REAP: A Large-Scale Realistic Adversarial Patch Benchmark

Nabeel Hingun*
UC Berkeley
nabeel126@berkeley.edu

Chawin Sitawarin*
UC Berkeley
chawins@berkeley.edu

Jerry Li
Microsoft
jerrl@microsoft.com

David Wagner
UC Berkeley
daw@cs.berkeley.edu

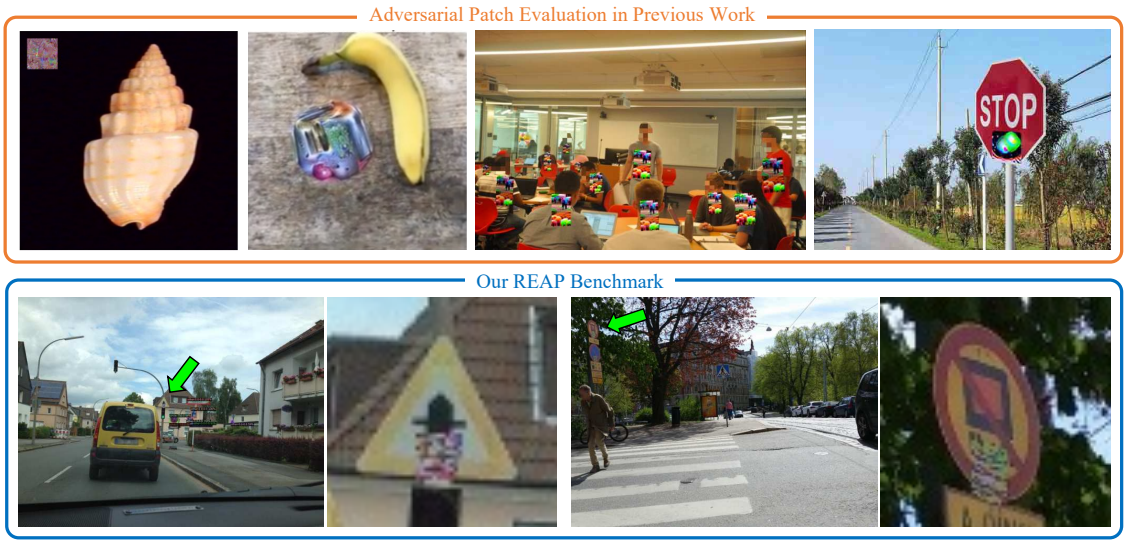


Figure 1: When digitally evaluating patch attacks, past work (top row) ignores many real-world factors and thus may yield a misleading evaluation. We develop the REAP benchmark (bottom row) that more realistically simulates the effect of a real-world patch attack on road signs, accounting for the pose, the location, and the lighting condition.

Abstract

Machine learning models are known to be susceptible to adversarial perturbation. One famous attack is the adversarial patch*, a particularly crafted sticker that makes the model mispredict 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 enables the evaluations on real images 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 geometric and lighting transformations for applying a digitally generated patch realistically onto the sign. Using our benchmark, we perform the first large-scale assessments of adversarial patch attacks under realistic conditions. Our experiments suggest that 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. Our benchmark is released publicly at* https://github.com/wagner-group/reap-benchmark.

1. Introduction

Research has shown that machine learning models lack robustness against adversarially chosen perturbations. Szegedy et al. [54] first demonstrated that one can engineer perturbations that are indiscernible to the human eye yet that cause neural networks to misclassify images with high confidence.

*Equal contribution.

Since then, there has been a large body of academic work on understanding the robustness of neural networks to such attacks [18, 39, 55, 8, 27, 57, 36, 7, 22].

One particularly concerning type of attack is the *adversarial patch attack* [6, 17, 24, 53, 10, 23, 33, 43, 52, 66, 21, 61]. These are real-world attacks, where the attacker's objective 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 potential impact on autonomous vehicles. A malicious agent could, for instance, produce a sticker that, when placed on a stop sign, cause a self-driving car to believe it is (say) a speed limit sign, and fail to stop. Indeed, similar attacks have already been demonstrated both in academic settings [31, 16, 50] and on real-world autonomous vehicles [56].

Despite this significant risk, research on these attacks has stalled to a certain extent because quantitatively evaluating the significance of this threat is challenging. The most accurate approach would be to conduct real-world experiments, but they are very expensive, and at present, not practical to do at a large scale. This leaves much to be desired compared to other branches of computer vision research, where the availability of benchmarks such as ImageNet have reduced the barriers to research and spurred tremendous innovation.

Instead, researchers 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 approaches have major drawbacks. Although the former simulates more realistic conditions, the sample size is very small, and typically one cannot draw statistical conclusions from the results [6, 17, 53, 10, 66, 20, 21, 61]. Additionally, because of the ad-hoc nature of these evaluations, it is impossible to compare the results across different papers. Ultimately, such experiments only serve as a proof of concept for the proposed attacks and defenses, but not as a rigorous evaluation of their effectiveness.

In contrast, a digital simulation of attacks/defenses allows quantitative evaluation [24, 33, 65, 46, 37, 62, 58, 44]. However, it is difficult to accurately capture all of the challenges that arise in the real world. Past work often made unrealistic assumptions, such as that the patch is square, axis-aligned, can be located anywhere on the image, fully under the control of the attacker, and ignore noise and variation in lighting and pose (see top row of Fig. 1). Consequently, it is unclear if these evaluations are actually reflective of what would happen in real-world scenarios.

