REAP: A LARGE-SCALE REALISTIC ADVERSARIAL PATCH BENCHMARK

Anonymous authors
Paper under double-blind review

ABSTRACT

Machine learning models are known to be susceptible to adversarial perturbation. One famous attack is the *adversarial patch*, a sticker with a particularly crafted pattern that makes the model incorrectly predict the object it is placed on. This attack presents a critical threat to cyber-physical systems that rely on cameras such as autonomous cars. Despite the significance of the problem, conducting research in this setting has been difficult; evaluating attacks and defenses in the real world is exceptionally costly while synthetic data are unrealistic. In this work, we propose the REAP (REalistic Adversarial Patch) Benchmark, a digital benchmark that allows the user to evaluate patch attacks on real images, and under real-world conditions. Built on top of the Mapillary Vistas dataset, our benchmark contains over 14,000 traffic signs. Each sign is augmented with a pair of geometric and lighting transformations, which can be used to apply a digitally generated patch realistically onto the sign, while matching real-world conditions. Using our benchmark, we perform the first large-scale assessments of adversarial patch attacks under realistic conditions. Our experiments suggest that adversarial patch attacks may present a smaller threat than previously believed and that the success rate of an attack on simpler digital simulations is not predictive of its actual effectiveness in practice.

1 INTRODUCTION

Research has shown that machine learning models lack robustness against adversarially chosen perturbations. The pioneering work of Szegedy et al. (2014) first demonstrated that one can engineer perturbations that are indiscernible to the human eye which can cause neural networks to misclassify images with high confidence. Since then, there has been a large body of academic work on understanding the robustness of neural networks to such attacks Goodfellow et al. (2015); Moosavi-Dezfooli et al. (2016); Tanay & Griffin (2016); Carlini & Wagner (2017); Kurakin et al. (2017); Tramèr et al. (2017); Madry et al. (2018); Bubeck et al. (2019); Ilyas et al. (2019).

One particularly concerning type of attack is the *adversarial patch attack* (Brown et al., 2018; Eykholt et al., 2018; Karmon et al., 2018; Sitawarin et al., 2018; Chen et al., 2019; Jan et al., 2019; Liu et al., 2019b; Patel et al., 2019; Sharma et al., 2019; Zhao et al., 2019; Huang et al., 2020; Wu et al., 2020). These are real-world attacks, where the objective of the attacker is to print out a patch, physically place it in a scene, and cause a vision network processing the scene to malfunction. These attacks are especially concerning because of the impact they could have on autonomous vehicles. The worry is that a malicious agent could, for instance, produce a sticker so that, when it is placed on a stop sign, a self-driving car would believe it is (say) a speed limit sign, and fail to stop. Indeed, similar attacks have already been demonstrated both in academic settings (Liu et al., 2019a; Evtimov et al., 2017; Sato et al., 2021), as well as on real-world autonomous vehicles (Tencent Keen Security Lab, 2019).

However, despite the significant risks that adversarial patch attacks pose, to a certain extent, research on these attacks has stalled. This is in large part because quantitatively evaluating the significance of this threat is challenging. The most accurate approach would be to conduct experiments in the real world, but this is very expensive, and at present, not practical to do at a large scale. (Compare to research on computer vision, where the availability of benchmarks such as ImageNet have reduced the barriers to research and spurred tremendous innovation.) As a result, it is difficult to accurately assess the effectiveness of new patch attacks and defenses against patch attacks.

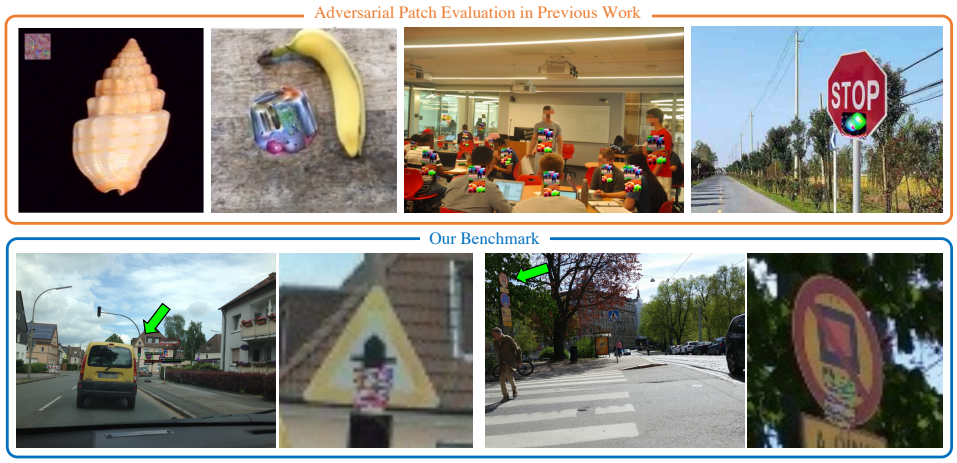


Figure 1: We provide a more realistic way to evaluate patch attacks. Past work (top row) ignores many real-world factors and thus might yield a misleading evaluation. We develop a benchmark (bottom row) that supports a more realistic simulation of the effect of a real-world patch attack on road signs, simulating the effect of the pose, location, and lighting on rendering and image capture.

Instead, researchers in this area turn to one of two techniques: either, they physically create their attacks and try them out on a small number of real-world examples by physically attaching them to objects, or they digitally evaluate patch attacks using digital images containing simulated patches. Both of these approaches have major drawbacks. Although the former approach simulates more realistic conditions, the sample size they obtain is very small, and typically one cannot draw statistical conclusions from the results. Additionally, because of the ad-hoc nature of these evaluations, it is impossible to compare the results across different papers. Ultimately, such experiments can only serve as a proof of concept that under some situations, the proposed attacks and/or defenses work, but do not serve as a rigorous evaluation of their effectiveness.

In contrast, a digital simulation of the attacks and/or defenses can allow us to obtain some quantitative measures. However, it is difficult to ensure that the simulation accurately captures all of the challenges that arise in the real world. Past work on defending against patch attacks has often made unrealistic assumptions, such as that the patch is square, axis-aligned, can be located anywhere on the image, and is fully under the control of the attacker, and has ignored many effects, such as noise and variation in lighting and pose. This is because evaluations have been done on standard computer vision datasets, which do not possess the tooling necessary for more realistic simulation. Consequently, it is unclear if the numbers obtained are actually reflective of what would happen in real-world scenarios. See Fig. 1 (top row) for examples of past works, where the patch is not constrained to be on the target object, does not respect the pose and shape of the object, and/or ignores the lighting conditions.

1.1 Our contributions

The REAP Benchmark: In this work, we propose the REalistic Adversarial Patch Benchmark (REAP), the first large-scale standardized benchmark for security against patch attacks, that has tooling which allows us to model realistic patch attacks. Given the significance of adversarial patch attacks for autonomous vehicles in particular, we choose to focus on computer vision systems for recognizing road signs. Motivated by the aforementioned shortcomings of prior evaluations, REAP was designed with the following principles in mind:

1. **Large scale evaluation:** REAP consists of a collection of 14,651 images of road signs drawn from the Mapillary Vistas dataset. This allows us to use REAP to draw quantitative conclusions about the effectiveness of attacks on the dataset.
2. **Realistic patch rendering:** REAP has tooling, which, for every road sign in the dataset, allows us to realistically render any digital patch onto the sign, matching factors such as where to place the patch, the camera angle, lighting conditions, etc. Importantly, this transformation is differentiable, and so one can still perform backpropagation through the rendering process.

