Customizable Adversarial T-shirt

Haoyang Li [1] **Ruiqi Zhu** [1] **Ziyao Cheng** [1] **Lirong Yao** [1]

Abstract

We propose a method to create customized adversarial T-shirts that protects a person from being detected by a visual object detection model. It is customized in two ways: the adversarial noise is trained over customized data and the shape of the adversarial noise is restricted by a customized stencil. Our method is able to fool the YOLOv5 model with a T-shirt patched with customized adversarial noise carved by a Cornell bear stencil.

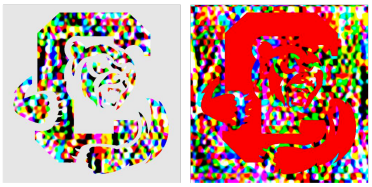

Figure 1. An example of customized adversarial noise. The left uses a stencil of the Cornell bear to carve out the noise and the right uses it to carve in.

1. Introduction

Since it was discovered that deep learning models are vulnerable to imperceptible adversarial perturbations/noises (Szegedy et al., 2013), there has been no positive application of adversarial noises until recent years. After we know that adversarial attack against object detection is possible (Xie et al., 2017) and can also live in the physical world (Kurakin et al., 2018), an interesting idea of crafting an adversarial T-shirt that shields one from being detected by an object detection model arises in the literature (Thys et al., 2019) (Kaidi Xu, 2020) (Wu et al., 2020). However, the adversarial T-shirt they proposed retains patterns that are solely designed to attack the model, or in other words, has no aesthetic design or commercial use. Motivated by the fact that clothes should be designed for human value, we propose to make the adversarial noise on a T-shirt customizable by having the attack tailored to a specific person and allowing an input stencil designed by human to shape the adversarial noise.

Although we designed the framework to be compatible with the physical world, the work is mainly done in the digital world. We require the customer to film a short video of themselves with a QR code patched to a T-shirt of their own choice. We also allow them to provide a customized stencil. The customized adversarial noise is obtained by conducting a low frequency attack against an object detection model over the visual data provided by the customer with perturbed region restricted by the stencil. This process is shown in Figure 2, and an example of the generated adversarial noise is shown in Figure 1.

[1]Cornell University. Correspondence to: Haoyang Li <hl2425@cornell.edu>.

Lirong Yao came up with the idea. Haoyang Li proposed and implemented the method using SSD Lite. Ruiqi Zhu and Ziyao Cheng migrated the method to YOLO and conducted the experiments.

2. Related Works & Motivation

Object Detection A general visual object detection model takes an image as input and outputs a list of bounding boxes, confidence scores and classes, each triple corresponding an object detected by the model. In practice, only objects with a confidence score higher than a threshold will be considered as a detected object, and otherwise it is a false positive. The previous two adversarial T-shirt studies (Wu et al., 2020) (Kaidi Xu, 2020) use YOLO (Redmon et al., 2015) and Faster R-CNN (Ren et al., 2015) as the targets.

Adversarial Attack Attack with adversarial noise can mislead the system without intruding it, which makes it more dangerous. Categorized by how much information the attacker knows, it can be divided into white-box attack where the attacker knows everything about the target model, and black-box attack where the attacker can only query the model. Any attack lies in between can be referred as gray-box attack. Since we focus on customizing the noise, we will only consider white-box attack similar to the previous adversarial T-shirt studies (Wu et al., 2020) (Kaidi Xu, 2020).

The original adversarial attack only lives in the digital world, which still requires intruding the system (Szegedy et al., 2013), but soon it is also found that a physical attack is also possible (Kurakin et al., 2018). Transferring adversarial noise from the digital world to the physical world

