

YOLO26 Threat Mapping

From Attack Surface to Diffusion-Prior Defence

Christian David Varela

Cybersecurity Researcher, Psypher Labs · San Francisco, California · September 2026

Abstract—Attacking a neural network with white-box access to its weights is training in reverse: gradient descent with respect to the input instead of the parameters. This paper maps the attack surface of YOLO26-nano under that attacker, quantifying the cost of full-frame suppression, its survival of 8-bit quantisation, targeted object fabrication, and cross-scene universal adversarial patches. We show that defences inside the detector fail because every statistic computed in its forward pass is a differentiable function of the input pixels, or has a smooth surrogate, which reduces evasion to one more term in the attacker's objective. External verification succeeds by reading a quantity absent from that objective: a score-matching denoiser, trained to reverse a discrete Brownian diffusion over clean photographs, measures the denoising residual of the pixels under each reported box. Against attacks built without knowledge of it, the check isolates the synthetic universal patch, passes a patch cut from a photograph, and does not separate bounded full-frame perturbations from genuine detections. Because the denoiser is itself a differentiable network, exfiltration of its weights reinstates the white-box gradient path, so its value ends when its weights leak, and it is built to be replaced.

Index Terms—adversarial examples, object detection, YOLO, adversarial patches, adaptive evaluation, second-model verification, diffusion models, denoising score matching, anomaly detection, threat modeling.

I. INTRODUCTION

What this paper asks. Three questions, in order. How cheaply can an attacker with the weights control what YOLO26-nano reports? Can the detector recognise the manipulation from its own forward pass? If not, what can, and for how long? The answers: control costs a change below one grey level per pixel; internal detection fails against a gradient-aware adversary, for a reason that is general; and an external check on whether the pixels under a box are a photograph succeeds, until the checking model's own weights are compromised.

An object detector reports the presence and position of objects in a frame. Downstream systems act on that report directly, so a suppressed detection is indistinguishable from an empty scene. This paper quantifies the adversarial budget required to dictate that report against YOLO26-nano [1].

The techniques are established: adversarial examples on vision models [2], the single-step gradient attack [3], projected gradient descent under an L-infinity bound [4], and suppression and fabrication objectives adapted from earlier detectors [5], [6], [7]. The contribution is measurement against a modern target: the pixel budget of each operational failure. Section V evaluates perturbations solved against the image they attack; Section VI evaluates universal patches, single optimised files that transfer to photographs they have never seen.

The paper has two parts. Part A, Sections II to VI, maps the attack surface: the detector, the adversary's capabilities, and the cost of each objective. Part B, Sections VII and VIII, analyses defence: why verification inside the detector fails, and a diffusion-prior network trained only on clean images that isolates synthetic content. Section IX synthesises both parts

and Section X concludes.

Related work. Adversarial patches against detectors run from the universal patch of Brown et al. [8] through the printed person patch of Thys et al. [9] to patches optimised to look photographic [10]; Section VI measures the digital form of that lineage against a current suppression-free head, and Section VIII-G meets the photographic case as a limit of the defence. Certified defences against patches exist for classifiers [11] and, for the hiding attack, for detectors [12]; adversarial training has been carried to detectors [13], [14]. Section VII states why neither is attempted here. Denoising-diffusion residuals have served for anomaly and out-of-distribution detection [15], [16]; Section VIII applies that readout per reported box, under an adversary, and prices its resolution. Query-only attacks [17] and transfer to unpublished models [18] are outside this work.

PART A — THREAT MAPPING

II. THE TARGET

What this section shows. YOLO26-nano ships as one checkpoint that runs in two modes: with classical non-maximum suppression, or without it, the configuration the model is named for.

Both are evaluated. Everything that follows depends on how the head decides: thousands of candidate positions, each with eighty independent class scores, no background class, no normalisation across classes. One detail fixes how every attack is written: the sigmoid that produces the final score has almost no gradient on empty regions, so every objective is optimised against the raw logit beneath it.

One checkpoint exposes two detectors. In ultralytics 8.4.144, `predict()` with no arguments runs the one-to-many head

followed by classical non-maximum suppression; passing `nms=False` sets the head's end-to-end flag and runs the one-to-one head with no suppression. The selection is `Detect.forward` in `nn/modules/head.py`, which reads the one-to-one branch when that flag is set, and `nn/autobackend.py`, which sets the flag from `nms` is `False`. We confirmed the split at the kernel: the default path calls `torchvision.ops.nms` once per image and the opt-in path never does. The library's default is the suppressed path; we designate it `cfgA`, the suppression-free path `cfgB`, and measure both.

The head emits three hundred candidate slots per call; on the 1,000 held-out images the most crowded photograph used 30 of them under `cfgB` and 29 under `cfgA`.

On the same 1,000 images `cfgA` reports 4.85 detections per image and leaves 2.1% of images empty; `cfgB` reports 4.52 and leaves 2.7% empty. On the 500 tuning images the means are 4.91 and 4.55: the two paths return different detection sets on the same photographs, and the difference is the one-to-many head's extra reports before suppression.

$$f_{\theta}: x \in [0, 1]^{3 \times 640 \times 640} \mapsto (z, b), \\ z \in \mathbb{R}^{8400 \times 80}, \quad b \in \mathbb{R}^{8400 \times 4} \quad (1)$$

The detector as one function. The input x is a colour picture of 640 by 640 pixels with every value scaled to lie between 0 and 1. The network with weights θ maps it to two arrays: z , one raw score (a logit) for each of the 8,400 candidate positions and each of the 80 object classes, and b , one box, four coordinates, for each candidate position. Nothing about the map is random and every operation inside it is differentiable, which is why the attacker can ask how any entry of z changes when any pixel of x changes.

$$D(x) = \{(i, c) : z_{ic}(x) > \tau\}, \\ \tau = \sigma^{-1}(0.25) = \log \frac{0.25}{0.75} \approx -1.10 \quad (2)$$

What the detector reports. A position i is reported as an object of class c when its logit for that class exceeds the reporting threshold τ . The library's default threshold is a confidence of 0.25; on the logit scale, before the sigmoid, that is minus 1.10. Every attack objective in this paper is written on z and compared with τ , never on the sigmoid confidence, because the sigmoid flattens to nearly zero slope on empty background and the gradient there is unusable.

A. Architecture relevant to the attack surface

Four properties of the head define the exposure. It is anchor-free over three strides, giving 8,400 candidate positions at 640x640. Each position carries 80 independent sigmoid scores and no background class, so at the output layer raising one class score costs nothing in any other; what couples the classes is the shared backbone beneath, which is where the

gradient acts. The box branch regresses four unbounded distances through a single distribution bin, where earlier generations used a bounded distribution. And `cfgB` fills its 300 output slots by score rank over (anchor, class) pairs, regardless of absolute confidence.

A fifth property governs construction rather than exposure. Scores leave the head through a sigmoid whose derivative has a median of 2.46e-07 on background positions, against 0.210 on positions already above threshold. A gradient that must reach the background has to be taken on the pre-activation logit, and every objective in Section V is written there.

III. ADVERSARIAL THREAT MODEL

What this section shows. The adversary is white-box: architecture, weights, and the gradient of every output with respect to the input, which for a public checkpoint costs nothing.

It writes at one of three points in the inference path, the file, the tensor, or a bounded region, and the point of entry fixes what the perturbation must survive. Its goals are availability, an empty frame, and integrity, a fabricated class, a displaced box, or a duplicated one. Confidentiality is not evaluated.

A. Adversary knowledge

Attacks assume complete knowledge: architecture, weights, and input-output gradients. The adversary runs the checkpoint the operator deploys. Section V-B measures what disturbance alone achieves without it; query-only attacks that read the detector's answers [17] are outside this work (IX-B). The defender, where one appears in Part B, sees the frame and the detector's raw output, logits and boxes, and may run further forward passes; no latency budget is imposed.

B. Adversary capability

Capability is the adversary's insertion point in the inference path, and it dictates the transformations the perturbation must survive (Table 1). All interventions are digital; nothing is printed, photographed or staged.

TABLE 1. Adversary capability, by position in the inference path.

position	what is written	must survive	measured in
file	the image file before decoding	decode, 8-bit rounding	Sec. V-F
tensor	the whole model input	nothing further	Sec. V-C
region	a bounded area of that input	8-bit rounding, when written as a file	Sec. V-H, VI

$$\|\delta\|_\infty \leq \varepsilon, \quad x + \delta \in [0, 1]^{3 \times 640 \times 640} \quad (3)$$

A full-frame perturbation. The attacker adds a change delta to every pixel. The infinity norm being at most epsilon means no single pixel value moves by more than epsilon, so a budget of 1/255 is one grey level per pixel. The second condition says the result must still be a valid picture. This is the standard L-infinity threat model of the adversarial-examples literature.

$$x \oplus_m p = (1 - m) \odot x + m \odot p, \\ m \in \{0, 1\}^{640 \times 640}, \quad \sum_u m_u = a \cdot A_{\text{photo}} \quad (4)$$

A patch. The mask m is one inside a square region and zero elsewhere, and it covers a fraction a of the photograph's own area A_{photo} , not of the padded 640 by 640 canvas: one per cent of a 640 by 480 photograph is about 55 pixels square. The operator replaces the pixels under the mask with the patch content p and leaves everything else untouched. Inside the square the attacker may write any value at all; outside it may write nothing. The universal patch of Section VI and the stealth patch are both of this form.

C. Adversary goals

Failures map onto the conventional objectives of availability and integrity (Table 2). Confidentiality is not evaluated: nothing here attempts to extract parameters or training data.

TABLE 2. Failures established against this target, by security objective.

objective	failure	section
availability	whole-image output suppression	V-C
availability	suppression from a bounded patch	V-H
integrity	injection of a chosen class at a chosen place	V-G, VI
integrity	removal of one detection with the rest preserved	VII-D
integrity	displacement of a box with label and confidence held	V-I
integrity	duplicate rows for a single object	V-J
confidentiality	not evaluated	-

D. Attack timing

All attacks act at inference. Training, the training data and the distribution of the weights are out of scope.

IV. EXPERIMENTAL SETUP

What this section shows. One checkpoint identified by its hash; one public corpus split once by seed; *blanked* and *vanished* defined before any run; a Wilson interval on every per-image rate, with the image as the unit; and a gradient path validated against finite differences, because the library detaches the suppression-free branch in training mode and an attack written the obvious way would report the model as robust.

The section closes with a ledger of every experiment: its split, its sample and its eligibility rule. With these fixed, a failed attack is a budget result, not a tooling artefact.