3. **Realistic image distribution:** REAP consists of images of signs taken under realistic conditions, mimicking for instance what a self-driving car would see from its sensors. As a result, the images consist of road signs of many varying sizes and distances from the camera sensor, and under various lighting conditions and degrees of occlusion. We also take steps to minimize the distributional shift from the original Mapillary Vistas dataset.

We believe that REAP will help support research in patch attacks by enabling a more accurate evaluation of new attacks and defenses. We plan to release REAP publicly before the conference.

Evaluations with REAP: With our new benchmark in hand, we also perform the first large-scale evaluations of existing attacks on object detectors under realistic conditions. We consider the "per-class" threat model where the goal of the attacker is, for a single type of sign, to devise a patch that causes the object detector to fail to detect signs of that type. We evaluate existing attacks on two different object detection architectures, and we find the following conclusions:

1. **Existing patch attacks are not that effective.** Perhaps surprisingly, our first conclusion is that on our benchmark, existing attacks do not succeed on a majority of images. This is in contrast to simpler attack models such as ℓ_p-bounded perturbations where we can typically obtain nearly 100% attack success rate, and even to previously reported numbers for patch attacks on simpler benchmarks, which were much higher than what we see on REAP. This latter point is related to our next conclusion, which is:
2. **Performance on synthetic data is not reflective of performance on REAP.** We find that the success rates of attacks on synthetic versions of our benchmark and the full REAP are only poorly correlated. We conclude that performance on simple synthetic benchmarks is not predictive of attack success rate in more realistic conditions.
3. **Lighting and patch placement are particularly important.** Finally, we investigate what transforms in the patch rendering are the most important, in terms of the effect on the attack success rate. We find that the most significant first-order effects are from the lighting transform, as well as the positioning of the patch. In contrast, the perspective transforms—while still important—seem to affect the attack success rate somewhat less.

While we believe these conclusions are already quite interesting, they are only the tip of the iceberg of what can be done with REAP. We believe that going forward, the REAP benchmark can serve as a very useful tool for both understanding the power of physical world patch attacks, as well as for the development of better defenses against such attacks.

2 Related Work

The literature on adversarial patches, and adversarial attacks more generally, is vast and a full review is beyond the scope of this paper. For conciseness, here we will only survey the most relevant works. Since their introduction in Brown et al. (2018); Karmon et al. (2018); Eykholt et al. (2018), there have been a variety of adversarial patch attacks proposed in the literature (Sitawarin et al., 2018; Chen et al., 2019; Jan et al., 2019; Liu et al., 2019b; Patel et al., 2019; Sharma et al., 2019; Zhao et al., 2019; Huang et al., 2020; Wu et al., 2020). Of particular interest to us are the attacks on object detection of road signs (Eykholt et al., 2018; Chen et al., 2019; Zhao et al., 2019). There have also been a slew of proposed defenses for such attacks, see e.g., (Zhang et al., 2020; Xiang & Mittal, 2021; Rao et al., 2020; McCoyd et al., 2020; Mu & Wagner, 2021).

2.1 Existing Benchmarks and Evaluation

Existing evaluations of adversarial patch attacks and defenses fit into one of two categories:

Small scale, real-world tests. A common methodology used to test the transferability of the adversarial patch to the physical world is to print it out, physically place it onto an object, and capture pictures or videos of the patch for evaluation (Brown et al., 2018; Eykholt et al., 2018; Chen et al., 2019; Zhao et al., 2019; Hoory et al., 2020; Huang et al., 2020; Wu et al., 2020). While this method provides the most realistic evaluation of the adversarial patches, it has a number of downsides. First, this evaluation methodology is by nature very time-consuming and hence limits the number of images that can be used for testing. Consequently, one cannot extract quantitative conclusions from the

results of these tests. Additionally, the methodologies vary across different papers, and so it is hard to compare their results. For instance, the pictures of the adversarial patches are taken under different angles, lighting conditions, or from varying distances. Sometimes, the adversarial patches themselves are printed using different printers (Chen et al., 2019; Zhao et al., 2019).

Digital simulation. The other approach is to simulate the effects of the adversarial patch by digitally inserting it into the image. As mentioned previously, this has the advantage that it can be done more scalably. This has been done to varying degrees of sophistication. One of the most common, but also simplest, ways this is done is to simply apply the patch to the image at some random position, and with some simple transformations, for instance, those induced by expectation over transformation (Brown et al., 2018; Eykholt et al., 2018; Karmon et al., 2018; Liu et al., 2019b; Hoory et al., 2020; Zhang et al., 2020; Rao et al., 2020; McCoyd et al., 2020; Xiang & Mittal, 2021; Wang et al., 2021). In contrast, our benchmark positions the patches realistically, and with more realistic transformations.

Arguably the benchmarks most similar to the one we propose are the ones in Zhao et al. (2019) and Huang et al. (2020). In Zhao et al. (2019), they use a synthetic benchmark where stop signs with patches are digitally inserted into images with realistic camera angles. However, their benchmark does not account for lighting conditions. In contrast, all signs in our dataset are real, and we also produce a transformation to match lighting conditions. The former allows us to more realistically model how real-world agents would interact with signs, and the latter is particularly important since we find that matching lighting is particularly important to obtain results that reflect the effectiveness of patches in realistic conditions.

In Huang et al. (2020), they produce a benchmark of 20 fully virtual scenes, over which a user can control multiple cameras and lighting conditions. In these scenes, the user can then place adversarial patches and study their effectiveness under various perspective/lighting transformations. We view this benchmark as somewhat orthogonal to ours: their benchmark allows one to study the effectiveness of an attack in a very controlled space but on a small scale, whereas our benchmark allows for a quantitative measure of the effectiveness of the attack under realistic scenarios, in many different contexts. Also, they do not consider road signs and corresponding attacks on autonomous vehicles.

We also note that photorealistic simulation frameworks have been proposed to study the reliability of computer vision systems in related settings (Leclerc et al., 2021). However, to our knowledge, these frameworks do not have the tooling necessary for evaluating adversarial patches.

3 Adversarial Patch Benchmark

3.1 Overview

Our dataset consists of a collection of images containing traffic signs. Each sign comes with a segmentation mask, and we label them with a class (based on their shape and size).

So far, this is more or less standard. The main additional feature of our benchmark is that, for each sign, we also provide an associated rendering transformation. Given a digital patch, this transformation allows us to apply the patch on the sign in a way that respects the scaling, orientation, and lighting of the sign in the image. We emphasize that a separate transformation is inferred individually for each sign, in order to ensure that the transformation is accurate for every image. Moreover, the rendering transformation is fully differentiable, which allows our dataset to be used to generate patch attacks and to apply adversarial defenses along the line of adversarial training.