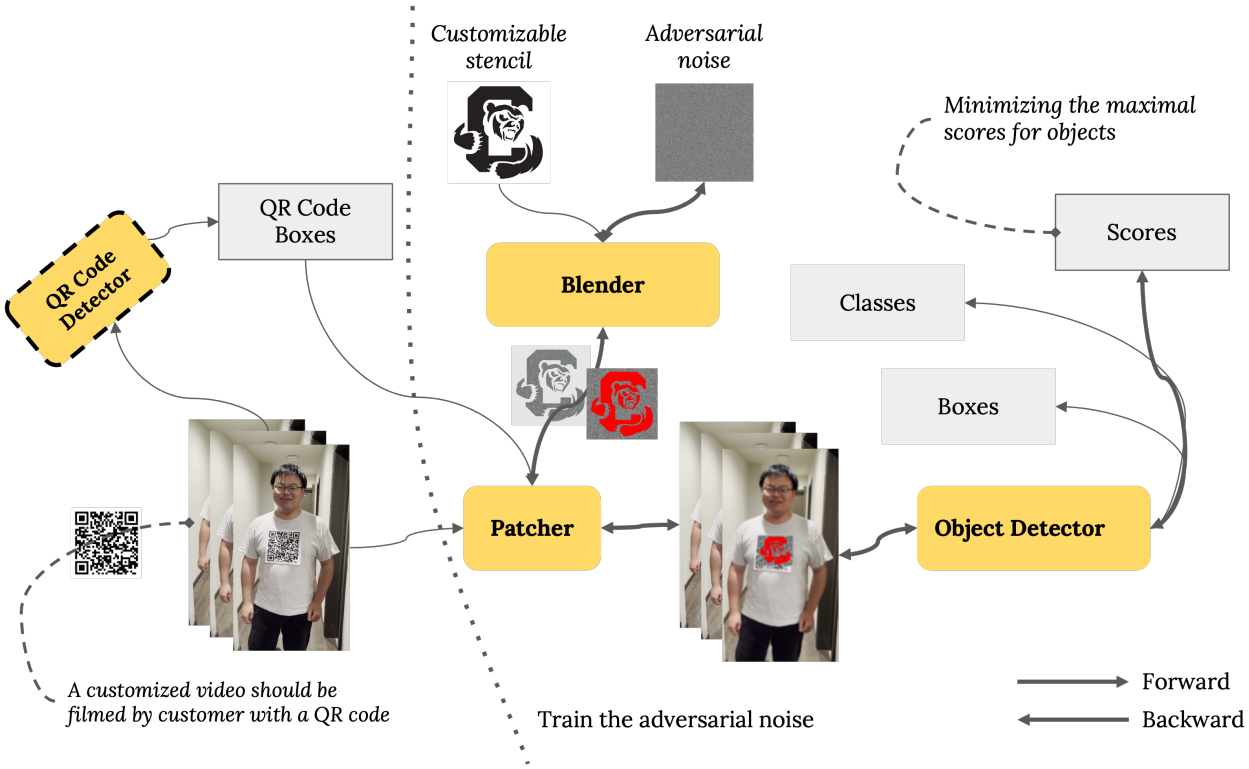

Figure 2. The schematic diagram of our method.

will require the adversarial noise to be robust to distortions brought by image processing, including JPEG compression, the distortion caused by printer, camera and the filming environment (Kaidi Xu, 2020).

The researchers in existing adversarial T-shirt studies considered how to fool the model robustly under these distortions, but they overlooked the fact that it is human who wears these T-shirts. These adversarial noises are quite effective to fool the machines, but too ugly for people to wear. Naturally, we wonder if we can customize the noise and make it more fashionable.

Crafting adversarial perturbations is essentially an optimization problem. Many well-known methods including FGSM (Goodfellow et al., 2014), PGD (Madry et al., 2017), C&W attack (Carlini & Wagner, 2017), etc. are optimization algorithms specifically designed for adversarial perturbations. To make the adversarial noise more effective, one needs to prevent overfitting just like preventing overfitting when training the model. However, since we are creating customized noise, what if we overfit to the customer? This can be possible by training adversarial noise on customized data.

3. Method

We make the adversarial T-shirt customizable in two ways. First, it is tailored to a specific person, .i.e. the customer itself. As you can see from the left part of Figure 2, the customer will need to print a general QR code out and paste it onto a specific region of a T-shirt of pure color, record a short video of himself and send it to us. We will break this video into frames, and run a QR detector, on which there are a large amount of open source implementations, and create a small dataset tailored to the customer.

The dataset consists of two parts: images with clear QR codes in sight, and corresponding boxes of the QR codes. These QR codes mark the region where adversarial noises are going to be patched. It provides a cheap and simple way to tracking the physical location and movement of T-shirt patterns.

The second customizable part is the adversarial noise itself. We make use of a popular tool in graffiti, namely stencil. The stencils are easily customizable and there exist matured techniques to transform any image into a stencil. The customer can provide us with a stencil of its own choice or an image out of which we can create a stencil. These stencils are used to carved out the adversarial canvas (or the other way), as shown in Figure 2. In other words, it applies a customizable L_0 regularization on the adversarial noise.