The target is `yo1o26n.pt`, sha256 9b09cc8bf347f0fc8a5f7657480587f25db09b34bf33b0652110fb03a8ad4fef, 2,408,932 parameters in the fused inference graph (2,572,280 before fusion, with both heads). Software: ultralytics 8.4.144, torch 2.14.0, torchvision 0.29.0, OpenCV 5.0.0, Python 3.12.13, on macOS-26.6.2-arm64-arm-64bit; perturbations are optimised on the Apple GPU and every reported number is scored on the CPU. The corpus is COCO val2017, 5,000 images, the vendor's own validation set, split once under a fixed seed into 500 for tuning, 500 for training universal patches, 1,000 held out and 3,000 sealed and unused. Unless stated, a per-image attack runs on the first 100 or 50 images of the tuning split in list order, and the number of eligible images is given with each result; Table 3 lists every experiment. Every figure and table is generated from result files carrying a provenance block.

An image is *blanked* when it returns no detection above the 0.25 threshold. A detection has *vanished* when no adversarial box of the same class overlaps the clean box by IoU 0.5, so a detection whose class changes counts as vanished. Images without clean detections are excluded from vanishing denominators. L-infinity perturbations target photograph pixels only; letterbox padding is preserved. Suppression rates carry Wilson intervals [19] over images; where several seeds are run the interval is taken at the image count on the seed mean, never on the pooled count, and the seed range is given beside it. Per-detection rates are additionally clustered on the image, since detections nest within scenes. AUCs in Part B carry frame-level bootstrap intervals. No correction for multiple comparisons is applied; the paper reports measured rates with their intervals and makes no significance claims.

The gradient path is validated before use. On one image, autograd's input gradient agrees with float64 central differences at 32 pixels to a maximum relative error of 1.3e-08 and along five random directions to 9.3e-07. The scoring path is bit-identical to the stock library; the differentiable handle's raw logits differ from the stock path by at most 4.3e-04 (relative 6.3e-06) on 16 images, with no change to any detection set. CPU and Apple-GPU forward passes agree on the detection count and classes of 100 of 100 tuning images, with matched boxes at IoU above 0.9999 and raw logits differing by at most 3.7e-03; thread count moves raw logit bits

on 30 of 50 images and no detection. Attacks run in evaluation mode. In training mode the library detaches the suppression-free branch (`detect.forward`), so a white-box attack written the natural way optimises nothing and reports the model as robust; the validation rules that out.

$$\hat{p}_{\pm} = \frac{\hat{p} + z^2/2n \pm z\sqrt{\hat{p}(1-\hat{p})/n + z^2/4n^2}}{1 + z^2/n}, \quad z = 1.96 \quad (5)$$

The interval on every rate. When k of n images are blanked, the observed rate is $\hat{p}$ equal to k over n . The Wilson score interval replaces the naive plus-or-minus with a centre pulled slightly towards one half and a width that stays sensible when $\hat{p}$ is near 0 or 1, which the naive interval does not. With z equal to 1.96 the interval covers the true rate 95 percent of the time. Every bracketed pair of numbers in this paper is this interval.

$$\text{AUC}_{\rightarrow} = \Pr[S_{\text{fab}} > S_{\text{gen}}] \quad (6)$$

Directed separation. Take one item the attack fabricated and one genuine item at random; the directed AUC is the probability that the fabricated one has the higher score S . A value of 0.5 is a coin toss, 1.0 is perfect separation, and a value below 0.5 means the score runs the wrong way, the attacked item looking more normal than the genuine one. The paper reports it as it comes and never folds it around one half, because the direction is itself a finding; where a two-sided statistic is scored (VIII-G) it is a different score, itself reported directed.

TABLE 3. Every experiment in the paper: the split it ran on, the sample, the eligibility rule and the head. Tuning = the 500-image tuning split, first n in list order; held = the 1,000-image held split; univ = the 500-image universal-patch training split. n is requested $\rightarrow$ eligible where the two differ.

experiment (section)	split	n	eligibility	cfg
sanity bound (V-A)	tuning	100 $\rightarrow$ 97	clean detections > 0	B
random noise (V-B)	tuning	100 $\rightarrow$ 97	clean detections > 0	B
budget curve, 3 seeds (V-C)	tuning	100 $\rightarrow$ 97	clean detections > 0	A, B
step calibration (V-D)	tuning	50 $\rightarrow$ 49	clean detections > 0	B
head vs operator (V-E)	tuning	100 $\rightarrow$ 97	clean detections > 0	B, B+NMS
8-bit survival (V-F)	tuning	100 $\rightarrow$ 97	clean detections > 0	B
JPEG survival (V-F)	tuning	100 $\rightarrow$ 97	clean detections > 0	B
detection statistics, Figs. 5, 9, 10 (V-C)	tuning	80	clean detections > 0	B
targeted fabrication (V-G)	tuning	100; 100; 52	target class absent from ground truth	B
per-image patch (V-H)	tuning	50 $\rightarrow$ 49	clean detections > 0	B

experiment (section)	split	n	eligibility	cfg
box mislocalisation (V-I)	tuning	50 (238 det.)	clean detections > 0	B
duplicates (V-J)	tuning	50 $\rightarrow$ 49	clean detections > 0	B (A scored)
universal patch, 3 patches (VI-C)	held	90 $\rightarrow$ 39	no person in ground truth	B (A recorded)
delivery ladder (VI-D)	univ, unused	15	no person reported clean	B
stealth patch (VI-E)	held	39	no person in ground truth	B
universal bounded delta (VI-E)	held	44; 24	no person in GT, ≥ 1 clean det.	B
generator (VI-E)	held	30	no person in ground truth	B
coverage census, quiet-output (VII)	tuning	20	≥ 1 clean det., no person in GT	B
noise-probe adaptive (VII-A)	tuning	25	as above	B
selective removal (VII-D)	tuning	30	≥ 2 clean detections	B
clean null, quiet-output (VII-D)	tuning	500	—	B
flip gate, non-adaptive (VII-A)	held	100 PGD; 300	no person in ground truth	B
flip gate, adaptive (VII-A)	tuning	40	no person in ground truth	B
transform bank (VII-A)	tuning	15	no person in ground truth	B
second detector, transfer (VII-B)	tuning	30	no person in ground truth	B
second detector, adaptive (VII-B)	held; tuning	200; 15	no person in ground truth	B
certified masking (VII-C)	tuning	40	no person in ground truth	B
denoiser census (VIII-C)	val2017 crops	200	—	—
denoiser detector test (VIII-E)	tuning	25	no person in GT, ≥ 1 clean det.	B

Reading the table. Every reported rate in the paper traces to one row. The tuning split carries every per-image attack of Section V and most of the defence measurements of Section VII, because hyperparameters were set there and the held split was reserved for the universal patch, the image-specific attacks and the confirmation runs of Section VII. The denoiser of Section VIII was trained on train2017 and has never seen any val2017 image, so for it every row is held out; the row for its detector test says tuning because the detector attacks in it were built with hyperparameters set on that split. `cfgB` is the suppression-free head; `A` means the default head with NMS.

V. ATTACK EVALUATION

What this section shows. This section prices each adversarial objective against a pixel budget, with the picture and the model in the attacker's hands.

Two reference points come first: an unbounded attack that must succeed, which validates the optimisation, and random noise that must fail, which isolates what the gradient contributes. Between them we price suppression, survival of 8-bit quantisation, targeted fabrication, patch suppression, box mislocalisation and duplicate injection. Suppression is the cheapest objective and operates at a fraction of a grey level per pixel; fabrication and mislocalisation demand larger budgets that remain practical.

Each attack is reported as its method and its measured effect. Sections V-A and V-B fix the ceiling and the floor against which the rest are read.

A. Sanity bound: the unbounded gradient attack

$$x_{k+1} = \text{clip}_{[0,1]}(x_k - \alpha \text{sign} \nabla_x L(x_k)) \quad (7)$$

The attack with no budget. Each step moves every pixel by a fixed amount alpha in the direction that lowers the attack loss L, using only the sign of the gradient, and then clips the result back into the valid pixel range. There is no projection onto a ball because there is no ball. If this attack cannot reach its goal the tools are broken; if it reaches it in a handful of steps, as it does, then any failure at a finite budget is the budget's doing and nothing else.

Method. Sign-gradient steps of 4/255 on the raw class logits, projected onto the valid pixel range only, halting once every candidate position sits half a logit below threshold. 100 images, cfgB.

Effect. 1.000 of eligible images blanked, Wilson [0.962, 1.000], in a median of seven steps. Directed at a class absent from the clean output, the same procedure produced that class on 1.000 of images. Both objectives reach their ceiling, so every failure at a finite budget below is the budget's doing. The perturbation is plainly visible, which places this configuration outside the threat model of Section III.

B. The floor: random noise, no attack

$$\begin{aligned} \delta &\sim \text{Uniform}\{-\varepsilon, +\varepsilon\}^{3 \times 640 \times 640} \quad \text{or} \\ \delta &\sim \text{Uniform}[-\varepsilon, +\varepsilon]^{3 \times 640 \times 640} \end{aligned} \quad (8)$$

The control. The first form picks a random corner of the allowed region: every pixel moves by exactly plus or minus epsilon, chosen at random. The second picks a random point inside it. Neither uses the model or the gradient. Whatever these achieve is what disturbance alone achieves, and every gradient result in this section is read against it, so that a number means attacked and not merely perturbed.

Method. Three gradient-free arms at each budget: a random corner of the L-infinity ball, uniform noise inside it, and the most damaging of twenty independent corner draws. 97

eligible images, cfgB. The worst of twenty is a random search with twenty queries, an upper envelope of what noise achieves, not a no-attack control. This is what disturbance alone achieves; query-only attacks that read the detector's answers are stronger [17] and are outside this work.

Effect. Suppression stays near zero until the perturbation is plainly visible, reaching 0.134 at 32/255. At and below 4/255 the count of removed detections is negative: noise adds false positives faster than it removes true detections, so precision degrades while recall holds. Every gradient result below is read against this floor.