Fig. 2 and Fig. 4 give an overview of the process to obtain these transformations which we will describe in Section 3.4.1 and Section 3.4.2. We produce candidate annotations, visually inspect all of them individually, and manually fix any wrong annotation. In total, we label 14,651 traffic signs across 8,433 images.

3.2 Datasets

We build our benchmark using images from the Mapillary Vistas dataset (Neuhold et al., 2017).[1] The Mapillary dataset includes 20,000 street-level images from around the world which have been

[1] www.mapillary.com/dataset/vistas

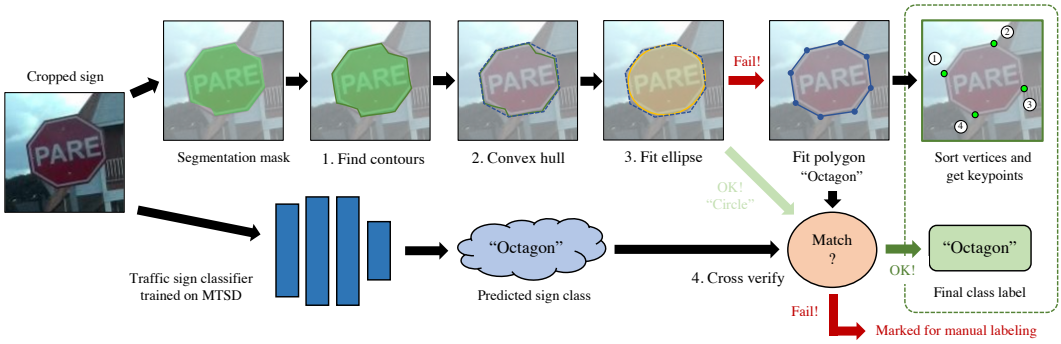


Figure 2: The automated procedure we use to extract the keypoints from each traffic sign.

annotated with bounding boxes of 124 object categories, including traffic signs. We use images from the Mapillary training and validation sets and discard images that do not contain any traffic sign.

3.3 Traffic Sign Classification

For our benchmark, we want to test adversarial patches on traffic signs of different shapes and sizes. Unfortunately, the Mapillary Vistas dataset does not include such labels, i.e., all traffic signs are grouped under one class. Instead of labeling these signs by hand, we train a ResNet-18 on another similar dataset, Mapillary Traffic Sign (MTSD) (Ertler et al., 2020), to classify samples from Mapillary Vistas. MTSD contains granular labeling of over 300 traffic sign classes with bounding box annotations, but we cannot use it in place of Vistas because it lacks segmentation labels.

We group the signs into 11 classes by shape and size, namely circle, triangle, upside-down triangle, diamond (S), diamond (L), square, rectangle (S), rectangle (M), rectangle (L), pentagon, and octagon. The remaining signs that do not belong to one of these classes are labeled as a background class or "others" which will be ignored when we compute the metrics. After training the ResNet-18, which achieves about 97% accuracy on the validation set of MTSD, we use it to infer the first set of candidate class labels of Mapillary Vistas.

In addition to extending the problem to a multi-class setting, we use sign classification to determine the physical size of different signs. Since we hope to specify the patch size in real units (e.g., inches) instead of pixels, this allows the patch to scale (approximately) correctly when applied to the signs. A limitation is that a single type of sign may come in different sizes, e.g., stop signs can be 24", 30", 36", or 48", depending on the type of road they are located on. We pick a canonical size for each type of sign based on the size specified for "Expressway" according to the official U.S. Department of Transportation's guideline.[2] Appendix A describes our design decision in detail.

3.4 Transformations

The traffic signs in our dataset vary in shape, size, rotation, and orientation. When digitally applying a patch to a sign, we first need to apply a perspective or 3D transform to the adversarial patch to simulate the orientation. Next, we take into account the fact that pictures of real-world traffic signs are taken under different lighting conditions by applying a relighting transform to the patch. The importance of these transformations is highlighted in Fig. 3.

3.4.1 Geometric Transformation

To determine the parameters of the perspective transform, we need *four* keypoints for each sign in our dataset. We infer the keypoints for a particular traffic sign using only its segmentation mask (which is provided in the Mapillary Vistas dataset) by following the four steps below (also visualized in Fig. 2):

1. *Find contours*: First, we find the edge or the contour of the segmentation mask.
2. *Compute convex hull*: Then, we find the convex hull of the contour to correct annotation errors and occlusion. This does not affect the already correct masks which should already be convex.

[2] https://mutcd.fhwa.dot.gov/htm/2003/part2/part2b1.htm

Figure 3: Example ablation of the geometric and relighting transforms in our dataset. The rightmost stop sign has a patch rendered with a perspective and relighting transform which makes it more realistic. The first and second images have patches that are too bright whereas the first and third images have patches that do not respect the sign's orientation.

3. *Fit polygon and ellipse*: We try to fit an ellipse to the convex hull, to find circular signs. If the fitted ellipse results in a larger error than some threshold, we know that the sign is not circular and therefore fit a polygon instead.
4. *Cross verify*: We verify that the shape obtained from the previous step matches with the ResNet's prediction. If not, the sign is flagged for manual inspection.

The last step is finding the keypoints. For polygons, we first match the vertices to the canonical ones and then simply take the four predefined vertices as the keypoints. For circular signs, we use the ends of their major and minor axes as the four keypoints. Then, we use these keypoints to infer a perspective transform appropriate for this sign. Triangular signs are a special case as we can only identify a maximum of three keypoints which means we can only infer a unique affine transform (six degrees of freedom). Note that this transform is a linear transformation, and hence is fully differentiable. Lastly, we manually check all annotations and correct any errors.

3.4.2 Relighting Transformation

Each traffic sign in our dataset has two associated relighting parameters, $\alpha, \beta \in \mathbb{R}$. Given a patch $\mathbf{P}$, its relighted version $\mathbf{P}_{\text{relighted}} = \alpha\mathbf{P} + \beta$ is rendered on the scene as depicted on the bottom row of Fig. 4. We infer α, β by matching the histogram of the original sign (e.g., the real stop sign on the upper-right of Fig. 4) to a canonical image (e.g., the synthetic stop sign on the upper-left): in particular, we set β as the 10th percentile of all the pixel values (aggregated over all three RGB channels) on that sign and α as the difference between the 10th and 90th percentile. In doing so, we make an assumption that the relighting can be approximated with a linear transform where α and β represent contrast and brightness adjustments, respectively. We choose the 10th and the 90th percentiles, instead of the 1st and the 99th, to filter noise in the real image. Finally, note that like before, since this transformation is linear, it is differentiable.

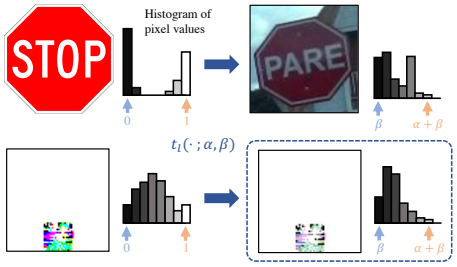


Figure 4: Computing relighting parameters (top) and applying the transform (bottom).

4 Experiments on our Benchmark

Our benchmark can be used to evaluate attacks and defenses under various threat models, e.g., making objects appear vs disappear, using a universal patch vs a targeted attack, etc. In this paper, we pick one typical setting where the adversary tries to make a traffic sign disappear or be misclassified.