1.1. Our Contributions

The REAP Benchmark: We propose REalistic Adversarial Patch Benchmark (REAP), the first large-scale standardized benchmark for security against patch attacks. Motivated by the aforementioned shortcomings of prior evaluations, we design REAP with the following principles in mind:

1. **Large-scale evaluation:** REAP consists of 14,651 images of road signs drawn from the Mapillary Vistas dataset. This allows us to draw quantitative conclusions about the effectiveness of attacks/defenses 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 fast and differentiable so one can still perform backpropagation through the rendering process.
3. **Realistic image distribution:** REAP consists of images taken under realistic conditions, including variation in sizes and distances from the camera as well as various lighting conditions and degrees of occlusion.

Evaluations with REAP: With our new benchmark in hand, we also perform the first large-scale evaluations of existing attacks on object detectors. We evaluate existing attacks on three different object detection architectures: Faster R-CNN [48], YOLOF [9], and DINO [64]. We also implement and evaluate a baseline defense adapted from adversarial training [36] to defend against patch attacks on object detection. The conclusions we find are:

1. **Existing patch attacks are not that effective.** Perhaps surprisingly, existing attacks do not succeed on a majority of images on our benchmark. This is in contrast to simpler attack models such as ℓ_p-bounded perturbations or patch attacks on simpler benchmarks, where the attack success rate is near 100%. Moreover, adversarially trained models can almost completely stop the attacks with only a minor performance drop on benign data.
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 REAP will help support future research in adversarial patches by enabling a more accurate evaluation of new attacks and defenses.

2. Related Work

Adversarial patch attacks. 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, we only survey the most relevant works. Since their introduction in Brown et al. [6], Karmon et al. [24], Eykholt et al. [17], there have been a variety of adversarial patch attacks proposed [23, 33, 43, 52, 21, 61]. Of particular interest to us are the ones on object detection of road signs [17, 53, 10, 66].

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 [6, 17, 53, 10, 66, 20, 21, 61]. While this method provides the most realistic evaluation, it has a number of downsides. First, it 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. Additionally, they are difficult to standardize across papers, making their result not directly comparable. 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 [10, 66].

Completely simulated environment. Another line of work considers purely simulated environments for evaluating adversarial patches such as CARLA [14, 32, 45] and AttackScenes [21]. A huge advantage of this method is that it has the most precise and the most flexible control of the environment, e.g., cameras and objects can be placed anywhere. However, it is labor-intensive to build a diverse set of scenes digitally, and it compromises heavily on realism. Another example is 3DB [28], a photorealistic simulation for studying the reliability of computer vision systems. Nevertheless, it lacks the tooling necessary for evaluating adversarial patches and does not contain any driving scene, a setting to which adversarial patches are most applicable. Our benchmark utilizes images of real and diverse driving scenes and focuses on realistically simulating only the adversarial patches.

Digital simulation. This third approach takes a middle road and simulates the effects of the adversarial patch by digitally inserting it into a real image. This has been done at scale and to varying degrees of sophistication. One of the most common, but also simplest, ways this is done is to apply the patch to the image at some random position, and with some simple transformations, for instance, those induced by expectation over transformation [6, 24, 33, 65, 46, 37, 62, 58, 44]. This approach violates all the physical constraints and hence, is far from being realistic.

Arguably the benchmarks most similar to ours are the ones in Zhao et al. [66] and Braunegg et al. [5]. Zhao et al. [66] digitally insert synthetic stop signs with patches into images with realistic camera angles. However, they do not account for lighting conditions, and the target object itself is synthetic. In contrast, all signs in our dataset are real, and we also produce a transformation to match lighting conditions. In Section 4.3, we find that these two factors affect the evaluation metrics to a large extent. APRICOT [5] contains images of real scenes with a printed adversarial patch. Compared to ours, APRICOT is smaller in size (1K vs 14K images) and is heavily inflexible as it comes with a pre-defined adversarial patch with a fixed size and location.

Defenses. There have also been a slew of proposed defenses against patch attacks, e.g., [19, 41, 65, 62, 46, 37, 40, 11]. Most examine object classification. Only a handful consider object detection, which may be more relevant in practice [12, 63]. We choose to experiment with adversarial training [36] as a defense because, to the extent of our knowledge, it has not been applied in this setting (Rao et al. [46] study patch adversarial training on classifiers). It is also known to be a strong baseline and arguably the only effective defense across other ℓ_p-norms [13]. Importantly, unlike the other defenses listed above, adversarial training does not make assumptions about the number or size of the patch.

3. Adversarial Patch Benchmark

3.1. Overview

Our dataset is a collection of images containing traffic signs, each of which comes with a segmentation mask and a class. 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.

Figs. 2 and 3 give an overview of the process to obtain the *geometric* (Section 3.4.1) and the *relighting* (Section 3.4.2) transformations, respectively. We use an algorithm to generate the candidate annotations automatically, visually inspect each of them, and then 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 [42]. It includes 20,000 street-level images from around the world, annotated with bounding boxes of 124 object categories, including traffic signs. A limitation of Vistas is that all traffic signs are grouped under one class.

This creates a challenge for us, because our patch-rendering process depends on the size and shape of the sign. Without this information, the rendering is less realistic. We deal with this challenge by grouping the signs so that all signs in the same group can use the same geometric transform procedure. The grouping process is described in the next section.