TABLE 4. Random perturbation. Negative values in the last column indicate detections added.

budget	one draw	worst of 20	detections removed
1/255	0.000	0.000	-0.012
2/255	0.000	0.010	-0.024
4/255	0.021	0.031	-0.010
8/255	0.010	0.072	+0.034
16/255	0.041	0.082	+0.093
32/255	0.134	0.216	+0.354

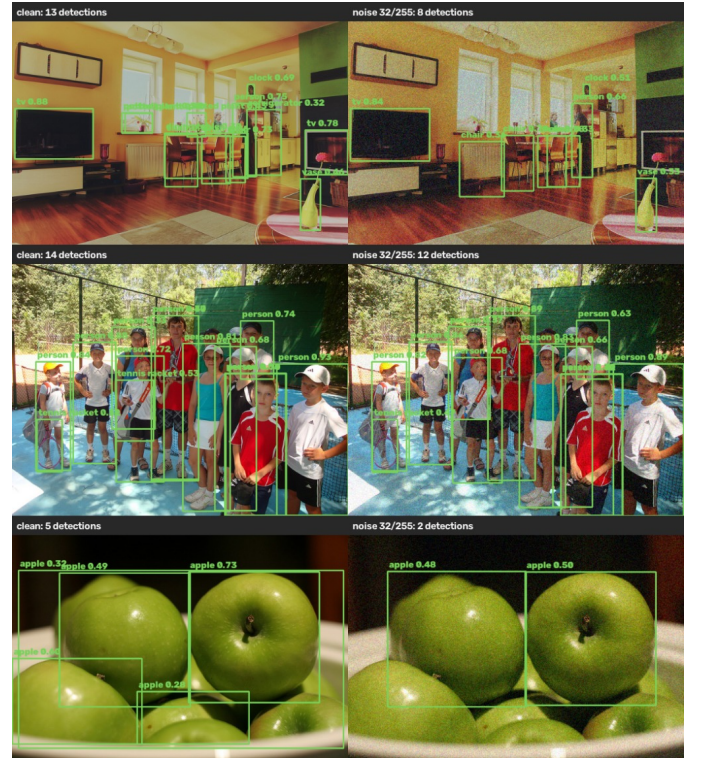


Fig. 1. Random noise at 8 and 32 of 255. The perturbation is visible; the detections persist.

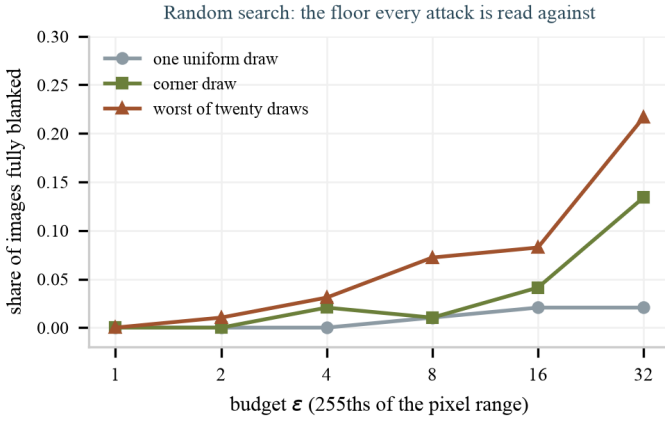


Fig. 2. Share of images fully blanked by random perturbation, against budget, for one uniform draw, a corner draw and the worst of twenty draws. All three stay below a third even at 32/255, where the perturbation is plainly visible. This is the floor against which every gradient result in Section V is read: the gradient attacks reach nine in ten at a budget thirty times smaller.

C. Object-vanishing attack (bounded PGD)

$$\delta_{k+1} = \Pi_{\|\delta\|_\infty \leq \epsilon}(\delta_k - \alpha \text{sign} \nabla_\delta L(x + \delta_k)) \quad (9)$$

Projected gradient descent, the one update rule. Starting from a random delta inside the budget, each step moves every pixel by alpha, one sixteenth of the budget here, in the direction that lowers the loss, and the projection Π clips the result so that no pixel has moved by more than epsilon in total. It runs for a fixed number of steps and stops early once the loss reaches zero. Every bounded attack in this paper is this rule with a different loss.

$$L_{\text{van}}(x) = \sum_{i=1}^{8400} \max(0, \max_c z_{ic}(x) - (\tau - \kappa)), \quad (10)$$

$\kappa = 0.5$

The vanishing loss, as run. For each candidate position i take its strongest class logit. If that logit is above the threshold minus a margin kappa of half a logit, it contributes the excess; if it is already below, it contributes nothing. The loss is zero exactly when every position's strongest score sits at least half a logit under the reporting line, that is, when the frame is empty with room to spare against rounding.

Method. Projected gradient descent under an L-infinity budget: step one sixteenth of the budget, 120 steps, random start, three seeds. The loss sums a hinge over each position's strongest class logit, driving it half a logit below threshold and releasing it there. Success is taken at the best iterate.

Effect. Table 5 and Figure 3. Two images in three lose their entire output at 0.5/255 and nine in ten at 1/255 in the suppression-free head; the default head is harder at the smallest budgets (0.282 against 0.385 at 0.25/255) and within the interval from 0.75/255 up. The three seeds differ by a

standard deviation of 0.006 to 0.027, a range of at most 0.052, below the sampling interval at every budget. At the same budget the random search of Section V-B removes nothing.

TABLE 5. Images fully blanked, three seeds, 97 images. Rates are the seed mean; intervals are Wilson with the image as the unit ($n = 97$), taken on that mean. The final column is the share of individual detections removed in cfgB.

budget	cfgB	95% CI	cfgA	95% CI	det. gone
0.25/255	0.385	[0.29, 0.48]	0.282	[0.20, 0.38]	0.650
0.50/255	0.670	[0.57, 0.76]	0.609	[0.51, 0.70]	0.891
0.75/255	0.814	[0.73, 0.88]	0.793	[0.70, 0.86]	0.950
1.00/255	0.928	[0.86, 0.96]	0.898	[0.82, 0.94]	0.976
1.50/255	0.983	[0.93, 1.00]	0.997	[0.96, 1.00]	0.996

Reading the table. Each row is one pixel budget. The columns give the share of images left with no detection, for the suppression-free head and for the default head with NMS, each with its 95% interval, and the share of individual detections removed. Down the first column: two images in three at half a level per pixel, nine in ten at one level, nearly all at one and a half. The NMS path is harder at the two smallest budgets and within the interval of the suppression-free path from 0.75/255 up.

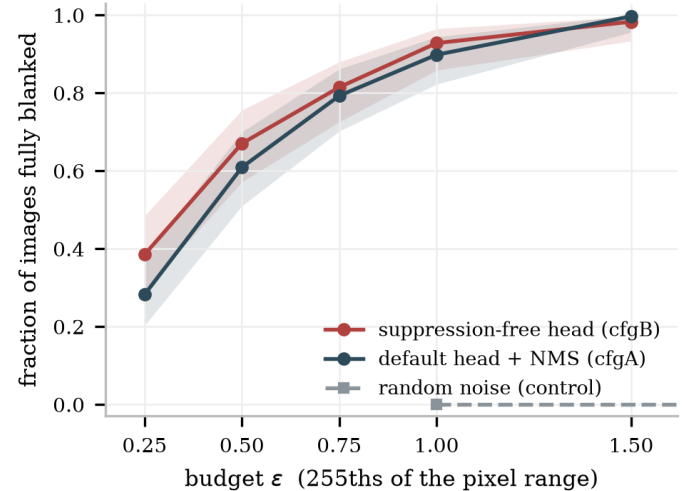


Fig. 3. Share of images left with no detections, against budget. Bands are Wilson intervals with the image as the unit ($n = 97$) on the seed mean; grey is random search at matched budgets.

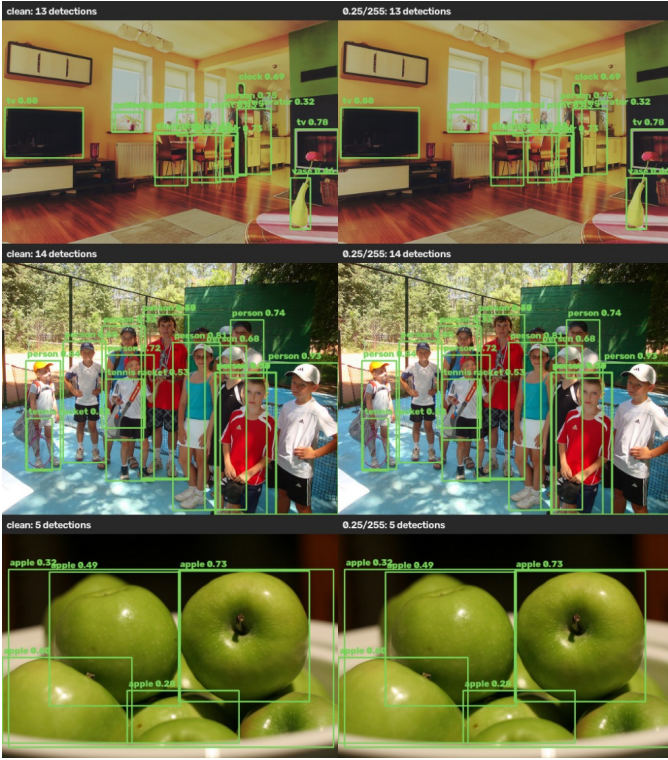


Fig. 4. One scene clean and at three budgets, 0.25, 0.50 and 1.00 of 255.

Figure 5 gives the distribution behind the rate. At 0.5/255 the mass sits at total loss with a short tail: output ends rather than thins.

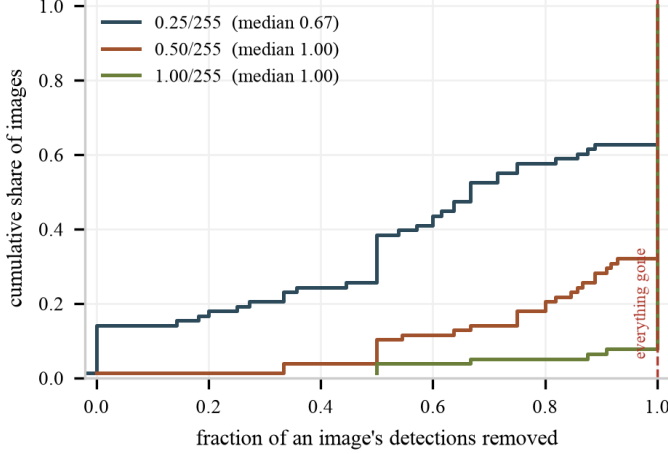


Fig. 5. Share of its output each image loses, as a cumulative distribution.



Fig. 6. Object vanishing on three scenes from the tuning split. Left: the unmodified file with everything the detector reports. Right: the same photograph after a bounded gradient attack at 1/255, scored through the same pipeline, reporting nothing. The perturbation is one grey level per pixel and is not visible at this scale, which is the availability result in visual form: the failure is not preceded by degraded boxes or falling confidences.