4.1 Experiment Setup

Traffic sign detectors. We experiment with two object detection models: Faster R-CNN (Ren et al., 2015) and YOLOv5 (Jocher et al., 2022). Both models are trained on the MTSD dataset to predict bounding boxes for all 11 traffic sign classes plus one background class. We follow the training method and hyperparameters from Neuhold et al. (2017) to train both models. The performance of

Table 1: Performance of the models on two different datasets without attacks. "MTSD" represents the test set of the MTSD dataset on which the models are trained. mAP is the mean average precision, commonly used to represent performance in object detection tasks. We follow COCO's method for computing mAP (Lin et al., 2014). mAP_w is the class-weighted version analogous to mFNR and mFNR_w which are defined in Section 4.2.

Datasets	Faster R-CNN				YOLOv5			
	mFNR ↓	mFNR_w ↓	mAP ↑	mAP_w ↑	mFNR ↓	mFNR_w ↓	mAP ↑	mAP_w ↑
MTSD	29.9	29.7	55.0	56.0	17.2	14.7	69.3	71.3
Our Benchmark	16.2	17.4	67.0	64.4	14.2	14.5	69.7	69.6

the models on benign data is reported in Table 1. As mentioned in Section 4.2, we report the false negative rate (FNR) in addition to the commonly used mAP scores. For FNR, the score threshold is chosen as one that yields the maximum F1 score on the validation set of MTSD.

Synthetic Benchmark. We use canonical synthetic signs, one per class, as a baseline to compare our REAP benchmark to. Similarly to Eykholt et al. (2018), the synthetic sign is placed at a random location in the image and randomly rotated between 0 and 15 degrees. The synthetic benchmark can be used for both generating and testing the adversarial patch. For testing, we use a total of 5,000 images randomly selected from our REAP benchmark.

Attack algorithms. We use the RP2 attack (Eykholt et al., 2018) to generate adversarial patches for YOLOv5, and the Shapeshifter attack (Chen et al., 2019) for Faster R-CNN, as each attack was created for a specific type of model. We generate one patch per one sign class. We assume that the adversary has access to 50 held-out images from our benchmark. Each of the 11 classes in our dataset has a specific set of 50 images which are all guaranteed to contain at least one sign of the class that is being attacked. Thus, an adversarial patch for a sign can be generated using these 50 images from our benchmark. We also generate an adversarial patch using a synthetic sign from that class, as has been done in prior work. Finally, we use whichever patch performs better (i.e., has a higher attack success rate). For more detail on the setup, please see Appendix B.

4.2 Evaluation Metrics

Here, we define *a successful attack* as a patch that makes the sign either (i) undetected or (ii) classified to a wrong class (i.e., one of the other 11 classes, or the background class). Similarly to previous work, we measure the effectiveness of an attack by the *attack success rate* (ASR), defined as follows. Given a list of signs $\{x_i\}_{i=1}^N$ and the corresponding perturbed version (i.e., with an adversarial patch applied to it) $\{x_i'\}_{i=1}^N$,

$$\text{Attack Success Rate} = \frac{\sum_{i=1}^N \mathbf{1}_{x_i \text{is detected}} \ \wedge \ \mathbf{1}_{x_i' \text{is not detected}}}{\sum_{i=1}^N \mathbf{1}_{x_i \text{is detected}}}. \tag{1}$$

Additionally, we also report false negative rate (FNR), which is simply the fraction of signs that the model fails to detect and classify correctly. We report both metrics for each class of the signs as well as their average (mFNR and mASR). We also compute an average of the metrics weighted by the number of samples in each class (mFNR_w and mASR_w).

4.3 Results

Patch attacks against road signs are less effective than previously believed. From Table 2, a 10"×10" adversarial patch only succeeds for about 18% and 25% of the signs on our realistic benchmark. Even though we use undefended models that were normally trained without any special data augmentation that would enhance the robustness, attacks are not very effective. For comparison, a universal adversarial perturbation under ℓ_2 and ℓ_∞ norms achieves above 80% success rate (Moosavi-Dezfooli et al., 2017).

Table 2: ASR and FNR of the adversarial patches on the two traffic sign detectors. One adversarial patch is generated per class of traffic signs. For sign-specific metrics, see Table 5.

Patch Size	Benchmarks	Faster R-CNN				YOLOv5			
		mFNR	mFNR_w	mASR	mASR_w	mFNR	mFNR_w	mASR	mASR_w
No patch	Synthetic	11.8	17.4	n/a	n/a	10.8	9.6	n/a	n/a
	Ours	16.2	17.4	n/a	n/a	14.2	14.5	n/a	n/a
Small (10”×10”)	Synthetic	54.4	67.3	49.6	61.8	75.9	73.6	72.2	70.4
	Ours	35.1	30.5	24.7	17.7	46.1	32.5	44.4	24.9
Medium (10”×20”)	Synthetic	69.7	87.3	67.7	85.6	88.2	90.8	85.0	88.3
	Ours	47.9	42.6	40.5	32.4	60.1	48.6	58.7	42.6
Large (two 10”×20”)	Synthetic	98.1	98.1	99.5	99.5	100	100	100	100
	Ours	79.3	84.2	76.2	81.7	85.9	90.0	85.0	89.3

Figure 5: Examples of Small (10" x 10"), Medium (10" x 20") and Large (two 10" x 20") patches applied to random signs from our benchmark.

A 10”×10” patch is likely easily noticeable to humans, so such an attack might be easily detected and might soon be removed. Adversarial attacks are most troubling when they are imperceptible; patches as large as 10”×10” (or larger) are likely to draw attention, which may make them less of a threat in practice. As shown in Fig. 7, nearly all sign classes seem to resist attack at least to some extent.

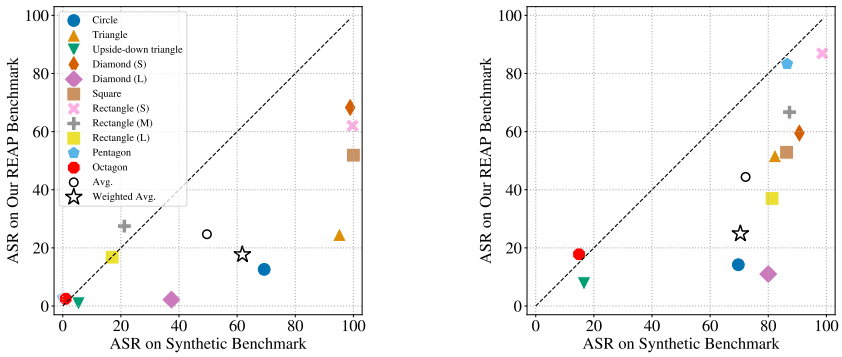
Our findings are consistent with prior works that investigate physical-world attacks on stop signs. In these works, the attack is often clearly visible. For instance, Eykholt et al. (2018) and Zhao et al. (2019) use a patch that is close to our two 10”×20” patches which is why they observe a high attack success rate similar to our results with the larger patch size.