In the forward pass of each training step, the current adversarial noise is regularized with the stencil by a blender and patched to a frame of the image onto the QR code region before fed into an object detector. The optimization goal is to minimize the maximal scores of boxes in the outputs of the object detector. There are multiple losses that achieve the goal, and the most simple one is to directly minimize the maximal score.

To make it realistic, the patch should be done by replacement other than addition and we also applied a frequency domain regularization to eliminate high frequency components in the adversarial noise using the method described in (Guo et al., 2018). Low-frequency regularization also reduces the distortions brought by JPEG compression.

The problem is formulated as follows:

$$\min \mathcal{L}(\mathcal{D}(p \circ \delta' + (1-p) \circ x), y_{\text{adv}}) \tag{1}$$

where

$$\delta' = \mathcal{T}(s \circ \text{DCT}^{-1}(\eta_r \cdot \text{DCT}(\delta)) + (1-s) \circ w) \tag{2}$$

The adversarial noise δ is applied with a low-frequency regularization using DCT (Discrete Cosine Transform), where η_r is a mask parameterized by $r \in [0, 1]$ that only keeps the bottom r low frequency weights and regularized in spatial domain with the stencil s, where w is a pure color image of the same size to either carve out the stencil or carve in the stencil. The regularized noise is then translated and padded to the location of the QR code, the process is denoted as $\mathcal{T}$. It is then patched to the input x and fed into the object detector $\mathcal{D}$ using a patch mask p, where only the QR code region is set to one. The loss $\mathcal{L}$ is built on the output of $\mathcal{D}$ and an adversarial target y_{adv} which denotes the desired output for a successful adversarial attack.

In practice, we do not need to find the optimal solution to the problem above. We just need to find a solution that reduces the maximal confidence score lower than the threshold, and the model would think it is a false positive.

4. Experiments

As a proof of concept, we first examine the effectiveness of the attack formulation described above with a relatively simple task, attacking object classification on ResNet(He et al., 2016)[1]. Compared to object detection, object classification is more prone to attack because defining a qualified adversarial target (i.e., the target output which we use to define an adversarial loss) is straightforward. In our experiments, we keep the cross-entropy loss untouched and define the adversarial target y_{adv} as the inverse of the correct prediction:

$$y_{\text{adv}} = \mathbf{1}_n - y \tag{3}$$

where n is the number of classes predicted by ResNet and y is the one-hot ground truth label. Results of our attack are shown in Table 4 and Figure 2. We can see that limiting the attack in low frequency space requires more iterations to successfully attack, although both high and low frequency attacks can elicit a wrong class label within 20 SGD iterations.

Figure 3. Left: Perturbed dog image using a cat stencil that elicits the wrong label on ResNet after 10 iterations, $r = 1$. Right: Perturbed dog image using a cat stencil that elicits the wrong label on ResNet after 20 iterations, $r = 0.05$.

Input	Prediction	Score
original image	Samoyed	0.00339
perturbed, 10 steps, r = 1	birdhouse	0.00140
perturbed, 10 steps, r = 0.05	Samoyed	0.00233
perturbed, 20 steps, r = 0.05	quill	0.00217

Table 1. ResNet object classification results on various inputs. Perturbed images are shown in Figure 2.

Our next step is to attack an object detection model, and we choose YOLOv5[2]. The difference between this task and the previous one is four-fold:

1. The attack is in physical world[3], i.e., the perturbed input $\hat{x}$ is represented as a physical image before passing into the detection model. This means each pixel i satisfies $i \in [0, 1]$.

2. The attack is more limited in that adversarial noise must be contained inside the T-shirt area of the image.

[1]We use the ResNet-50 pretrained on ImageNet dataset provided by Torchvision

[2]https://github.com/ultralytics/yolov5

[3]It is in a simulated physical world, we are unable to conduct a real physical attack

This means key feature areas (like human heads) are not perturbed at all.

3. Loss definition is not apparent. Since the output of YOLO contains both the predicted bounding boxes (with confidence) and their corresponding class labels, the authors created a customized loss function in their original implementation[4].

4. It is a gray-box attack. Our attack goal is to fool the inference model of YOLO, which is essentially its training model wrapped by additional pre-and-post processing steps that block gradient computations. However, only the training model of YOLO allows gradient propagation. This means we can only use the training model as a differentiable proxy for evaluating D, and work around the processing steps to guarantee the perturbed output also can fool the inference model.