Fig. 7. Eight scenes at a fixed budget of 1/255, clean above and attacked below. The effect is not particular to a scene type: indoor and outdoor, crowded and sparse, large objects and small all lose their output at the same budget, which is what makes the rate in Table 5 a property of the model rather than of the images we happened to select.



Fig. 8. The perturbation at 1/255 beside the photograph it is added to, amplified so that it can be seen at all. At true scale it is one part in 255 of the pixel range. The amplified panel shows spatial structure; the noise baseline of Section V-B, at thirty times this budget, has none by construction.

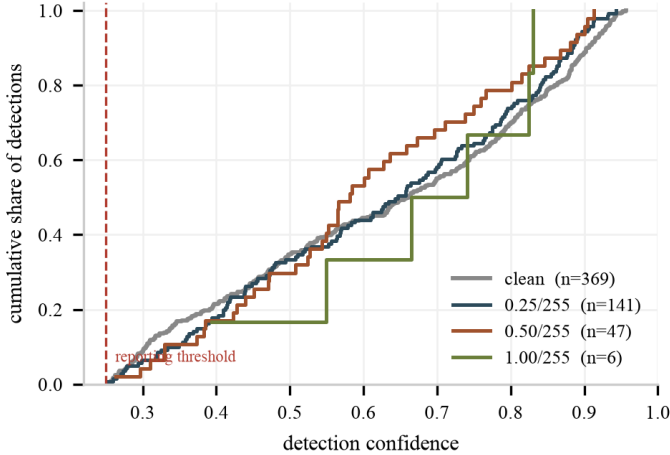


Fig. 9. Confidence distribution of the detections that survive the attack, against the clean distribution on the same 80 tuning images. At 0.5/255 the 47 surviving detections show no detectable shift from the 369 clean ones (two-sample Kolmogorov-Smirnov $D = 0.16$, permutation $p = 0.24$); at 1/255 six survive, too few to test. Detections are removed outright rather than pushed toward the reporting threshold, so the confidence values an operator observes carry no indication that an attack is in progress.

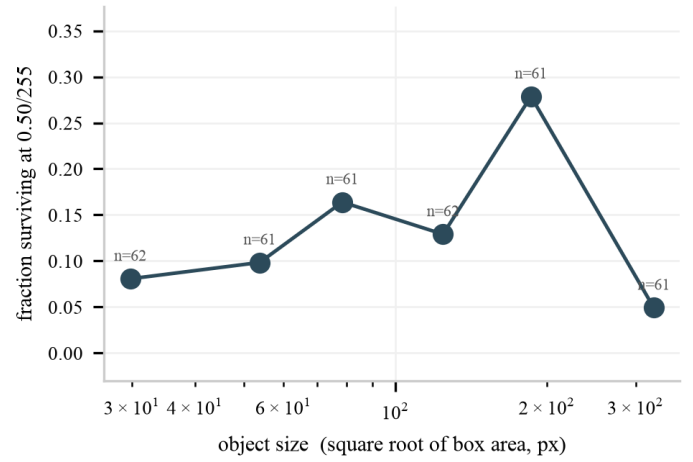


Fig. 10. Survival rate of a detection against the size of its bounding box at 0.5/255, 369 detections on 80 tuning images in six size bins of about 61 each. Survival by bin, smallest to largest, is 0.08, 0.10, 0.16, 0.13, 0.28 and 0.05: it varies across bins by more than sampling noise (permutation $p = 0.004$) but shows no trend with size, and the largest sixth survives at 0.05. Size does not protect an object, so the headline rate is not carried by small or marginal detections.

D. PGD attack calibration: step size and iteration count

Method. Step size and iteration count are selected on the tuning split before any reported rate is measured. A step larger than the budget cannot descend at small budgets, and an attack stopped before convergence understates the exposure. Both are attacker-side defects, and both would be reported as robustness of the model.

Effect. A step of one sixteenth of the budget converges where larger steps oscillate. At 0.25/255 on 49 tuning images the blanking rate at that step is 0.367 after 100 steps and 0.367 after 500, unchanged at every checkpoint between; a step of one thirty-second fluctuates between 0.306 and 0.408 with no trend. The reported attacks use one sixteenth and 120 steps, past the plateau. Convergence is judged on the images that have not yet succeeded: the rate over the last hundred steps must not change and the median loss of the unfinished images must have stopped falling, never by a mean over all images, which is dominated by finished images and plateaus whether or not the rest are still improving.

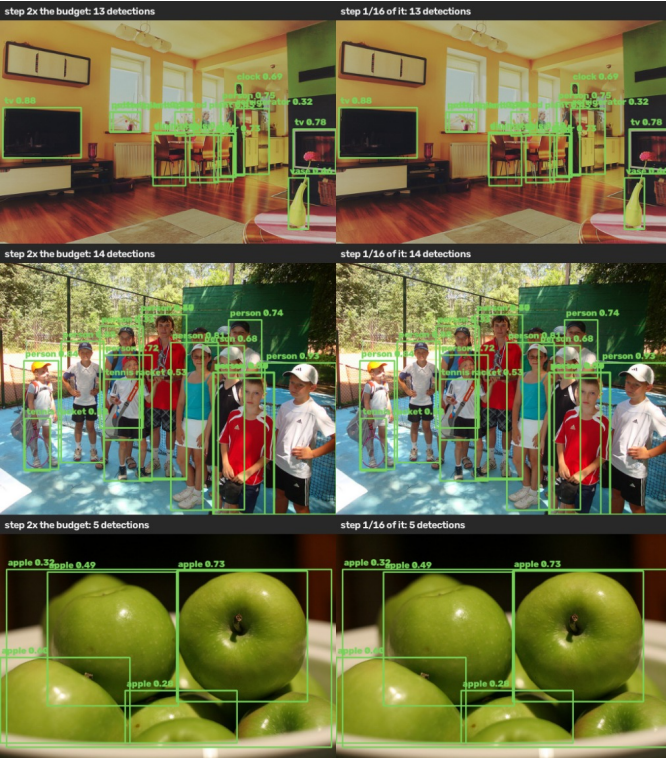


Fig. 11. One scene at 0.25/255 under a step twice the budget and under a sixteenth of it. The first cannot descend, because its step is larger than the ball it is projected into; the second removes the output. The difference is a property of the attack, not of the model.

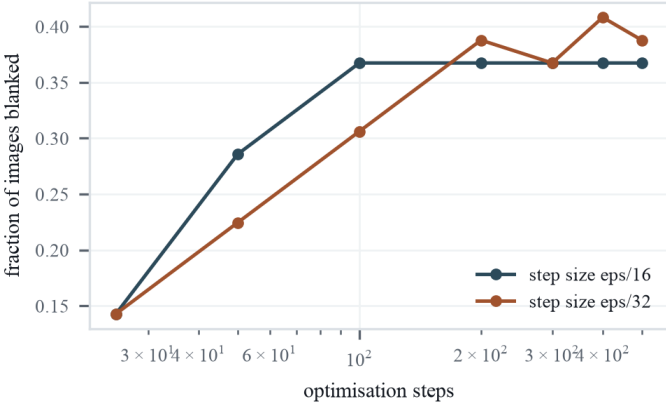


Fig. 12. Blanking rate against iteration count for the two smallest step sizes. A curve that flattens has converged; one still rising at the cap has been interrupted, and any rate read from it would be a lower bound reported as a result.

E. Ablation: detection head versus suppression operator

Method. cfgA and cfgB differ in the learned head and in the selection rule at once, so a difference between them cannot be attributed. A third configuration varies one factor: cfgB’s head output passed through classical suppression instead of top-k. Both arms read the same logits and score the identical perturbation, paired image by image.

Effect. The selection rule accounts for none of the difference: zero disagreements across 485 paired comparisons, 97 images at five budgets. Once scores fall below threshold

there is nothing for either rule to select. The exposure resides in the learned head.

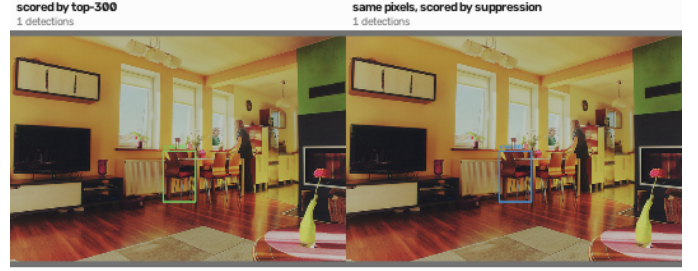


Fig. 13. One perturbation scored through both selection rules.

F. Object-vanishing attack after 8-bit quantization

$$x_q = \frac{\lfloor 255(x + \delta) \rfloor}{255},$$

(11)

success is scored on $f_\theta(x_q)$

Surviving the file. Writing the attacked picture to disk rounds every pixel to the nearest of 256 levels; the brackets denote rounding to the nearest integer. A perturbation solved in floating point may not survive that rounding. So the success of the attack is scored on the rounded picture x_q , exactly what a detector fed from a file would receive, and not on the floating-point picture the optimiser saw.

Method. Inside the process a perturbation exists in floating point; reaching a deployed detector means writing it to an image. Rounding is applied to the letterboxed 640×640 model input, whose interpolated values do not sit on the 8-bit grid, which is why a change smaller than one level can still move some pixels across a rounding boundary. At every checkpoint where the float perturbation blanks an image we round it to 8 bits and score it again, so the quantised measure is the same estimator on the same population; one seed, 97 images. A second delivery, JPEG at qualities 95, 90 and 75 applied to the same 8-bit input, is measured the same way at 1/255 and 2/255.

Effect. After rounding the attack blanks 0.082 of images at 0.5/255 (8/97, [0.042, 0.154]), 0.495 at 0.75/255, 0.845 at 1/255 [0.760, 0.904] and 0.990 at 2/255, where it matches its floating-point rate; the dip at 1.5/255 (0.711) is one seed and is unexplained. Lossless delivery as a file is therefore within reach from 0.75/255 and complete by 2/255. Delivered as JPEG, the same perturbation blanks 0.134 of images at quality 95, 0.052 at quality 90 and 0.021 at quality 75 at 1/255, and 0.186, 0.052 and 0.031 at 2/255, against a lossless rate of 0.990 (97 images, one seed, compression applied to the 8-bit model input). The full-frame suppression attack survives lossless delivery and does not survive JPEG at any tested quality; the file adversary of Table 1 holds for PNG and fails for JPEG at these budgets. The universal patch of Section VI-D, a different attack, survives JPEG down to quality 30.



Fig. 14. A float perturbation and the same perturbation written as 8-bit.