3.3. Traffic Sign Classification

Grouping traffic signs by their shape and size has two advantages. First, it allows more accurate geometric transforms as previously mentioned. Second, it allows us to study multi-class sign detection. Instead of labeling the Vistas signs by hand, we train a ResNet-18 on a similar dataset, Mapillary Traffic Sign (MTSD) [15], to classify them. MTSD contains granular labeling of over 300 traffic sign classes, but we cannot use it in place of Vistas as it lacks segmentation labels required to compute the geometric transforms.

We created two versions of the benchmark: REAP and REAP-S. REAP is our main benchmark with the classes matching those of MTSD. However, most of the classes contain fewer than 10 samples so we only keep the 100 most common classes. We need a sufficient number of samples per class because (i) some will be further filtered out in the preprocessing and (ii) the samples will be split into a "training" (for the attacker to "train" the adversarial patch) and a test set (for evaluating the attack). Conversely, REAP-S groups 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. REAP-S serves as a simpler alternative to REAP and is also intuitively more "defender-friendly." For both REAP and REAP-S, the remaining signs that do not belong to the classes are labeled as a background class or "others" which will be ignored when we compute the metrics.

Since Mapillary Vistas does not come with these labels, we first trained a ConvNeXt model [35] on MTSD, which achieves about 98% accuracy on the validation set, to generate the candidate class labels. The labels were then automatically corrected when we compute the parameters for the geometric transform in Section 3.4.1. For the remaining ones that cannot be automatically verified, we manually inspected and corrected them.

Our grouping of the signs in REAP-S has an extra benefit. Since each class is (approximately) associated with a standardized physical size, we can specify the patch size in real units (e.g., inches) instead of pixels. The real unit is arguably more useful for estimating the threat of adversarial patches than constraining the size by the number of pixels. One complication is that a single class of sign may come in different sizes, e.g., stop signs can be 24", 36", or 48", depending on the kind of road they are located on, but usually one size is more common. The Vistas dataset does not contain sufficient information to distinguish between these sizes so we pick one canonical size for each sign type. Specifically, we select the size specified for "Expressway" according to the official U.S. Department of Transportation's guideline. Appendix A describes our design decision in detail.

3.4. Transformations

We render adversarial patches with two types of transformations: *geometric* and *relighting*. Since the traffic signs in our dataset vary in shape, size, and orientation, we first need to apply a geometric, specifically perspective or 3D, transform to the patch to simulate these variations. Next, we account for 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. 4.

3.4.1 Geometric Transformation

To determine the parameters of the perspective transform, we need four keypoints for each sign. 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 contour**: First, we find 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 correct masks, as they should already be convex.
3. **Fit polygon and ellipse**: We fit an ellipse to the convex hull, to find circular signs. If the fitted ellipse results in a larger error than a certain 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 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. These keypoints are used 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 linear and hence is fully differentiable. Lastly, we manually check all annotations and correct any errors.

Update August 18, 2023: In the previous version of the paper, we only present REAP-S which was called REAP.

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

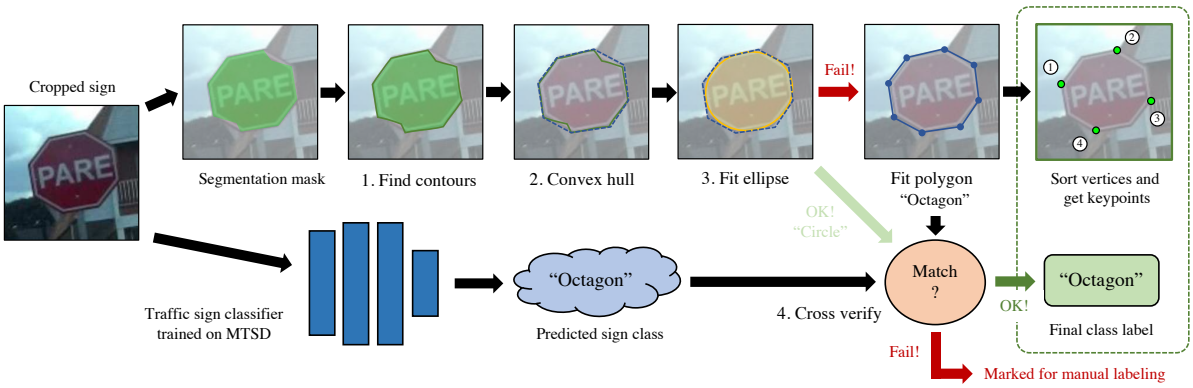


Figure 2: The automated procedure we use to extract the keypoints from each traffic sign.

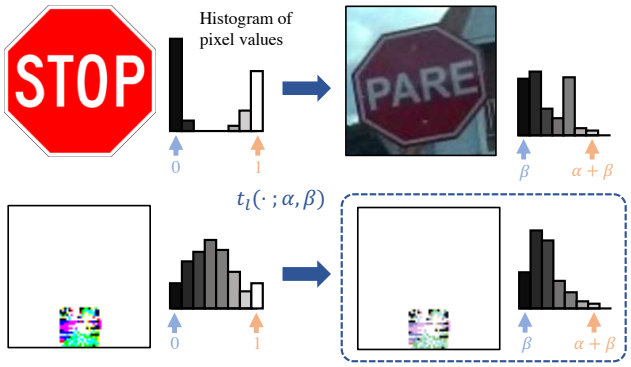


Figure 3: Computing relighting parameters (top) and applying the transform (bottom).