We also experimented with a “per-image” attack where we generate one adversarial patch for each instance of traffic sign, as opposed to one patch per class (see Table 9). ASR for the “per-image” attack is much higher than a per-class patch (40% relative increase on average). While such attacks are harder to generate and might be more fragile in practice, this result suggests that better attack algorithms may exist and that a specifically targeted adversary could still be an important threat.

ASR measured on synthetic data is not predictive of ASR measured on our realistic benchmark. We compared our benchmark to a synthetic benchmark intended to be representative of methodology often found in prior work: we take a single synthetic image of a road sign, then generate attacks against it (instead of a real image). Table 2 and Fig. 6 show that there is a large difference between metrics as measured on such a synthetic benchmark compared to our benchmark. The gap can be up to 50 percentage points on average or even up to 5× difference for the smallest patch size.

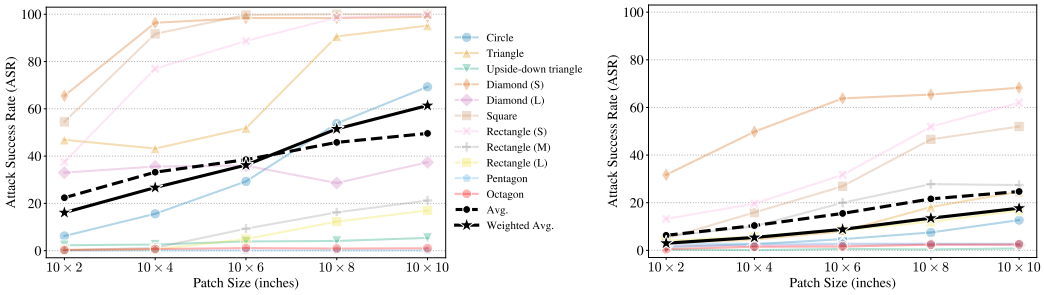
Fig. 6 and Table 5 in Appendix C compares ASR on the two benchmarks by class of the traffic signs. If the two ASRs were similar, all data points would be close to the diagonal dashed line. Instead, most of the data points are below the line, suggesting that the synthetic benchmark consistently overestimates the ASR. Moreover, there is no clear relationship between the two measurements of ASR. The ordering is not preserved, and the gap varies significantly among different sign classes.

The lighting transform affects the attack’s effectiveness more than the geometric transform. Table 3 shows results from an ablation study showing how the transformations our benchmark applies to the patch affect its ASR. For both YOLOv5 and Faster R-CNN, our realistic lighting transform has a much larger effect than the geometric transform. Without the lighting transform, the average ASR increases by approximately 10 percentage points for YOLOv5, and 15 pp for Faster R-CNN. This observation explains why the synthetic benchmark as well as synthetic evaluations in previous works overestimate ASR, as prior works do not consider lighting.



(a) Faster R-CNN (b) YOLOv5

Figure 6: Comparison of ASR on synthetic benchmark vs on our realistic benchmark for the two models. The dashed line marks the points with an equal ASR on both synthetic and our benchmarks.



(a) Synthetic benchmark (b) Our realistic benchmark

Figure 7: ASR of all sign classes at different patch sizes on Faster R-CNN. ASR is higher on the synthetic benchmark than our benchmark, for all patch sizes and almost every traffic sign class.

Table 3: Attack success rate on our realistic benchmark when other choices of transformations are applied, for some representative classes of signs. "No 3D" means the perspective transform is replaced by only translation and scaling. "No light" means there is no relighting transform, but we still apply the perspective transform. The full table is included in the supplementary material.

Models	Transforms	Circles	Triangles	Diamonds(L)	Squares	Rectangles(L)	Octagons	*mASR*	$mASR_w$
Faster R-CNN	No 3D	13.0	24.8	2.7	55.2	17.2	2.7	24.7	18.0
	No light	30.8	44.1	6.3	62.5	47.6	6.1	36.5	32.3
	Ours	12.7	24.6	2.2	52.0	16.7	2.5	24.7	17.7
YOLOv5	No 3D	16.6	49.8	11.4	61.3	39.9	16.0	46.0	26.8
	No light	26.7	60.0	14.8	62.8	55.0	22.6	52.0	34.6
	Ours	14.2	51.7	11.0	52.9	37.0	17.8	44.4	24.9

5 Conclusion

We construct the first large-scale benchmark for evaluating adversarial patches. Our benchmark consists of over 14,000 signs from real driving scenes, and each sign is annotated with the transformations necessary to render an adversarial patch realistically onto it. Using this benchmark, we experiment with adversarial patch attacks on two object detectors. We find that adversarial patches of a clearly visible size fool undefended models on less than 25% of the signs, where attacks are much more successful. This is in contrast to similar settings such as adversarial examples with bounded ℓ_p-norms. We hope that our benchmark will provide a foundation for more realistic evaluation of patch attacks and drive future research on defenses against them.

REFERENCES

Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv:1712.09665 [cs]*, May 2018. 1, 3, 4

Sebastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 831–840. PMLR, June 2019. 1

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57, 2017. doi: 10.1109/SP.2017.49. 1

Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng (Polo) Chau. ShapeShifter: Robust physical adversarial attack on faster R-CNN object detector. In Michele Berlingerio, Francesco Bonchi, Thomas Gärtner, Neil Hurley, and Georgiana Ifrim (eds.), *Machine Learning and Knowledge Discovery in Databases*, pp. 52–68, Cham, 2019. Springer International Publishing. ISBN 978-3-030-10925-7. 1, 3, 4, 7

Christian Ertler, Jerneja Mislej, Tobias Ollmann, Lorenzo Porzi, and Yubin Kuang. The mapillary traffic sign detection and classification around the world. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 5

Ivan Evtimov, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. Robust physical-world attacks on machine learning models. 2017. 1

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramèr, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Physical adversarial examples for object detectors. In *12th USENIX Workshop on Offensive Technologies (WOOT 18)*, Baltimore, MD, August 2018. USENIX Association. 1, 3, 4, 7, 8

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. 1

Shahar Hoory, Tzvika Shapira, Asaf Shabtai, and Yuval Elovici. Dynamic adversarial patch for evading object detection models. *arXiv:2010.13070 [cs]*, October 2020. 3, 4

Lifeng Huang, Chengying Gao, Yuyin Zhou, Cihang Xie, Alan L. Yuille, Changqing Zou, and Ning Liu. Universal physical camouflage attacks on object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 3, 4

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 1

Steve T.K. Jan, Joseph Messou, Yen-Chen Lin, Jia-Bin Huang, and Gang Wang. Connecting the digital and physical world: Improving the robustness of adversarial attacks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:962–969, July 2019. ISSN 2374-3468, 2159-5399. doi: 10.1609/aaai.v33i01.3301962. 1, 3

Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, TaoXie, Kalen Michael, Jiacong Fang, imyhxy, Lorna, Colin Wong, Zeng Yifu, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Max Strobel, Mrinal Jain, Lorenzo Mammana, and xylieong. Ultralytics/yolov5: V6.2 - yolov5 classification models, apple M1, reproducibility, ClearML and deci.ai integrations. Zenodo, August 2022. 6

Danny Karmon, Daniel Zoran, and Yoav Goldberg. LaVAN: Localized and visible adversarial noise. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2507–2515. PMLR, July 2018. 1, 3, 4