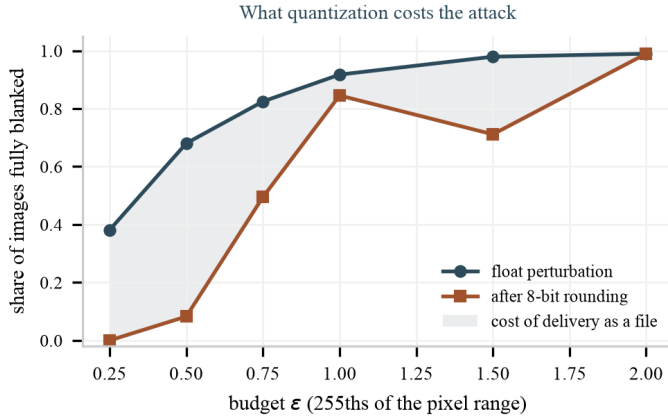


Fig. 15. Blanking rate before and after 8-bit rounding, against budget. The shaded band between the two curves is what delivery as an image file costs the attack. It is the whole attack below 0.5/255, where the perturbation is finer than the quantization grid, and closes to nothing by 2/255. This curve is what separates the file position from the tensor position of Table 1.

G. Targeted fabrication attack (object insertion)

$$L_{\text{fab}}(x) = \min_{i \in R} [\max(0, (\tau + \kappa) - z_{i, c^*}(x)) + w \cdot d(b_i(x), B)] \quad (12)$$

The insertion loss, as run. The attacker names a class c^* and a target box B . Over the candidate positions R whose boxes fall inside the target box, each position pays a hinge for how far its logit for c^* still sits below the threshold plus the margin κ of half a logit, and a penalty d equal to one minus the overlap of its box with the target, weighted by w equal to one. The minimum means the easiest candidate carries the attack. The loss is zero once one position reports c^* above threshold at the requested place. In the arm that asks for the class anywhere in the frame the box term is dropped and R is every position.

Method. Suppression removes output; fabrication inserts it. The loss drives the logit of a chosen class above threshold, in one arm anywhere in the frame and in a second at a pre-registered box, a centred rectangle covering an eighth of the canvas, counted as success only when an emitted row of that class overlaps the target by half. Three target classes were fixed before the run, one common and two rare: person, toothbrush and hair drier. Images whose ground truth contains the target class are excluded, which leaves 52 of 100 tuning images for person and all 100 for the other two. Step one sixteenth of the budget, 150 steps, one seed. The null is each class's clean emission rate on the same images, which is zero in every cell.

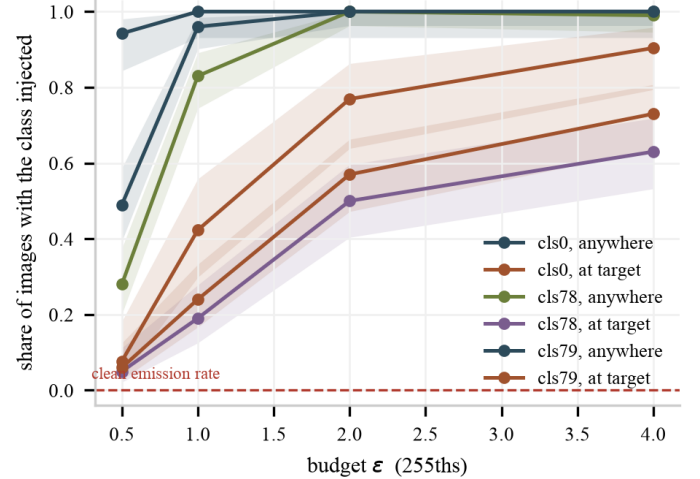


Fig. 16. Share of images on which the chosen class is emitted at the chosen location, against budget. The dashed line is the clean emission rate.



Fig. 17. Targeted object insertion on three scenes; the chosen class is toothbrush and the budget is printed on each panel. Left: the unmodified photograph. Right: the same photograph after a bounded attack that raises the chosen class inside the chosen target box, with the inserted detection drawn. The perturbation is confined to the stated L-infinity budget and is not visible at this scale; the inserted box carries a confidence above the reporting threshold, so nothing downstream distinguishes it from a real one.

Effect. For person, injection at the chosen location succeeds on 0.077 of 52 images at 0.5/255, 0.423 at 1/255, 0.769 at 2/255 and 0.904 at 4/255 [0.794, 0.958]; toothbrush and hair drier reach 0.73 and 0.63 of 100 at 4/255. Asked for the class anywhere in the frame, the same attack places a person on 0.942 of images at 0.5/255 [0.844, 0.980] and on every image from 1/255; toothbrush rises from 0.49 to 1.00 and hair drier from 0.28 to 1.00 over the same budgets. The gap between the two arms at 0.5/255 is the cost of placement, not of fabrication: half a grey level per pixel invents a person somewhere in nineteen frames of twenty. Placed fabrication costs roughly an order of magnitude more budget than suppression: the budget that blanks two images in three places a chosen object on fewer than one in ten.

H. Per-image adversarial-patch attack

$$p^*(x) = \arg \min_{p \in [0, 1]^{3 \times s \times s}} L_{\text{van}}(x \oplus_m p) \quad (13)$$

Suppression from a square. The vanishing loss is unchanged; what changes is what the attacker may write. Instead of moving every pixel by a little, it may fill one square of side s , placed by the mask m , with any content at all. The content p is solved by the same signed gradient steps, restricted to the square, for this picture only, and then measured on it.

Method. Following the adversarial-patch construction [8], [9], the perturbation is unconstrained in value but confined to a contiguous square covering a fixed fraction of the image; pixels outside it are untouched, and the measured maximum perturbation outside the patch is 0.0. Each area runs against three matched baselines at the same area and placement: mid-grey, uniform noise, and a natural crop from a detection-free region of a donor photograph. Placement is the image centre or a search over a coarse grid.

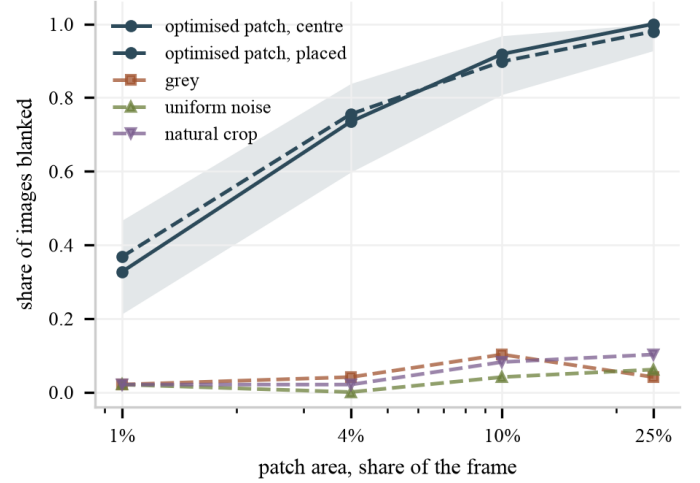


Fig. 18. Share of images blanked by a patch solved for that image, against the patch's area: one, four, ten and twenty-five per cent of the photograph, suppression-free head, 49 tuning images, one seed. Red: the optimised patch. Grey and blue: squares of the same size and place filled with flat grey or random noise, the controls, which stay near zero at every size. The curve rises from a third of images at one per cent to all of them at a quarter of the frame, and the flat controls show the effect comes from the patch's content, not from covering part of the scene.

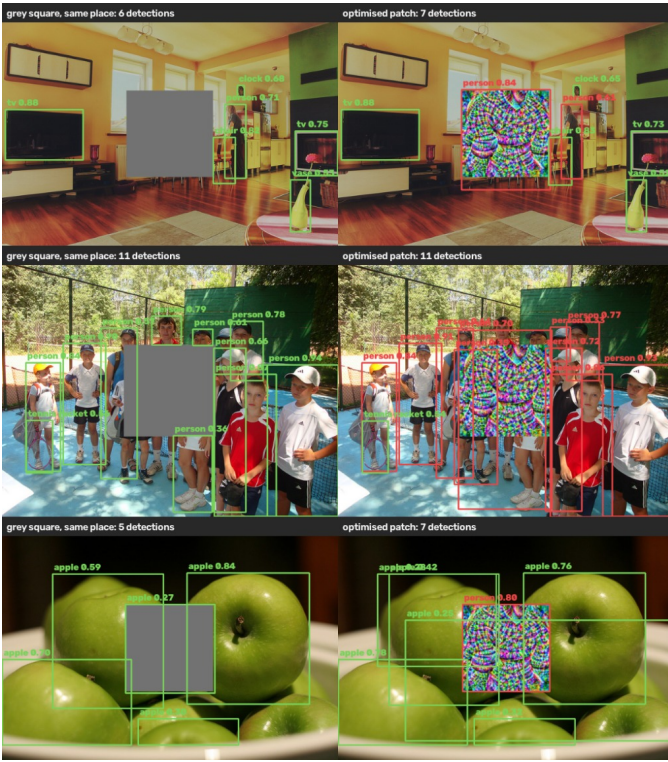


Fig. 19. The per-image adversarial patch on three scenes at a tenth of the frame. Left: a mid-grey square of the same size at the same position, which is the occlusion control and leaves most detections standing. Right: the optimised patch at that same position. The difference between the two columns is the effect attributable to the content of the square rather than to the square being there, and it is the quantity the Effect paragraph of this section reports.

Effect. A patch covering one percent of the photograph, about 54 pixels square, blanks 0.327 of images (16/49, [0.212, 0.466]). Four percent reaches 0.735, ten percent 0.918, and a quarter of the frame 1.000, all at centre placement. The grey square at the same place and size blanks 1, 2, 5 and 2 of 49 images (0.020, 0.041, 0.102, 0.041; the non-monotone step is one seed and its intervals overlap) and the noise square less, so the effect is carried by the content of the patch, not by occlusion. Placement search over a 3×3 grid, choosing the cell that most lowers the frame's strongest logit after a 30-step probe, reaches 0.367, 0.755, 0.898 and 0.980: within a few points of the centre, above it at the two smallest areas and below it at the two largest, because the probe is a proxy and the searched arm is a best-of-nine statistic. Centre placement is the reported figure.