For a $640 \times 640 \times 3$ PIL image, the YOLO inference model first convert it into tensor form, and then pass it into the training model backbone. The training model outputs a tensor $y \in \mathbb{R}^{25200\times 85}$, corresponding to 25200 candidate bounding boxes proposed by the detection heads, each in the form of [center x, center y, width, height, confidence, scores of 80 classes]. The output y is then run through NMS (Non-maximum Suppression), which selects a subset of bounding boxes using an iterative algorithm. It is guaranteed that each output bounding box must have a confidence value higher than a confidence threshold, which is set to 0.25 in the default implementation.

To define an appropriate loss function, we decide to ignore the geometric parameters of the bounding boxes as well as the class probabilities, only target the confidence scores in y. This is because that the confidence score is computed as the IoU (Intersection over Union) of a predicted bounding box and its corresponding ground truth box. Therefore, training the adversarial noises against an confidence value is equivalent to pushing the bounding box away from the ground truth box. We observe that a large majority of the 25200 confidence scores are close to 0, and only a few (about 10) are above the confidence threshold. This inspires us to choose an L_1 loss function, and set the adversarial target as

$$y_{\text{adv}} = \mathbf{0}_{25200} \tag{4}$$

We use the sum reduction for this L_1 loss between y and y_{adv}, as it provides a tight upper bound for the maximum confidence score of any bounding box. In other words, this loss can propagate through non-maximum suppression.

[4]https://github.com/ultralytics/yolov5/blob/master/utils/loss.py

Adversarial attack makes some small changes to an input and elicit a different prediction. It is therefore fair to predict that small changes to an adversarial input might void the attacking effect. As mentioned before, our attack is gray-box because the YOLO training and inference models differ by additional processing steps. Two problems rise from these steps:

1. The conversion between a tensor and a PIL image is lossy, i.e., pil2tensor(tensor2pil(x)) $\neq x$.

2. Input images are first transformed using a letter box approach, which involves reshaping the input image to fit a 640×640 frame while keeping its ratio. Additional pixels are padded to create a $640\times640\times3$ image that is then passed into the training model and non-maximum suppression. Finally, the output bounding boxes are resized to match the original input size.

Both problems can amplify the discrepancy between our training goal and the actual adversarial goal. We address the first problem by adding a compensation step when constructing the perturbed image given adversarial noise. This guides the adversarial noise to minimize the adversarial loss of the perturbed tensor after the pil2tensor transform. Concretely, this means:

$$x' = \text{offset}_{\text{no\_grad}} + \delta' + x \tag{5}$$

where

$$\text{offset}_{\text{no\_grad}} = \text{pil2tensor}(\text{tensor2pil}(\delta'+x)) - (\delta'+x) \tag{6}$$

We address the second problem by requiring the input image to be of shape 640 $\times$ 640 $\times$ 3, the default image size of YOLO. Alternatively we can implement a differentiable twin of all processing steps, but adding this physical constraint is a simple way to bypass the effects of letter box reshaping.

Under different conditions, including whether the perturbed input is generated with PIL adjustments, different low frequency filtering thresholds ($r \in \{0.25, 0.5, 0.75\}$ and patch settings ($x' \in \{x'_{\text{replace}}, x'_{\text{add}}\}$[5], where

$$x'_{\text{replace}} = p \circ \text{sigmoid}(\delta') + (1-p) \circ x \tag{7}$$

uses a sigmoid function to guarantee the perturbed tensor is within the range of physical images prior to be passed into tensor2pil for the inference model, and

$$x'_{\text{add}} = p \circ \delta' + (1-p) \circ x \tag{8}$$

[5]Patching by replacement simulates a patching in physical world

Figure 4. Left: YOLO inference result on a person wearing T-shirt. Middle: Perturbed T-shirt wearing image using a white Cornell bear stencil which cause YOLO to output no detection boxes during inference. Right: Perturbed T-shirt wearing image using a black Cornell bear stencil which cause YOLO to output no detection boxes during inference.

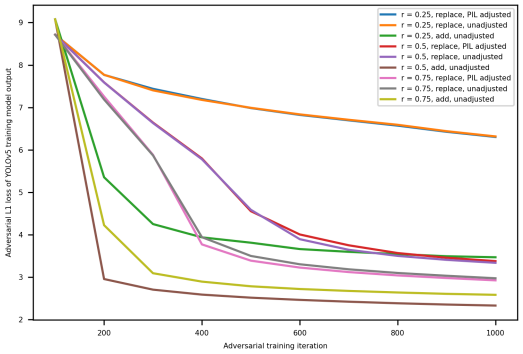

Figure 5. The adversarial iterations of experiments with stable training. Loss is evaluated on the YOLO training model.

provides no such guarantee, we conducted 12 experiments and the results are shown in Figure 4, 5, 6 and Table 2 (we use the Adam optimizer with its default parameters).