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv:1607.02533 [cs, stat]*, February 2017. 1

Guillaume Leclerc, Hadi Salman, Andrew Ilyas, Sai Vemprala, Logan Engstrom, Vibhav Vineet, Kai Xiao, Pengchuan Zhang, Shibani Santurkar, Greg Yang, et al. 3db: A framework for debugging computer vision models. *arXiv preprint arXiv:2106.03805*, 2021. 4

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision – ECCV 2014*, pp. 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1. 7

Aishan Liu, Xianglong Liu, Jiaxin Fan, Yuqing Ma, Anlan Zhang, Huiyuan Xie, and Dacheng Tao. Perceptual-sensitive gan for generating adversarial patches. In *AAAI Conference on Artificial Intelligence*, 2019a. 1

Xin Liu, Huanrui Yang, Ziwei Liu, Linghao Song, Hai Li, and Yiran Chen. DPatch: An adversarial patch attack on object detectors. *arXiv:1806.02299 [cs]*, April 2019b. 1, 3, 4

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 1

Michael McCoyd, Won Park, Steven Chen, Neil Shah, Ryan Roggenkemper, Minjune Hwang, Jason Xinyu Liu, and David Wagner. Minority reports defense: Defending against adversarial patches. In Jianying Zhou, Mauro Conti, Chuadhry Mujeeb Ahmed, Man Ho Au, Lejla Batina, Zhou Li, Jingqiang Lin, Eleonora Losiouk, Bo Luo, Suryadipta Majumdar, Weizhi Meng, Martín Ochoa, Stjepan Picek, Georgios Portokalidis, Cong Wang, and Kehuan Zhang (eds.), *Applied Cryptography and Network Security Workshops*, pp. 564–582, Cham, 2020. Springer International Publishing. ISBN 978-3-030-61638-0. 3, 4

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. DeepFool: A simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 7

Norman Mu and David Wagner. Defending against adversarial patches with robust self-attention. In *Workshop on Uncertainty and Robustness in Deep Learning*, 2021. 3

Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5000–5009, Venice, October 2017. IEEE. ISBN 978-1-5386-1032-9. doi: 10.1109/ICCV.2017.534. 4, 6

Naman Patel, Prashanth Krishnamurthy, Siddharth Garg, and Farshad Khorrami. Adaptive adversarial videos on roadside billboards: Dynamically modifying trajectories of autonomous vehicles. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5916–5921, November 2019. doi: 10.1109/IROS40897.2019.8968267. 1, 3

Sukrut Rao, David Stutz, and Bernt Schiele. Adversarial training against location-optimized adversarial patches. In Adrien Bartoli and Andrea Fusiello (eds.), *Computer Vision – ECCV 2020 Workshops*, pp. 429–448, Cham, 2020. Springer International Publishing. ISBN 978-3-030-68238-5. 3, 4

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pp. 91–99, Cambridge, MA, USA, 2015. MIT Press. 6

Takami Sato, Junjie Shen, Ningfei Wang, Yunhan Jia, Xue Lin, and Qi Alfred Chen. Dirty road can attack: Security of deep learning based automated lane centering under physical-world attack. In *USENIX Security*, 2021. 1

Prinkle Sharma, David Austin, and Hong Liu. Attacks on machine learning: Adversarial examples in connected and autonomous vehicles. In *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, pp. 1–7, 2019. doi: 10.1109/HST47167.2019.9032989. 1, 3

Chawin Sitawarin, Arjun Nitin Bhagoji, Arsalan Mosenia, Mung Chiang, and Prateek Mittal. DARTS: Deceiving autonomous cars with toxic signs. *arXiv:1802.06430 [cs]*, May 2018. 1, 3

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. 1

Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv:1608.07690 [cs, stat]*, August 2016. 1

Tencent Keen Security Lab. Experimental Security Research of Tesla Autopilot, 2019. URL https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Research_of_Tesla_Autopilot.pdf. 1

Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. *arXiv:1704.03453 [cs, stat]*, May 2017. 1

Yajie Wang, Haoran Lv, Xiaohui Kuang, Gang Zhao, Yu-an Tan, Quanxin Zhang, and Jingjing Hu. Towards a physical-world adversarial patch for blinding object detection models. *Information Sciences*, 556:459–471, 2021. ISSN 0020-0255. doi: 10.1016/j.ins.2020.08.087. 4

Zuxuan Wu, Ser-Nam Lim, Larry S. Davis, and Tom Goldstein. Making an invisibility cloak: Real world adversarial attacks on object detectors. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (eds.), *Computer Vision – ECCV 2020*, pp. 1–17, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58548-8. 1, 3

Chong Xiang and Prateek Mittal. PatchGuard++: Efficient provable attack detection against adversarial patches. *arXiv:2104.12609 [cs]*, April 2021. 3, 4

Zhanyuan Zhang, Benson Yuan, Michael McCoyd, and David Wagner. Clipped BagNet: Defending against sticker attacks with clipped bag-of-features. In *2020 IEEE Security and Privacy Workshops (SPW)*, pp. 55–61, San Francisco, CA, USA, May 2020. IEEE. ISBN 978-1-72819-346-5. doi: 10.1109/SPW50608.2020.00026. 3, 4

Yue Zhao, Hong Zhu, Ruigang Liang, Qintao Shen, Shengzhi Zhang, and Kai Chen. Seeing isn't believing: Towards more robust adversarial attack against real world object detectors. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, pp. 1989–2004, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 978-1-4503-6747-9. doi: 10.1145/3319535.3354259. 1, 3, 4, 8

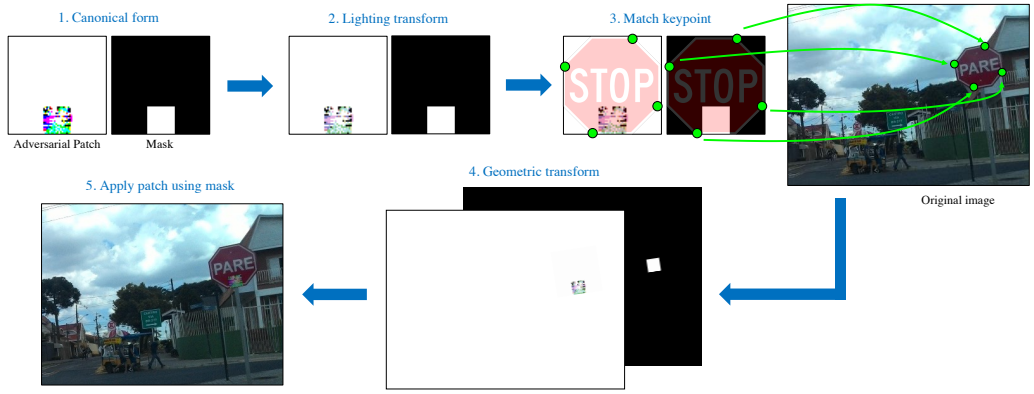


Figure 8: REAP's procedure for applying lighting and geometric transforms to a digitally generated adversarial patch.

Table 4: Dimension of the sign by classes in the REAP benchmark.

