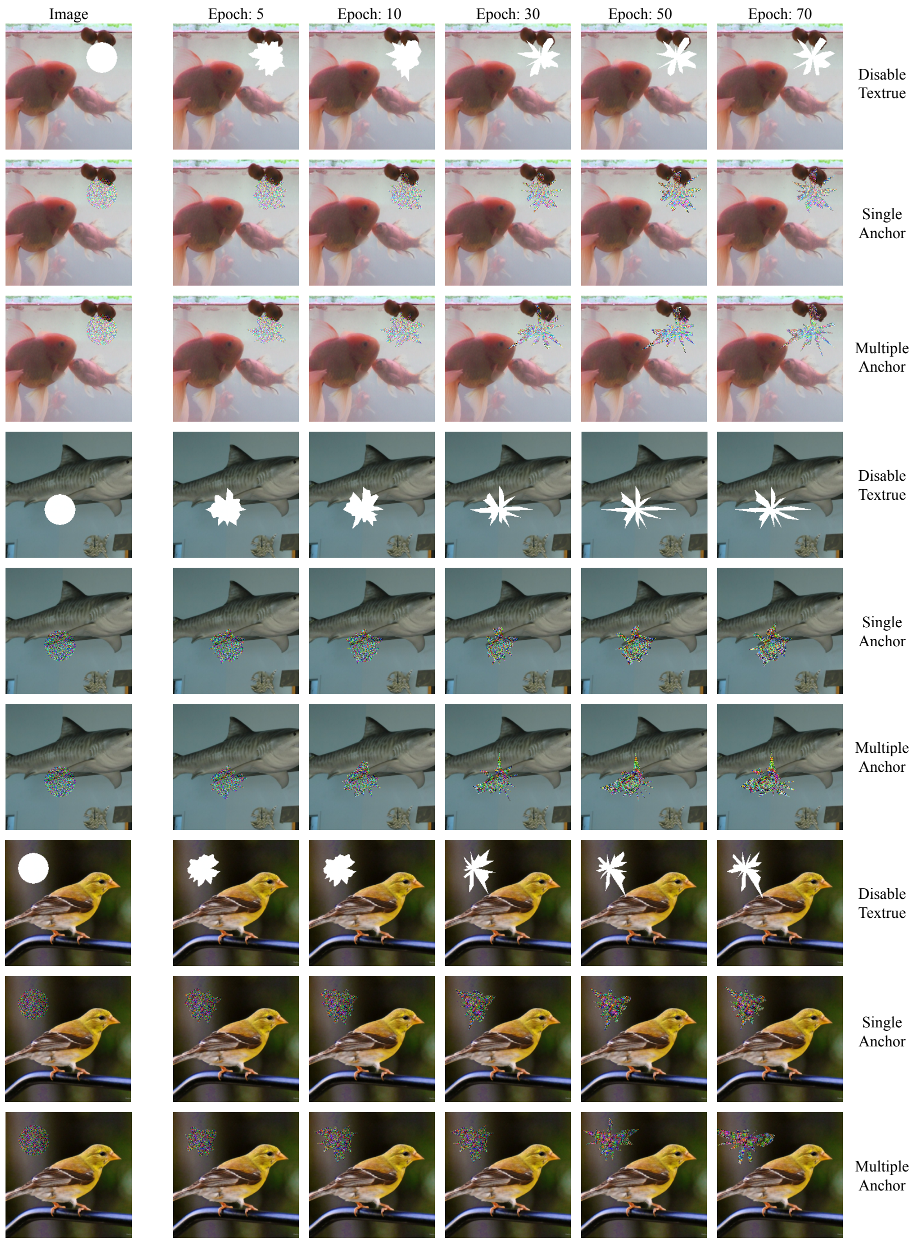


Fig. 5. Deformation process of DAPatch in untargeted attacks under 5% area. Please **zoom** images for better shape details.

References

1. Brown, T.B., Mané, D., Roy, A., Abadi, M., Gilmer, J.: Adversarial patch (2017), http://arxiv.org/abs/1712.09665 1
2. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net (2021), https://openreview.net/forum?id=YicbFdNTTy 3
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 770–778. IEEE Computer Society (2016). https://doi.org/10.1109/CVPR.2016.90, https://doi.org/10.1109/CVPR.2016.90 3
4. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 2261–2269. IEEE Computer Society (2017). https://doi.org/10.1109/CVPR.2017.243, https://doi.org/10.1109/CVPR.2017.243 3
5. Karmon, D., Zoran, D., Goldberg, Y.: Lavan: Localized and visible adversarial noise. In: Dy, J.G., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. Proceedings of Machine Learning Research, vol. 80, pp. 2512–2520. PMLR (2018), http://proceedings.mlr.press/v80/karmon18a.html 1
6. Liu, A., Liu, X., Fan, J., Ma, Y., Zhang, A., Xie, H., Tao, D.: Perceptual-sensitive GAN for generating adversarial patches. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019. pp. 1028–1035. AAAI Press (2019). https://doi.org/10.1609/aaai.v33i01.33011028, https://doi.org/10.1609/aaai.v33i01.33011028 1
7. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows (2021), https://arxiv.org/abs/2103.14030 3
8. Sandler, M., Howard, A.G., Zhu, M., Zhmoginov, A., Chen, L.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018. pp. 4510–4520. IEEE Computer Society (2018). https://doi.org/10.1109/CVPR.2018.00474, http://openaccess.thecvf.com/content_cvpr_2018/html/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.html 3
9. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1409.1556 3
10. Tan, M., Le, Q.V.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long

Beach, California, USA. Proceedings of Machine Learning Research, vol. 97, pp. 6105–6114. PMLR (2019), http://proceedings.mlr.press/v97/tan19a.html 3

11. Wong, E., Rice, L., Kolter, J.Z.: Fast is better than free: Revisiting adversarial training. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net (2020), https://openreview.net/forum?id=BJx040EFvH 3
12. Xie, C., Wu, Y., van der Maaten, L., Yuille, A.L., He, K.: Feature denoising for improving adversarial robustness. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019. pp. 501–509. Computer Vision Foundation / IEEE (2019). https://doi.org/10.1109/CVPR.2019.00059, http://openaccess.thecvf.com/content_CVPR_2019/html/Xie_Feature_Denoising_for_Improving_Adversarial_Robustness_CVPR_2019_paper.html 3