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. 3. We infer α, β by matching the histogram of the original sign (e.g., the real stop sign on the upper-right of Fig. 3) to a canonical image (e.g., the synthetic stop sign on the upper-left): in particular, we set β as the p-th percentile of all the pixel values (aggregated over all three RGB channels) on that sign and α as the difference between the p-th and $(1-p)$-th percentile. We call this the "percentile" method and explain why we chose it in Section 3.5. This method assumes that relighting can be approximated with a linear transform where α and β represent contrast and brightness adjustments. Like before, since this transformation is linear, it is differentiable.

3.5. Realism Test

In this section, we measure how realistic the patches are when rendered with different transform methods: three for geometric and eight for relighting. The geometric transforms include perspective (or homographic), affine, and translate & scale transforms. For relighting, we experiment with three methods, each of which can be carried out on different color spaces (RGB, HSV, and LAB): the percentile method, described in Section 3.4.2; polynomial fitting, where we find the polynomial that best fits the pixel values on each real sign given the corresponding pixel values on the digital one; and *Color Transfer* [47], which tries to match the mean and

Figure 4: 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.

Table 1: Comparison of different geometric and relighting transforms from our realism test (mean ± standard deviation of RMSE across 44 samples). The best results are in bold.

Transforms	Methods	Colors	RMSE (↓)
Geometric	Translate & Scale	n/a	1.72 ± 1.19
	Affine	n/a	1.35 ± 0.49
	Perspective (3D)	n/a	**1.13 ± 0.41**
Relighting	Percentile	RGB	**0.110 ± 0.034**
		HSV	0.227 ± 0.118
		LAB	0.652 ± 0.112
	Polynomial	RGB	0.113 ± 0.037
		HSV	0.118 ± 0.035
		LAB	0.161 ± 0.043
	Color Transfer	HSV	0.117 ± 0.035
		LAB	0.184 ± 0.062

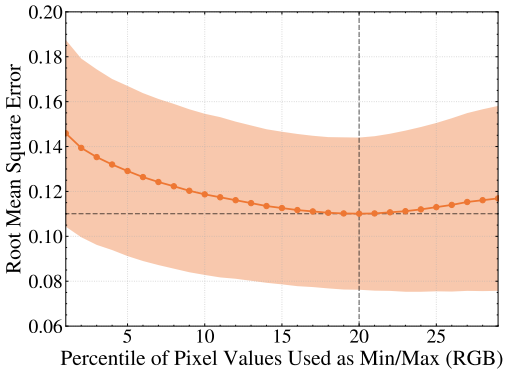


Figure 5: RMSE between the photographed and the rendered patches using the "percentile" method with different values of p. The shaded region denotes the standard deviation across 44 samples. $p = 20$ yields the lowest RMSE.

Figure 6: Random samples used in our realism experiments (left: real, right: rendered). Fig. 17 contains all samples.

the standard deviation of the pixel values.

We photograph 44 pairs of *real* traffic signs with and without an adversarial patch: 11 signs, one for each class, in four scenes and lighting conditions. For each sample, we hand-annotated the keypoints of both the signs and the patches. Then, given an image of the sign without the patch, we render the patch on it using the different transform methods. For geometric transforms, we measure the root mean square error (RMSE) between the rendered and the corresponding groundtruth *corners* of the patch. To compare the relighting methods, we compute the RMSE between the rendered and the corresponding groundtruth *pixel values* of the patch. We use the groundtruth geometric transform when computing the relighting parameters to disentangle the potential error from the geometric transform.

Table 1 reports the best RMSE achieved by each transform after a hyperparameter sweep (p, polynomial degrees, etc.). The perspective transform achieves the lowest RMSE as expected which emphasizes the importance of using the full 3D transform instead of simpler alternatives. For relighting, the percentile method with $p = 0.2$ performs as well as, or better than, any other at rendering the adversarial patches. Hence, these are the two transforms we use to construct the REAP benchmark and in all of the experiments in Section 4 unless stated otherwise. Fig. 6 visually compares the rendered patches with the groundtruth ones under these best transform methods. Appendix B contains more details.

4. Experiments on REAP 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 focus on the setting where the adversary tries to make a traffic sign *disappear* or be *misclassified* using the *per-class* attack, i.e., only one patch per class of objects, similar to Benz et al. [4]. We argue that this threat model is more realistic and more alarming as the attacker only needs to distribute several adversarial stickers that are effective across million of traffic signs. We assume the adversary has access to the target model (white-box).

4.1. Experiment Setup

Traffic sign detectors. We experiment with three object detection models, Faster R-CNN [48], YOLOF [9], and DINO [64], all trained on the MTSD dataset to predict bounding boxes for all 11 traffic sign classes plus the "other" class. We follow the training method and hyperparameters from Neuhold et al. [42]. As mentioned in Section 4.2, we report the false negative rate (FNR) in addition to mAP scores. For FNR, the score threshold is chosen as one that maximizes the F1 score on the validation set of MTSD.