Traffic Sign Class	Width (mm)	Height (mm)	Number of Samples in REAP
Circle	750	750	7971
Triangle	900	789	636
Upside-down triangle	1220	1072	824
Diamond (S)	600	600	317
Diamond (L)	915	915	1435
Square	600	600	1075
Rectangle (S)	458	610	715
Rectangle (M)	762	915	544
Rectangle (L)	915	1220	361
Pentagon	915	915	133
Octagon	915	915	637

A Additional Details of the Benchmark

A.1 Traffic Sign Classification

To train the ResNet-18 for classifying the traffic signs, we first collect the cropped signs from the MTSD dataset. The cropped signs leave 10% of padding on each side of the sign size and are resized to 128×128 pixels. We trained the ResNet-18 with a batch size of 128, a learning rate of 0.1, and a weight decay of 5×10^{-4}. The categorization of the signs along with their dimension and number of samples are in Table 4 and Fig. 9.

A.2 Applying the Transforms

Fig. 8 summarizes the steps to apply an adversarial patch using our REAP benchmark. Given a patch and a corresponding mask with respect to the canonical sign, we first apply relighting transform on the patch. Then, we use the annotated keypoints of the target sign to determine the parameters of the perspective transform which is then applied to both the patch and the mask. Throughout this paper, we use bilinear interpolation for any geometric transform. Finally, the transformed patch is applied to the image using the transformed mask.

To be precise, let $\boldsymbol{X}$, $\boldsymbol{P}$, and $\boldsymbol{M}$ denote the original image, the adversarial patch, and the patch mask, respectively. The final image $\boldsymbol{X}'$ is obtained by the following equation

$$\boldsymbol{X}' = t_g\left(\boldsymbol{M}\right) \odot t_g\left(t_l\left(\boldsymbol{P}\right)\right) + \left(1 - t_g\left(\boldsymbol{M}\right)\right) \odot \boldsymbol{X} \tag{2}$$

where $t_g(\cdot)$ and $t_l(\cdot)$ are the geometric and the relighting transforms which in fact, depend on the annotated parameters associated with $\boldsymbol{X}$.

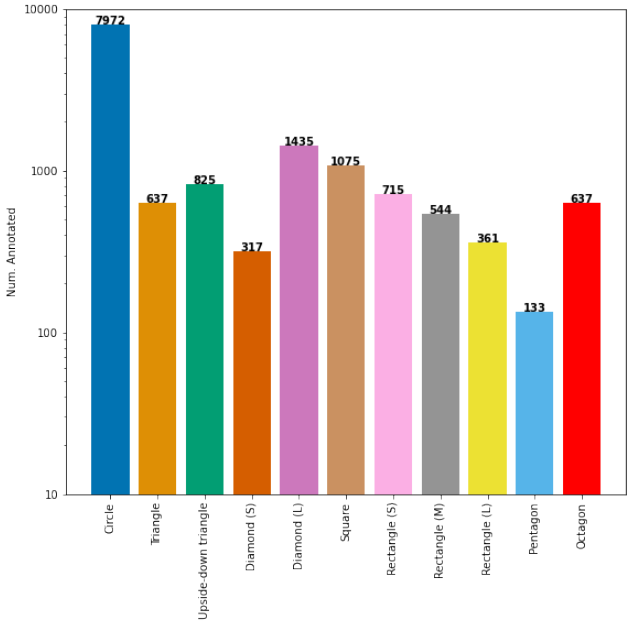


Figure 9: Class distribution of the traffic signs in our REAP benchmark.

We note that the mask is concatenated to the patch, and both are applied with the same geometric transform and interpolation. Therefore, $t_g(\boldsymbol{M})$ is no longer a binary mask like $\boldsymbol{M}$. This creates an effect where the transformed patch blends in more cleanly with the sign than the nearest interpolation does. Additionally, we also clip the pixel values after applying each transform to ensure that they always stay between 0 and 1.

B Detailed Experiment Setup

Here, we describe hyperparameters of the attack as well as the benchmarks. We re-implement both the RP2 and the ShapeShifter attacks based on the description provided in the paper. The ShapeShifter attack has an official and publicly available implementation in Tensorflow so we are able to compare our code to theirs directly.[3]

For both attacks, our default hyperparameters include 64×64 patch and object dimension, EoT rotation with a maximum 15 degrees, no color jitter for EoT. We use Adam optimizer with a step size of 0.01 for 1,000 iterations. The λ parameter used to encourage low-frequency patterns is set to 10^{-5}. We find that the choices of the patch dimension and λ do not affect the ASR when varying in a reasonable range. We use the same set of hyperparameters when generating adversarial patches from both our REAP and the synthetic benchmark.

C Additional Experiments

ASR by Classes for All Patch Sizes. Table 5, Table 6, and Table 7 contain a breakdown of ASR by sign class for the three patch sizes in Table 2. It is evident that the synthetic data do not overestimate the ASR on average but consistently across almost every traffic sign class. The trend is also consistent for all patch sizes. The gap becomes narrower for the two 10×20 patches as the ASR reaches 100% on both the synthetic and our benchmarks. However, it is almost undeniable that a patch of this size covers the majority of the sign area and is, in no way, inconspicuous.

ASR under varying hyperparameters of the synthetic benchmark. We conduct an additional ablation study to compare the effects of the hyperparameters of the synthetic benchmark as mentioned

[3] https://github.com/shangtse/robust-physical-attack

Table 5: Attack success rates by sign classes under synthetic vs our REAP benchmarks. "*Avg*" is an average, and "*WAvg*" is a weighted average by the number of samples in each class in our benchmark. The patch size is $10'' \times 10''$.

Models	Bench	Circ	Tri	UTri	Dia(S)	Dia(L)	Squ	Rec(S)	Rec(M)	Rec(L)	Pen	Oct	*Avg*	*WAvg*
Faster R-CNN	Syn	69.3	95.2	5.4	98.9	37.4	100.0	99.7	21.2	17.0	0.1	1.0	49.6	61.8
	REAP	12.7	24.6	0.8	68.3	2.2	52.0	62.0	27.4	16.7	2.8	2.5	24.7	17.7
YOLOv5	Syn	69.7	82.3	16.6	90.7	80.0	86.3	98.6	87.3	81.3	86.5	14.9	72.2	70.4
	REAP	14.2	51.7	7.7	59.5	11.0	52.9	86.9	66.7	37.0	83.3	17.8	44.4	24.9

Table 6: Attack success rates by sign classes under synthetic vs our REAP benchmarks with the patch size is 10"$\times$20".

Models	Bench	Circ	Tri	UTri	Dia(S)	Dia(L)	Squ	Rec(S)	Rec(M)	Rec(L)	Pen	Oct	*Avg*	*WAvg*
Faster R-CNN	Syn	98.0	99.7	0.3	100.0	75.7	100.0	100.0	92.7	71.0	1.6	5.4	67.7	85.6
	REAP	27.8	63.2	0.6	89.7	7.1	64.7	87.4	54.8	39.2	6.9	3.6	40.5	32.4
YOLOv5	Syn	96.2	95.0	21.6	99.8	71.3	100.0	100.0	97.3	99.2	99.6	55.3	85.0	88.3
	REAP	36.7	69.4	7.8	78.0	13.1	82.6	92.4	78.8	58.0	91.3	37.9	58.7	42.6

in Appendix B. Fig. 10 shows a similar scatter plot to Fig. 6a but with all the hyperparameters we have swept. This plot further strengthens the conclusions that the synthetic benchmark overestimates ASR and that it is not predictive of the ASR on our REAP benchmark. These observations persist across all the hyperparameter choices.