I. Bounding-box mislocalization attack

$$\begin{aligned}
 L_{\text{box}}(x) &= w \max(0, z_i^* - z_{ic}(x)) - \frac{g(b_i(x))}{S}, \\
 z_i^* &= \sigma^{-1}(\max(0.25, 0.9 p_i)) + \kappa \\
 \text{success: } &\text{IoU}(b_i(x + \delta), b_i(x)) < 0.5 \\
 \text{and } &\sigma(z_{ic}(x + \delta)) \geq \max(0.25, 0.9 p_i)
 \end{aligned} \tag{14}$$

Moving the box while keeping the label, as run. The hinge, with weight w equal to ten, holds the class logit of detection i above a target z -star set from that detection's own clean confidence p_i : the logit of nine tenths of p_i , never below the reporting threshold, plus a margin κ of 0.2. The geometric term g depends on the objective: for displacement it is the distance the box's centre has moved, for inflation the shorter side of the box, for collapse minus the sum of its sides, all in pixels and divided by a scale S of 100 pixels. Lowering the loss pushes that geometry while the hinge keeps the detection reported with its label and confidence. The attack has succeeded when the reported box overlaps the true one by less than half and the confidence has stayed above nine tenths of its clean value.

Method. Suppression and fabrication change whether an object is reported; this attack changes where. The loss displaces, inflates or collapses the regressed box while holding the class and requiring the adversarial confidence to stay above both the reporting threshold and nine tenths of its clean value, so the result is a moved detection, not a removed one followed by a new one. No random start; the step decays on a cosine from one sixteenth of the budget over 150 steps, because a fixed step bounces off the confidence constraint. Success is counted per detection when the best same-class candidate falls below IoU 0.5 against the clean box; a detection with no candidate is recorded as vanished and reported separately. 238 detections on 50 tuning images, one seed.

Effect. At 4/255 the displacement objective moves 0.290 of detections past the IoU threshold with label and confidence held [0.236, 0.351], at a median centre displacement of 16.7 pixels (mean 27.8) and a median matched overlap of 0.60; inflation reaches 0.307 [0.252, 0.368] and collapse 0.471 [0.408, 0.534]. With the projection removed and a realised change of up to 71/255, the three reach 0.584, 0.487 and 0.756. Uniform noise at 4/255 reaches 0.050 with a median IoU of 0.993 among matched detections, so displacement is an adversarial effect, not degradation. Displacement as a fraction of box size was not recorded; the size stratification of Fig. 10 is not available for this attack.

TABLE 6. Detections displaced past IoU 0.5 with class and confidence held, by objective.

budget	displace	inflate	collapse	noise
1.00/255	0.042	0.038	0.063	0.013
2.00/255	0.088	0.160	0.223	0.025
4.00/255	0.290	0.307	0.471	0.050
8.00/255	0.437	0.466	0.592	0.118
unbounded	0.584	0.487	0.756	0.000

Reading the table. Each row is one objective at one budget: move the box (displace), enlarge it (inflate) or shrink it (collapse), with label and confidence held above threshold. The columns give the share of detections whose box ends up overlapping its object by less than half, with the 95% interval, beside uniform noise at the same budget. Displacement is the cheapest and the one an operator cannot see: the box is drawn, labelled and confident, and it is on the wrong thing.

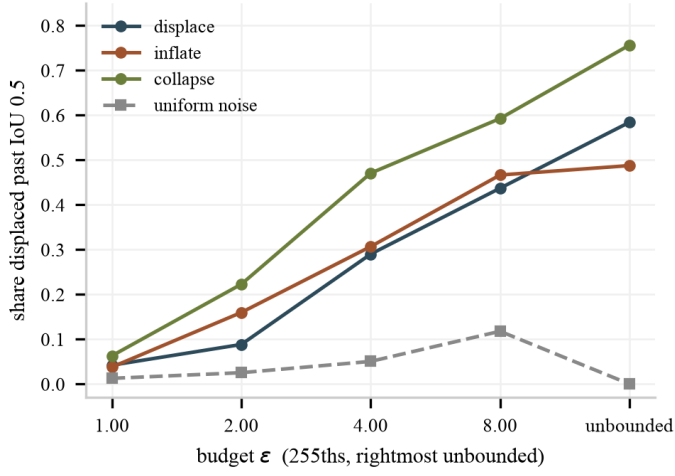


Fig. 20. Share of detections whose box slides off its object (overlap with the true box below one half) while the label and the confidence stay above threshold, against the pixel budget, suppression-free head, 50 images and 238 detections. Three objectives: move the box, inflate it, collapse it. Grey: uniform noise at the same budget, the control, near zero throughout. At 4/255 the displacement objective moves 0.290 of detections and the reported confidence does not fall, so nothing downstream notices.

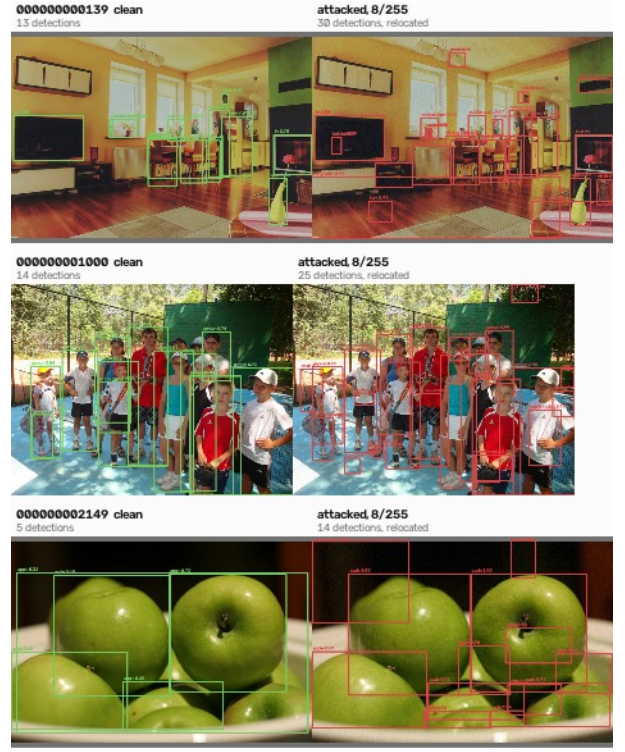


Fig. 21. Three scenes at 8/255. Boxes move from their clean positions while the class labels and the confidences are held, so the detector reports the same objects in the wrong places.

Degenerate and oversized boxes are largely out of reach. Rows with non-positive width or height appear at 0.002 to 0.007 under the collapse objective, and rows whose pre-clip area exceeds four times the canvas at 0.003 to 0.008 under inflation; both remain near their clean rate of zero.

J. Duplicate-detection injection attack

$$\begin{aligned}
 L_{\text{same}}(x) &= \sum_{c \in C_4} \max(0, (\tau + \kappa) - z_{a^*c}(x)) \\
 L_{\text{near}}(x) &= \max(0, (\tau + \kappa) - z_{a^*c}(x)) + \max(0, (\tau + \kappa) - z_{nc}(x)) \\
 &\quad \text{with } \text{IoU}(b_n, b_{a^*}) > 0.7
 \end{aligned} \tag{15}$$

Two ways to report one object twice. The position a-star is the one that already reports the object with class c . The first loss raises four class logits at that same position above the threshold plus the margin; because the suppression-free head spends its output rows on (position, class) pairs and copies one box to every pair it selects, the result is several rows with bit-identical coordinates and different labels. The second loss picks a neighbouring position n whose box overlaps a-star's by more than 0.7 and pays a hinge for each of the two logits still below the line; when both clear it the head, which never merges overlapping reports, emits the same object twice under the same label.

Method. cfgB spends its output budget on (anchor, class) pairs and copies one box to every selected pair, so one position can occupy several rows. Both forms are attacked with the losses above, step one sixteenth of the budget, 300 steps, one seed, on 49 eligible tuning images; the second form is also run with a box-tolerance term that holds the two boxes in place. Each image is compared against its own clean duplicate count, which is not zero. cfgA is scored as a structural control. Daedalus [7] attacked the suppression step of earlier detectors; here there is no suppression step to attack, and the duplicates are a property of the head's raw output.

Effect. On clean images cfgB emits 14 overlapping same-class pairs across 12 of 49 images and 3 bit-identical pairs; cfgA emits neither, since its path keeps one label per position. Under attack at 1/255 the bit-identical form increases on 0.551 of images (27/49, Wilson [0.413, 0.681]); the overlapping same-class form on 0.224 (11/49, [0.130, 0.359]), and on 0.490 (24/49, [0.356, 0.625]) with the box-tolerance term; the matched-budget noise null is 0/49 for both forms, [0.000, 0.073]. At 4/255 the three reach 1.000, 0.367 and 0.776. A consumer counting objects from the raw suppression-free output starts from a non-zero duplicate rate and can be driven well above it.

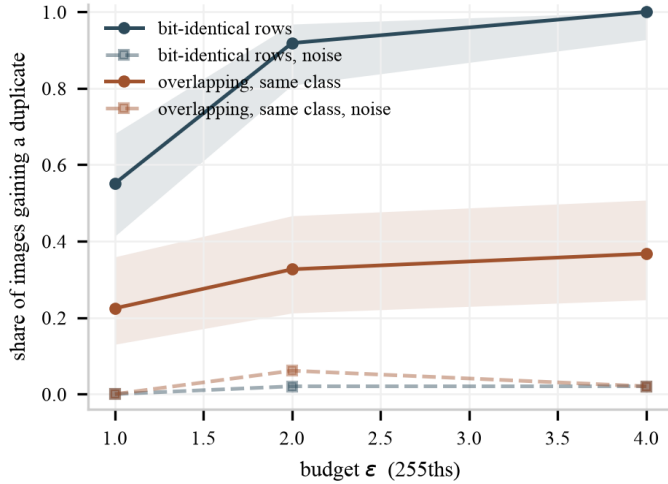


Fig. 22. Share of images gaining a duplicate row over their own clean count, against budget, with matched-budget noise as the control; 49 tuning images, one seed.



Fig. 23. Duplicate injection. Left: the clean image, one detection for the object. Right: the attacked image, two overlapping detections of the same class above threshold for the same object. Both carry confidences above the reporting threshold, so a consumer that counts objects from the raw output reports one too many with nothing to indicate a fault.

Two forms, and who is exposed. The bit-identical form exists because the output budget is spent on (anchor, class) pairs and one box is copied to every pair that selects it; it disappears when the head's class-agnostic flag makes the budget select positions instead, which on clean images costs 3 output rows, all themselves duplicates. The overlapping same-class form is a separate mechanism, unaffected by that flag. Neither form survives a consumer that applies its own non-maximum suppression at IoU 0.7, which removes both by construction, and cfgA cannot express either. The failure is a property of the raw suppression-free output, and it reaches a consumer only if that consumer reads it directly.