Attack algorithms. We use the RP2 attack [17] and the DPatch attack [33] to generate adversarial patches for all models. We assume that the adversary has access to 5 held-out images from our benchmark and use them to generate **one adversarial patch per sign class**. We note that this setting, referred to as *per-class* attack, is different from the usual white-box attack threat model where each sample is given a unique perturbation (we call it *per-instance* attack) and is more similar to the "universal" adversarial perturbation [38]. We argue that this threat model is more realistic and more alarming as the attacker only needs to distribute one adversarial sticker that are effective across million of traffic signs of the same type. Appendix D.5 discusses the threat models in more detail.

Each of the classes has a specific set of these 5 images each of which contains at least one sign of that class. For REAP-S, we use 50 images since there are more samples per class. In practice, an adversary may benefit from using more than 5 (or 50) images to generate the patch, but we set our limit here to leave sufficient samples for the evaluation phase. We do not find a significant difference in the performance of the two attacks (Appendix D.1) so we report only the results of DPatch attack with PGD optimizer in the main paper.

Defense algorithm. We use adversarial training with DPatch attack and five-step PGD as it performs the best empirically. The patches are generated *per-instance* at a random location to prevent overfitting to a specific one. To

Table 2: Mean FNR and mAP of the adversarial patches on six traffic sign detectors on REAP. "FRCNN" refers to Faster R-CNN, "Adv." indicates adversarially trained models. For defenders, lower FNR (↓) and higher mAP (↑) are better.

Patch Size	FRCNN		YOLOF		DINO		Adv. FRCNN		Adv. YOLOF		Adv. DINO	
	FNR	mAP	FNR	mAP	FNR	mAP	FNR	mAP	FNR	mAP	FNR	mAP
No patch	4.3	72.9	18.5	54.8	14.1	68.2	3.1	73.3	21.0	55.0	9.4	74.2
Small (10"×10")	15.4	59.4	33.7	43.5	32.0	60.4	3.8	71.8	22.5	54.7	1.8	80.6
Medium (10"×20")	22.4	46.5	42.7	36.6	35.4	52.6	6.1	66.8	27.1	51.9	1.2	80.1
Large (two 10"×20")	50.0	18.2	72.8	19.4	62.8	39.5	13.9	56.3	57.7	34.1	3.6	77.8

improve the effectiveness of adversarial training under a small number of steps, we cache patches generated in the previous epoch and use it as an initialization for the next one [67]. For more detailed setup and results, please see Appendix C.

Synthetic Benchmark. We use canonical synthetic signs, one per class, as a baseline to compare our REAP-S benchmark to (we cannot find canonical synthetic signs for all 100 classes of REAP). Similarly to Eykholt et al. [17], the synthetic sign is placed at a random location on one of 50 random background images and randomly rotated between 0 and 15 degrees. We use the synthetic benchmark for both generating and testing the adversarial patch. For testing, we use 2,000 background images per class, randomly selected from our REAP-S benchmark to keep the distribution of the scenes similar.

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., any of the other 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 version with an adversarial patch applied to it, $\{x_i'\}_{i=1}^N$,

$$\text{ASR} = \frac{\sum_{i=1}^N \mathbf{1}_{x_i \text{is detected}} \wedge \mathbf{1}_{x_i' \text{is not detected}}}{\sum_{i=1}^N \mathbf{1}_{x_i \text{is detected}}}. \quad (1)$$

ASR and FNR are easy to interpret but dependent on specific thresholds of both the confidence score and the IoU between the groundtruth and the detected boxes. Hence, we also report mAP which averages across these thresholds.

4.3. Main Results

Experiments on our REAP benchmark illuminate several findings that were not previously observed due to the lack of scalability and reproducibility of real-world experiments:

(1) Patch attacks against road signs are less effective than previously believed. From Table 2, a 10"×10" adversarial patch increases FNR by only 11–18 percentage points on the undefended models. For REAP-S (Table 3), the increase is 8–12 percentage points. For comparison, a class-wise adversarial perturbation under imperceptible ℓ_∞ norms achieves above 90% success rate [4]. More importantly, **on adversarially trained models, FNR remains almost identical before and after applying the patch** on Faster R-CNN and YOLOF. Surprisingly, adversarially trained DINO performs better on samples with adversarial patches than without. We hypothesize that this is a result of overfitting to some adversarial patches and not a clear sign of weak attacks or gradient obfuscation. We refer to Section 4.4 for additional detail.

This result implies that a well-known defense like adversarial training is effective and may be sufficient to protect against patch attacks in the real world. 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.

Our findings are also 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. [17] and Zhao et al. [66] 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. Nonetheless, a patch of this size surely breaches all notions of imperceptibility. Perhaps an interesting threat model to study in the future is to allow large patches but additionally constrain the perturbation with ℓ_∞-norm.

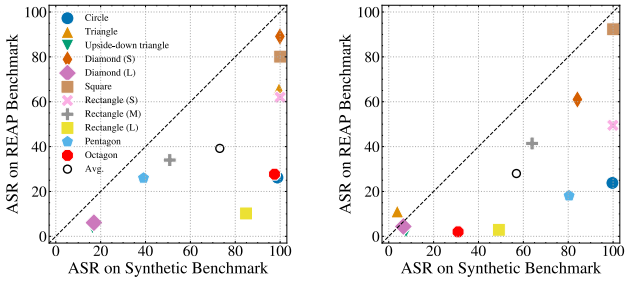
(2) ASR measured on synthetic data is not predictive of ASR measured on our realistic benchmark. We compare REAP-S to a synthetic benchmark intended to be representative of methodology often found in prior work: we take a single synthetic image of a road sign, then generate attacks against it (instead of a real image). Table 3 and Fig. 8 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 50–60 percentage points on average.