We have the following observations from the experiments:

1. While the low frequency constraint decreases the attack efficiency (inversely related to r), it also reduces the defense effect of processing steps such as transforming between PIL image and tensor forms (the training trajectories of PIL adjusted and unadjusted are the closest when $r = 0.25$). This is understandable considering image compression is also based on discrete cosine transform. In a real-world setting, additional transformations can also occur (e.g., surface distortion of the T-shirt and the distortions caused by camera and environment during capture), we suspect that limiting adversarial noises into lower frequency domain reduces their effects similarly.

2. In the cases where the constraint that perturbed tensor is within range prior to be passed into tensor2pil (when $x' = x'_{\text{add}}$) is absent, the adversarial noise converges faster (compared to all other settings) on the YOLO training model when there is no PIL adjustment, and is unstable (at least in the early iterations) when the adjustment is in place. On the other hand, we observe that the attack result is better on the YOLO inference model when with PIL adjustment. This suggests that compared to the sigmoid function, PIL adjustment might be a more limited but still useful regularization tool. We also observe that the condition with $x' = x'_{\text{add}}$, PIL adjusted setting, $r = 0.25$ corresponds to the least unstable training. This also indicates that the low frequency constraint is effective for our attack.

3. As mentioned in (Guo et al., 2018), low frequency adversarial attack reduces the time needed to acquire the desired adversarial noise.

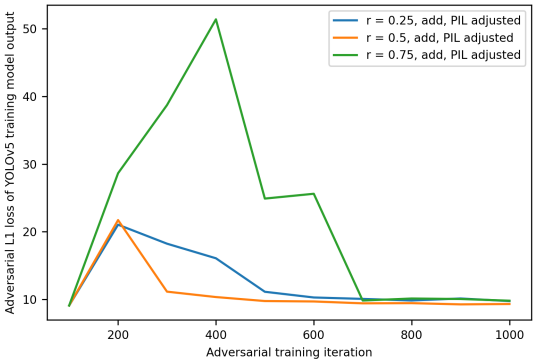

Figure 6. The adversarial iterations of experiments with unstable training. Loss is evaluated on the YOLO training model.

r	x'	PIL adjustment	time	attack result
0.25	x'_{replace}	adjusted	21 min	unsuccessful
0.5	x'_{replace}	adjusted	21 min	successful
0.75	x'_{replace}	adjusted	26 min	successful

Table 2. The attack results of the best permutation for each r after 1000 iterations. Successful attack means the final perturbed image elicits no detection from the YOLO inference model.

5. Conclusion & Future Work

We present a method to create customized adversarial T-shirts and successfully use it to fool the YOLOv5 with an adversarial T-shirt customized by a Cornell bear stencil. We observe that restraining adversarial noise in lower frequency

domain is also helpful for crafting a customized adversarial T-shirt. Potential future work include implementing a differentiable version of the image processing steps in YOLO to better simulate the real-world scenario. Besides, we also do not get to address the physical distortions and launch a real physical attack.

References

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. Ieee, 2017.

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

Guo, C., Frank, J. S., and Weinberger, K. Q. Low frequency adversarial perturbation. *arXiv preprint arXiv:1809.08758*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaidi Xu, Gaoyuan Zhang, S. L. Q. F. M. S. H. C. P.-Y. C. Y. W. . X. L. Adversarial t-shirt! evading person detectors in a physical world. *ECCV 2020: Computer Vision – ECCV 2020 pp 665–681*, 2020.

Kurakin, A., Goodfellow, I. J., and Bengio, S. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pp. 99–112. Chapman and Hall/CRC, 2018.

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

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection. 2015. URL http://arxiv.org/abs/1506.02640. cite arxiv:1506.02640.

Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

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

Thys, S., Van Ranst, W., and Goedemé, T. Fooling automated surveillance cameras: adversarial patches to attack person detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 0–0, 2019.

Wu, Z., Lim, S.-N., Davis, L. S., and Goldstein, T. Making an invisibility cloak: Real world adversarial attacks on object detectors. pp. 1–17, 2020.

Xie, C., Wang, J., Zhang, Z., Zhou, Y., Xie, L., and Yuille, A. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 1369–1378, 2017.