VI. UNIVERSAL AND IMAGE-SPECIFIC PATCH ATTACKS

What this section shows. Every attack in Section V required the target image and the model before delivery. This section removes that requirement.

A universal patch is optimised once over many scenes and fixed; pasted into photographs it has never seen, it makes the detector report a person who is not there. The patch is then hardened against what a delivery pipeline does to an image, resizing, compression, colour changes, and what survives is measured. Two further threats follow: an image-specific perturbation that stays within a bounded change of the host pixels and needs the model at the moment of attack, and a

patch cut from a real photograph that carries a small optimised change. The section closes with the systems in which an adversary can place a rectangle of pixels in front of a detector, and whether anyone would look.

Every attack in Section V is solved against the photograph it attacks. This section drops that requirement.

A. Threat model and adversary capability

The patch of Section V-H is disposable: solved for one photograph, useless on any other. A universal adversarial patch [8] is solved once across many scenes and then fixed. Its holder needs no model access at attack time and no knowledge of the target image, only the ability to place a rectangle of pixels in a frame. Every experiment in this section runs the suppression-free head, cfgB; the universal-patch scoring also records the default head, cfgA, and the two agree on every headline rate.

The patch is a file and is measured as one. Optimisation runs through a differentiable copy of the library preprocessing, which the tool’s own self-check holds within one 8-bit level of the library’s resize. Scoring writes the attacked photograph to disk, reads it back and runs the stock pipeline. An 8-bit rounding step in the forward pass keeps the optimised quantity representable as a PNG.

B. Universal adversarial-patch attack: construction

$$\begin{aligned} p^* &= \arg \min_p \mathbb{E}_{x \sim D} [L_{\text{ufab}}(x \oplus_m p)] \\ L_{\text{ufab}} &= \max(0, (\tau + \kappa) - \max_{i \in m} z_{i, c^*}) \end{aligned} \quad (16)$$

One patch for every scene. The patch p is solved by gradient descent on the average, over a training set D of scenes, of a fabrication hinge: the strongest logit for the target class c^* among the positions under the patch must clear the threshold plus the margin. The expectation over scenes is estimated by mini-batches of pasted images. Once solved the patch is a fixed file; at attack time it is only pasted, with no access to the model or to the new scene.

Objective. Fabrication lifts the highest logit of the target class at any anchor above the reporting threshold by a margin; vanishing drives every anchor’s best class logit below it. Both are written on raw logits, for the reason given in Section II-A.

Optimisation. The target class is person throughout. Each of 600 steps draws eight photographs from the 64, samples a placement and a scale between 6 and 16 per cent of the frame for each, and accumulates one sign vote per photograph rather than a sum of gradients, so no single crowded scene decides the step. Votes are combined with momentum 0.9 and applied as a sign step decayed on a cosine from 8/255.

Transformations. The hardened patch is optimised under expectation over transformation [20]: in-plane rotation to ± 20 degrees, brightness ± 0.35 , contrast ± 0.30 , gamma ± 0.35 ,

saturation down to 0.70, Gaussian blur to sigma 1.2, sensor noise of standard deviation 0.035, and a resize round trip down to a factor 0.55, each drawn uniformly within its range at every step. Every JPEG quality, grayscale conversion, rotation beyond 20 degrees, blur at sigma 3.0 and resize to half are outside that distribution and appear only at evaluation.

Splits and seeds. Training uses 64 photographs of the universal-training split; evaluation uses the sealed held-out split, untouched by every earlier experiment. Three patches are trained from different seeds, and each headline figure is their mean with the range beside it, because the patch, not the photograph, is the unit of independence.

Eligibility and scoring. A fabrication counts only in a photograph whose COCO ground truth contains no instance of the target class, so a patch that causes a genuinely present but missed object to be reported cannot score. Of 90 held-out photographs, 56.7% carry a person in ground truth and are excluded, leaving 39. The detector reports a person in 54.4% of the 90, and misses 2 that ground truth records. Those are the images the rule removes.

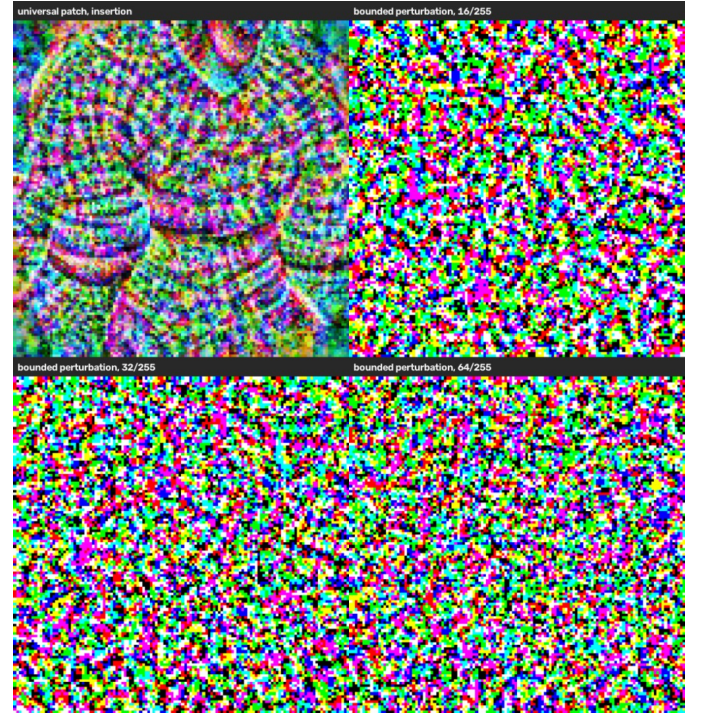


Fig. 24. The trained patch files at four times their stored size. Left to right: the universal fabrication patch and three universal bounded deltas at 16, 32 and 64 of 255, all 128 pixels square and about 49 kB. The deltas are differences added to the host pixels, shown here with zero mapped to mid-grey; solved once for every scene, none of the three fabricates a person on any of 44 held-out photographs (Section VI-E), which is the result that makes the per-photograph step of the generator necessary.

C. Universal patch attack: cross-scene transfer

Result. Across three independently trained patches on the sealed split, a patch covering four per cent of the photograph makes the detector report a person in 0.897 of scenes that contain none (range 0.872–0.923 over the three patches). Six

per cent reaches 0.974 and ten per cent 0.974. Two per cent reaches 0.316. A grey square and a uniform-noise square of the same size at the same position reach 0.000 and 0.000.

Whether the adversary also controls *where* the invented object appears is a stricter question, and the rate is lower: the fabricated box covers at least half the patch rectangle in 0.641 of cases at six per cent area. That criterion is satisfied by a large box as well as by a tight one; the overlap distribution was not recorded. The attack reliably adds an object and places it over the patch about two times in three.

The spread between the three patches is small, 0.026 standard deviation at four per cent area, so the rate is a property of the method, not of one fortunate training run.

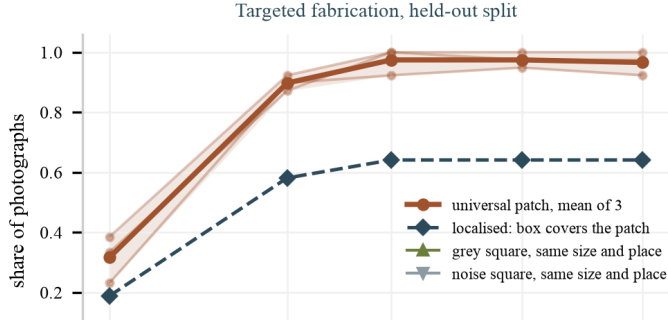


Fig. 25. Fabrication rate against patch area on the sealed held-out split, 39 photographs. The heavy line is the mean of three independently trained patches, the band their range, the thin lines each patch. The dashed line requires the inserted box to also cover the patch. The two flat lines are grey and uniform-noise squares of the same size and position, which is the rate attributable to occlusion alone. The small spread between patches is what makes this a property of the method rather than of one training run; Table 7 gives the numbers.

TABLE 7. Fabrication of a person on the sealed held-out split, by patch area. Every figure is the mean of three independently trained patches, with their range. Grey and noise are squares of the same size at the same position, and are the rate attributable to occlusion alone.

area	reported	range	on the patch	grey	noise
2%	0.316	0.231–0.385	0.188	0.000	0.000
4%	0.897	0.872–0.923	0.581	0.000	0.000
6%	0.974	0.923–1.000	0.641	0.000	0.000
10%	0.974	0.949–1.000	0.641	0.000	0.000
16%	0.966	0.923–1.000	0.641	0.026	0.026

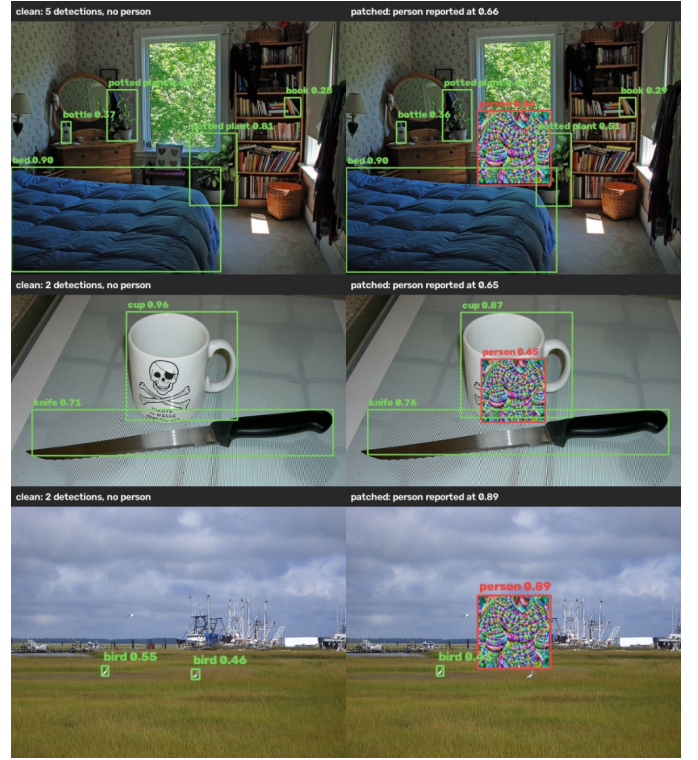


Fig. 26. The universal adversarial patch on three held-out photographs whose ground truth contains no person, at six per cent of the frame. Left: the detector on the original file, with every class it reports. Right: the same file with one 128-pixel PNG pasted in, and the person it now reports drawn in red. The same file is used in all three scenes and was trained on none of them, which is the property being demonstrated: the attacker needs no access to the target image.