Fig. 8 and Table 9 in Appendix D compare ASR on the two benchmarks by class of the traffic signs. If the two ASRs were similar, all data points would have lied close to the diagonal dashed line. Instead, most of the data points

Table 3: Mean ASR and FNR of the adversarial patches on the six traffic sign detectors on REAP-S. For sign-specific metrics, see Table 9. It is clear that evaluating on the synthetic signs overestimates the attack's potency in every setting. For defenders, lower FNR (↓) and ASR (↓) are better.

Patch Size	Benchmarks	FRCNN		YOLOF		DINO		Adv. FRCNN		Adv. YOLOF		Adv. DINO	
		FNR	ASR	FNR	ASR	FNR	ASR	FNR	ASR	FNR	ASR	FNR	ASR
No patch	Synthetic	19.8	n/a	17.0	n/a	12.7	n/a	15.7	n/a	19.0	n/a	5.8	n/a
	REAP-S (ours)	20.2	n/a	17.1	n/a	12.8	n/a	17.4	n/a	19.2	n/a	6.1	n/a
Small (10"×10")	Synthetic	76.9	73.1	89.8	88.6	58.8	56.9	50.0	43.9	76.8	73.4	24.1	22.6
	REAP-S (ours)	50.5	39.2	48.6	38.9	36.2	28.0	18.7	5.1	28.3	12.7	1.1	0.1
Medium (10"×20")	Synthetic	89.9	88.3	92.0	91.1	73.1	72.6	79.5	77.3	83.7	81.7	34.7	33.9
	REAP-S (ours)	64.4	56.1	60.5	52.8	45.5	38.4	33.8	23.5	46.5	34.7	1.3	0.1
Large (two 10"×20")	Synthetic	99.6	99.6	100.0	100.0	96.5	96.4	98.9	98.8	99.1	98.9	52.9	52.7
	REAP-S (ours)	85.2	82.0	88.2	86.1	85.1	83.5	59.3	53.6	69.9	64.3	5.1	4.3

Figure 7: Examples of small (10"×10"), medium (10"×20") and large (two 10"×20") patches applied to three of the signs from our benchmark. The large patch size is clearly visible and obscures the notion of imperceptibility. We still choose to experiment with it since it is approximately the same size used by prior work [17, 66].



(a) Faster R-CNN (b) DINO

Figure 8: ASRs on synthetic vs REAP-S benchmarks for Faster R-CNN (left) and DINO (right). The dashed line marks the points with an equal ASR on both datasets.

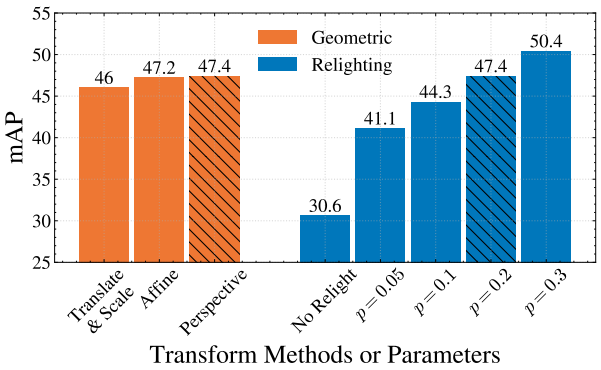


Figure 9: Effects of different geometric and relighting transform methods on the mAP of the YOLOF model under attack. The hatched bars are the default setting. When either geometric or relighting transform is varied, the other one is fixed to the default (perspective, $p = 0.2$).

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. If the *rankings* of the ASRs are well-correlated, we should expect ordering of the points to be similar in both horizontal and vertical directions, but this is not the case.

(3) The lighting transform affects the attack's effectiveness more than the geometric transform. Fig. 9 shows how the transformations our benchmark applies to the patch affect its mAP scores. For all models, our realistic lighting transform has a much larger effect than the geometric transform. Without the lighting transform, mAP decreases by 17 percentage points for YOLOF and 15 for Faster R-CNN (i.e., an increase of 23 and 14 points on the ASR). This observation explains why the synthetic benchmark as well as synthetic evaluations in previous works overestimate ASR.

4.4. Extended Attack Evaluation

Because these results were so surprising, we investigated the possibility that our attack algorithms are not sufficiently

Table 4: Robustness of the adversarially trained models under different attack threat models (10"×10" patch size). The per-instance attack has the highest ASR and the lowest mAP as expected, and there is no sign of gradient obfuscation.