Additionally, Fig. 10 demonstrates that there is a large variation in the ASRs measured by the synthetic benchmark when the hyperparameters vary. For instance, changing the rotation can affect the ASR up to 20%–40% for many signs. This emphasizes that results reported on a synthetic benchmark are sensitive to its hyperparameters, and we should take special care when using one. On the other hand, our REAP benchmark does not have a similar set of hyperparameters to sweep over since the transformations as well as the sign sizes are fixed with respect to each image.

An ablation study on the transforms used in our REAP benchmark. Here, we report the full results by class of Table 3 in Table 8. The relighting transform still affects the effectiveness of the patch attack to a greater degree than the geometrics transform.

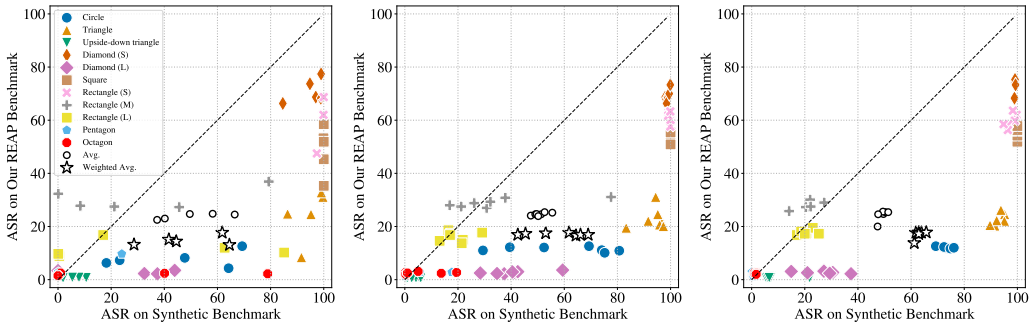
Per-image attack. As another ablation study, we experiment with the worst-case possible attack on our benchmark where the adversary can generate a unique adversarial patch for each image and is also aware of how the patch will appear in the image exactly. This setting is similar to the commonly studied "white-box" attack in the adversarial example literature. This threat model is particularly unrealistic for patch attacks because, in the real world, the adversary cannot predict apriori how the video or the image of the patch will be taken. Nonetheless, theoretically, this measurement is useful because the ASR in this setting should be the upper bound of any other setting including the "per-class" threat model we have considered throughout the paper.

Table 9 reports the per-image ASR on our REAP benchmark. We only compute the ASR for Faster R-CNN because this experiment is computationally expensive even when we reduce the attack iterations from 1,000 to 200. This experiment takes about 10 days to finish on an Nvidia GTX V100 GPU. On average, the per-image attack results in about 10 percentage points higher ASR than the per-class attack. This is a significant increase (about 40% relatively).

However, in the absolute sense, the ASR is only 35%, i.e., the patch attack only succeeds about one-third of the time in the worst-case scenario. There are two ways to interpret this observation: first, it could mean that an object detection model may be more robust to physical attacks than the researchers expect, and this makes coming up with an effective defense easier. The second way to view this result is that the previously proposed attack algorithms are far from optimal, and there is a large room for improvement from the attacker's side.

Table 7: Attack success rates by sign classes under synthetic vs our REAP benchmarks with two patches of size 10”×20”.

Models	Bench	Circ	Tri	UTri	Dia(S)	Dia(L)	Squ	Rec(S)	Rec(M)	Rec(L)	Pen	Oct	*Avg*	*WAvg*
Faster R-CNN	Syn	100	100	99.9	100	100	100	100	100	79.6	100	100	98.1	99.5
	REAP	88.2	86.7	20.1	99.6	53.5	99.7	100.0	95.5	40.5	63.9	90.3	76.2	81.7
YOLOv5	Syn	100	100	100	100	100	100	100	100	100	100	100	100	100
	REAP	97.7	85.3	37.6	99.0	64.5	98.6	100.0	98.4	58.0	98.4	97.9	85.0	89.3



(a) Synthetic object size (b) Rotation degrees (c) Color jitter intensity

Figure 10: A scatter plot similar to Fig. 6a where each point denotes a pair of ASRs measured by the synthetic and our REAP benchmark for each class of the signs and also for each hyperparameter choice for the synthetic benchmark. Particularly, we sweep three hyperparameters that control (a) patch and object dimension, (b) the range of rotation degree used in EoT, and (c) the color jitter intensity used in EoT.

D Additional Visualization of the Benchmark

Table 8: Attack success rate on our realistic benchmark when other choices of transformations are applied. "No 3D" means the perspective transform is replaced by only translation and scaling. "No light" means there is no relighting transform, but we still apply the perspective transform.

Models	Transform	Circ	Tri	UTri	Dia(S)	Dia(L)	Squ	Rec(S)	Rec(M)	Rec(L)	Pen	Oct	*mASR*	$mASR_w$
Faster R-CNN	No 3D	13.0	24.8	0.3	62.6	2.7	55.2	62.0	25.6	17.2	5.6	2.7	24.7	18.0
	No light	30.8	44.1	1.5	77.0	6.3	62.5	80.8	41.7	47.6	2.8	6.1	36.5	32.3
	Ours	12.7	24.6	0.8	68.3	2.2	52.0	62.0	27.4	16.7	2.8	2.5	24.7	17.7
YOLOv5	No 3D	16.6	49.8	8.1	57.2	11.4	61.3	90.6	66.2	39.9	88.9	16.0	46.0	26.8
	No light	26.7	60.0	8.5	70.1	14.8	62.8	92.6	72.1	55.0	86.5	22.6	52.0	34.6
	Ours	14.2	51.7	7.7	59.5	11.0	52.9	86.9	66.7	37.0	83.3	17.8	44.4	24.9

Table 9: Attack success rate of the *per-image* attack on our realistic benchmark. The model is Faster R-CNN, and the patch size is $10'' \times 10''$.

Metrics	Circ	Tri	UTri	Dia(S)	Dia(L)	Squ	Rec(S)	Rec(M)	Rec(L)	Pen	Oct	*Avg.*	*WAvg.*
FNR	34.2	42.4	7.6	87.5	16.6	76.1	94.1	53.9	42.5	15.4	9.6	43.6	37.9
ASR	20.1	35.3	1.7	86.4	3.7	72.5	88.1	44.2	22.0	8.3	5.0	35.2	26.4
AP	46.3	55.7	63.9	29.4	65.1	26.3	3.9	46.2	56.2	61.5	77.5	48.4	47.1

(a) Circle

(b) Triangle

(c) Diamond (L)

(d) Square

(e) Rectangle (L)

(f) Octagon

Figure 11: Examples of images from our benchmark after applying the 10”×10” patch. The sub-caption indicates the target sign class. We try to select images that the signs are large enough to see on the printed paper.