Attacks	ASR (↑)	mAP (↓)
Adv. Faster R-CNN		
No Attack	n/a	66.0
Per-Class Attack	5.1	65.7
Per-Instance Attack	**16.0**	**59.3**
Transfer from YOLOF	3.2	67.8
Transfer from Adv. YOLOF	7.0	63.6
Transfer from Synthetic	2.7	69.1
Adv. YOLOF		
No Attack	n/a	58.5
Per-Class Attack	17.7	51.3
Per-Instance Attack	**28.2**	**46.5**
Transfer from Faster R-CNN	13.5	53.1
Transfer from Adv. Faster R-CNN	7.9	55.4
Transfer from Synthetic	12.2	54.3
Adv. DINO		
No Attack	n/a	65.7
Per-Class Attack	0.1	75.1
Per-Instance Attack	**2.7**	**63.7**
Transfer from Adv. Faster R-CNN	0.1	76.5
Transfer from Adv. YOLOF	0.2	76.1
Transfer from DINO	0.0	79.6
Transfer from Synthetic	0.4	72.7

Augment	Strength	FRCNN		Adv. YOLOF		Adv. DINO	
		FNR	mAP	FNR	mAP	FNR	mAP
None	n/a	15.4	59.4	22.5	54.7	1.2	80.1
Color-jitter	0.1	16.1	58.8	23.1	54.7	1.2	80.1
	0.2	16.0	58.5	23.6	54.6	1.2	80.0
	0.3	15.5	59.0	23.3	54.7	1.3	80.1
Unif. noise	0.1	15.7	58.9	23.0	54.8	1.2	80.4
	0.2	15.4	58.8	22.6	54.7	1.4	80.1
	0.3	15.6	58.6	22.9	54.7	1.4	80.2

Table 5: FNR and mAP with color-jitter or random noise applied during the attack EoT on REAP. We use the small patch for Faster R-CNN and Adv. YOLOF, and the medium size for Adv. DINO. None of the augmentations seems to affect the potency of the attack.

strong (e.g., gradient obfuscation [2]) or that the adversarially trained models "catastrophically overfit" [59, 25, 1], i.e., they memorize the attack patterns during training but are not actually robust. In particular, we evaluate the adversarially trained models against *transfer* and *per-instance* attacks.

The per-instance attack generates one patch for each instance of traffic signs, as opposed to our default per-class patch. The transfer attack generates per-class patches from either a different source model or the synthetic data. Table 4 shows that the per-instance attack always achieves a higher ASR (and lower mAP) than the per-class attack, and the transfer attack has the lowest ASR in most cases. This result is expected and does not indicate that the gradient obfuscation or the catastrophic overfitting phenomenon is happening.

To further improve the robustness of the adversarial patches (i.e., making the attack transfer to other instances in the same class), we also tried to generate the adversarial patches by applying random augmentations including color-jitter and uniform noise injection (similar to expectation over transformation [3]). Table 5 shows that the augmentations with varying strength levels do not significantly affect the ASR of the attack.

Overall, the effectiveness of all the attacks remains limited against all the adversarially trained models. In particular, the ASR of the per-instance attack, which is an *upper bound* of ASR on all threat models, is only 3% on Adv. DINO on REAP-S. Based on these experiments (and others in Appendix D.6), we tentatively conclude that the adversarially trained classifiers truly do appear robust for the REAP detection task. Because this result is so surprising, further research is needed before we can have full confidence in this conclusion.

5. Conclusion and Future Directions

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 a broad range of models and attacks. We find that adversarial patches of a clearly visible size fool an undefended model on less than 28% of the signs and only 1% for a defended model. This is in contrast to adversarial examples with bounded ℓ_p-norm, where attacks nearly always succeed. **All in all, our experiments suggest that realistic constraints render patch attacks significantly less effective, and vanilla adversarial training is an effective defense against the current practical patch attacks.**

One interesting direction for future research is to explore whether attacks against object detectors can be improved. Also, in our experiments, adversarial training achieved strong robustness at the cost of degrading mAP on clean images by about 5 percentage points. It would be interesting to explore new defenses that have less impact on clean performance. 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

[1] Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. In *Advances in Neural Information Processing Systems*, volume 33, pages 16048–16059. Curran Associates, Inc., 2020. 9

[2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 274–283, Stockholmsmässan, Stockholm Sweden, July 2018. PMLR. 9

[3] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 284–293, Stockholmsmässan, Stockholm Sweden, July 2018. PMLR. 9

[4] Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon. Universal adversarial training with class-wise perturbations. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2021. 6, 7

[5] A. Braunegg, Amartya Chakraborty, Michael Krumdick, Nicole Lape, Sara Leary, Keith Manville, Elizabeth Merkhofer, Laura Strickhart, and Matthew Walmer. APRICOT: A dataset of physical adversarial attacks on object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 35–50, Cham, 2020. Springer International Publishing. 3

[6] Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv:1712.09665 [cs]*, May 2018. 2, 3

[7] Sebastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 831–840. PMLR, June 2019. 2

[8] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017. 2

[9] Qiang Chen, Yingming Wang, Tong Yang, Xiangyu Zhang, Jian Cheng, and Jian Sun. You only look one-level feature. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 6, 22

[10] 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, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 52–68, Cham, 2019. Springer International Publishing. 2, 3

[11] Zitao Chen, Pritam Dash, and Karthik Pattabiraman. Turning your strength against you: Detecting and mitigating robust and universal adversarial patch attacks. *arXiv preprint arXiv:2108.05075*, 2021. 3

[12] Edward Chou, Florian Tramèr, and Giancarlo Pellegrino. SentiNet: Detecting localized universal attacks against deep learning systems. In *2020 IEEE Security and Privacy Workshops (SPW)*, pages 48–54, May 2020. 3

[13] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. RobustBench: A standardized adversarial robustness benchmark. Technical report, 2020. 3

[14] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 3

[15] 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. 4

[16] 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. 2

[17] 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, Aug. 2018. USENIX Association. 2, 3, 6, 7, 8, 22

[18] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. 2

[19] Jamie Hayes. On visible adversarial perturbations & digital watermarking. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1678–16787, June 2018. 3

[20] Shahar Hoory, Tzvika Shapira, Asaf Shabtai, and Yuval Elovici. Dynamic adversarial patch for evading object detection models. *arXiv:2010.13070 [cs]*, Oct. 2020. 2, 3

[21] 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. 2, 3

[22] 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, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 2

[23] 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. 2, 3

[24] Danny Karmon, Daniel Zoran, and Yoav Goldberg. LaVAN: Localized and visible adversarial noise. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2507–2515. PMLR, July 2018. 2, 3

[25] Hoki Kim, Woojin Lee, and Jaewook Lee. Understanding catastrophic overfitting in single-step adversarial training. In

Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 8119–8127, 2021. 9

[26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 22

[27] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv:1607.02533 [cs, stat]*, Feb. 2017. 2

[28] 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. 3

[29] Mark Lee and Zico Kolter. On physical adversarial patches for object detection, June 2019. 22

[30] 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, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing. 16

[31] 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*, 2019. 2

[32] Xiruo Liu, Shibani Singh, Cory Cornelius, Colin Busho, Mike Tan, Anindya Paul, and Jason Martin. Synthetic dataset generation for adversarial machine learning research, July 2022. 3

[33] 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]*, Apr. 2019. 2, 3, 6, 22

[34] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 22

[35] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 4, 18

[36] 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. 2, 3

[37] 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, editors, *Applied Cryptography and Network Security Workshops*, pages 564–582, Cham, 2020. Springer International Publishing. 2, 3

[38] 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. 6

[39] 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. 2

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

[41] Muzammal Naseer, Salman Khan, and Fatih Porikli. Local gradients smoothing: Defense against localized adversarial attacks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1300–1307, 2019. 3

[42] 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)*, pages 5000–5009, Venice, Oct. 2017. IEEE. 3, 6

[43] 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)*, pages 5916–5921, Nov. 2019. 2, 3

[44] Maura Pintor, Daniele Angioni, Angelo Sotgiu, Luca Demetrio, Ambra Demontis, Battista Biggio, and Fabio Roli. ImageNet-patch: A dataset for benchmarking machine learning robustness against adversarial patches. *Pattern Recognition*, 134:109064, Feb. 2023. 2, 3

[45] Shreyas Ramakrishna, Baiting Luo, Christopher Kuhn, Gabor Karsai, and Abhishek Dubey. ANTI-CARLA: An adversarial testing framework for autonomous vehicles in CARLA, July 2022. 3

[46] Sukrut Rao, David Stutz, and Bernt Schiele. Adversarial training against location-optimized adversarial patches. In Adrien Bartoli and Andrea Fusiello, editors, *Computer Vision – ECCV 2020 Workshops*, pages 429–448, Cham, 2020. Springer International Publishing. 2, 3

[47] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley. Color transfer between images. *IEEE Computer Graphics and Applications*, 21(5):34–41, 2001. 5, 19

[48] 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, pages 91–99, Cambridge, MA, USA, 2015. MIT Press. 2, 6, 22

[49] E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. Bradski. Kornia: An open source differentiable computer vision library for PyTorch. In *Winter Conference on Applications of Computer Vision*, 2020. 13

[50] 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. 2

[51] Dori Shapira, Shai Avidan, and Yacov Hel-Or. Multiple histogram matching. In *2013 IEEE International Conference*

on Image Processing, pages 2269–2273, 2013. 18

[52] 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)*, pages 1–7, 2019. 2, 3

[53] 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. 2, 3

[54] 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

[55] Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv:1608.07690 [cs, stat]*, Aug. 2016. 2

[56] Tencent Keen Security Lab. Experimental Security Research of Tesla Autopilot, 2019. 2

[57] Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. *arXiv:1704.03453 [cs, stat]*, May 2017. 2

[58] 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. 2, 3

[59] Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020. 9

[60] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2, 2019. 13

[61] 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, editors, *Computer Vision – ECCV 2020*, pages 1–17, Cham, 2020. Springer International Publishing. 2, 3

[62] Chong Xiang and Prateek Mittal. PatchGuard++: Efficient provable attack detection against adversarial patches. *arXiv:2104.12609 [cs]*, Apr. 2021. 2, 3

[63] C. Xiang, A. Valtchanov, S. Mahloujifar, and P. Mittal. ObjectSeeker: Certifiably robust object detection against patch hiding attacks via patch-agnostic masking. In *2023 2023 IEEE Symposium on Security and Privacy (SP) (SP)*, pages 1366–1384, Los Alamitos, CA, USA, May 2023. IEEE Computer Society. 3

[64] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. DINO: DETR with improved DeNoising anchor boxes for end-to-end object detection, July 2022. 2, 6, 22

[65] 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)*, pages 55–61, San Francisco, CA, USA, May 2020. IEEE. 2, 3

[66] 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, pages 1989–2004, New York, NY, USA, 2019. Association for Computing Machinery. 2, 3, 7, 8

[67] Haizhong Zheng, Ziqi Zhang, Juncheng Gu, Honglak Lee, and Atul Prakash. Efficient adversarial training with transferable adversarial examples. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1178–1187, Seattle, WA, USA, June 2020. IEEE. 7