D. EOT-hardened patch attack and pipeline robustness

$$p_{\text{EOT}}^* = \arg \min_p \mathbb{E}_{x \sim \mathcal{D}} \mathbb{E}_{T \sim \tau} [L_{\text{fab}}(T(x \oplus_m p))] \quad (17)$$

Expectation over transformation. The same objective, but every pasted image is first passed through a random transformation T drawn from a family: rotation, rescaling, brightness and contrast changes, blur, sensor noise, JPEG compression. Averaging the loss over that family produces a patch that still works after a delivery pipeline has altered it, at the cost of some strength on the untransformed image.

Method. A patch that works on a clean paste has been tested against nothing. Each patch is put through sixteen operations of the kind an ingest pipeline performs, plus the plain paste, every frame written to disk and read back. The rotations are applied to the patch before it is pasted, the other operations to the whole frame after. Two patches are compared, one trained on clean pastes and one trained through the transformation distribution, identical otherwise, one patch per arm, at ten per cent of the frame on 15 photographs of the universal-patch training split that took no part in training (no person reported clean). A third column records what each operation does to

unpatched photographs, because a drop in the attack rate has two possible causes and that column separates them.

Result. The plain patch survives every JPEG quality down to 30, a halving of resolution, brightness and gamma changes, and sensor noise, at 1.000 of 15 in each case [0.796, 1.000]. It fails on grayscale conversion (0.600), heavy blur and rotation past 10 degrees. Training through transformations recovers most of that: blur at sigma 3.0 rises from 0.200 to 0.933, grayscale from 0.600 to 0.733 and rotation at 25 degrees from 0.133 to 0.533; rotation at 45 degrees defeats both. The training distribution reached blur sigma 1.2 and rotation 20 degrees; sigma 3.0 and 25 degrees lie outside it, so the gain there is extrapolation of the trained family, not memorisation of the test operation. With $n = 15$ the interval on any single row is wide; the pattern across rows, not one cell, is the result.

Validity column. A drop in the attack rate under some operation means either that the operation destroyed the attack or that it destroyed the detector. The third column applies the same operation to unpatched images and reports how many genuine detections survive, matched by class at IoU 0.5, which separates the two. Without it a row such as blur at sigma 3.0, where the attack falls to 0.200 for the plain patch, would be uninterpretable: that operation also removes 30 per cent of the detector’s genuine output on images with no patch in them. For the rotation rows the operation touches only the patch, so the column reads 1.000 by construction.

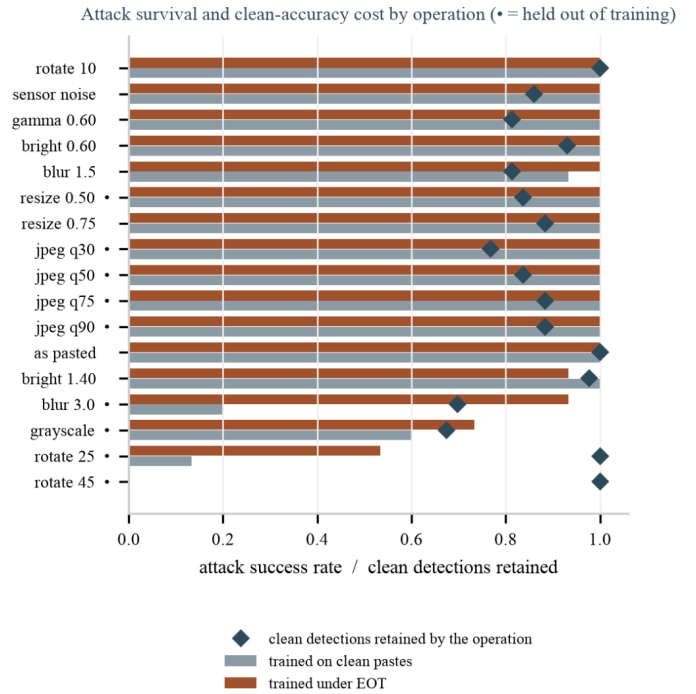


Fig. 27. Each row is one operation, ordered by how much it hurts the hardened patch; 15 photographs per row, one patch per arm. Bars give the share of photographs where the attack still works: pale for a patch trained on clean pastes, dark for one trained through the transformation distribution. Diamonds are a different quantity on the same axis — the share of genuine detections that survive that operation on unpatched photographs, the validity check that says whether a low bar means the operation destroyed the attack or destroyed the detector. Rotations are applied to the patch before pasting, so their diamonds sit at 1.0 by construction. A dot after the label marks an operation outside the hardened patch’s training distribution.

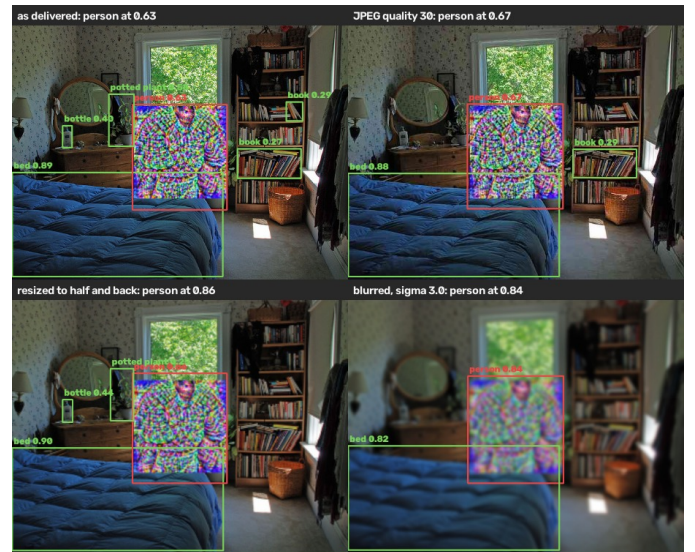


Fig. 28. One photograph carrying the hardened patch, put through four operations of an ingest pipeline and scored after each. The inserted person survives delivery as a quality-30 JPEG, a halving of resolution and a Gaussian blur at sigma 3.0, the last of which lies outside the training distribution. The confidence moves; the detection does not disappear.

E. Image-specific perturbation attack and the photographic patch

$$\begin{aligned}
\delta_0 &= \delta_u, \\
\delta_{k+1} &= \Pi_C(\delta_k - \alpha_k \text{sign} \nabla_{\delta} L_{\text{fab}}(x + m \odot f \odot \delta_k)) \\
C &= \{\|\delta\|_{\infty} \leq \varepsilon\} \cap \{\text{supp}(\delta) \subseteq R\}, \\
\delta_u &= \arg \min_{\|\delta\|_{\infty} \leq \varepsilon} \mathbb{E}_{x \sim \mathcal{D}}[L_{\text{ufab}}(x + m \odot f \odot \delta)]
\end{aligned} \tag{18}$$

A change made for one picture, as run. There is no generator network. The starting point δ_u is a universal bounded delta: one 128-pixel file solved once over the universal-patch training split with the fabrication hinge of equation 16, under the same bound ε , added to the host pixels through the rectangle mask m and a feather f that tapers it to zero at the rectangle edge. For a given picture the tool then runs 150 signed-gradient steps on the fabrication loss against that picture, with a step α decaying on a cosine from $2/255$, and projects onto C : no pixel moves by more than ε and the change is confined to a region R covering a tenth of the frame. The result is bounded and is solved for this picture only.

Construction. The patch of Section VI-C supplies its own content, which is foreign to the scene. This attack keeps the host pixels and adds an L-infinity bounded perturbation to them, feathered over the outer 18 per cent of the rectangle so no seam appears. The tool (`patchgen forge`, the author's own) first scores a 5×5 grid of placements and picks either the busiest local texture, where a bounded change is hardest to see, or the strongest detector response; it then refines the universal delta against the photograph in front of it inside the same bound and writes the file, quantised to 8 bits in the loop. The bound never changes and no foreign content is introduced: the change is a bounded delta on the host pixels.

The universal bounded delta alone. Solved once for every scene, the delta that seeds the tool fabricates nothing. On 44 held-out photographs with no person in ground truth and at least one clean detection, the universal delta invents a person in 0.000 of them at $16/255$, 0.000 at $32/255$ and 0.000 at $64/255$ (0/44, [0.000, 0.080] each); only at $128/255$, half the pixel range, does it reach 0.136. On a second sample of 24 at $32/255$, the universal delta scores 0.000 while the same bound solved per photograph scores 1.000 and the universal patch that replaces the rectangle 1.000. A bounded change and universality are not obtained together on this detector [21]: the obstacle is the requirement that one delta serve every scene, not the bound. That is why the per-photograph step below exists.

Perceptibility metrics. Two, both computed on the delivered file: the structural similarity of the changed region against the same region before, and the seam ratio, the mean gradient magnitude on the rectangle boundary divided by the mean inside it. A patch that replaces the rectangle scores 0.28 and 2.91; the bounded perturbation scores 0.85 and 0.81.

TABLE 8. The tool measured by running its command line once per photograph on 30 held-out photographs with no person in ground truth, so what is reported is the tool rather than a reimplement of it. The first two rows differ only in how the placement is chosen, and their agreement is what licenses choosing placement for concealment at no cost. The third row is the same universal delta applied without the per-photograph refinement, and is the control that shows the refinement, not the delta, carries the attack. The last two columns state what the adversary spends and how much the delivered file actually changed.

placement	refine	inserted	95% CI	s / photo	region SSIM
hide	150	0.833	[0.664, 0.927]	6	0.849
attack	150	0.833	[0.664, 0.927]	6	0.849
hide	0	0.000	[0.000, 0.114]	3	0.866

Made operational, the tool is a per-photograph generator of attacked files: given a photograph, search placements, refine the universal delta against that photograph inside the same bound, and write the file. The bound does not change and no foreign content is introduced.

Result. On 30 held-out photographs whose ground truth contains no person, at a bound of $16/255$ over a tenth of the frame, the tool invents one in 0.833 of them, interval 0.664 to 0.927, in a median of 6 seconds per photograph. The largest pixel change is 16 of 255, the structural similarity of the region that changed is 0.849, and the seam ratio is 0.81.

The per-photograph step is the whole attack. The same delta at the same placement, applied without it, invents in 0.000 of the same photographs (0/30, [0.000, 0.114]). Choosing the placement for concealment rather than for attack strength gives 0.833 against 0.833, 25 of 30 in both arms; at this n a difference of up to 0.15 would go undetected, so the claim is that no cost was measured, not that there is none.

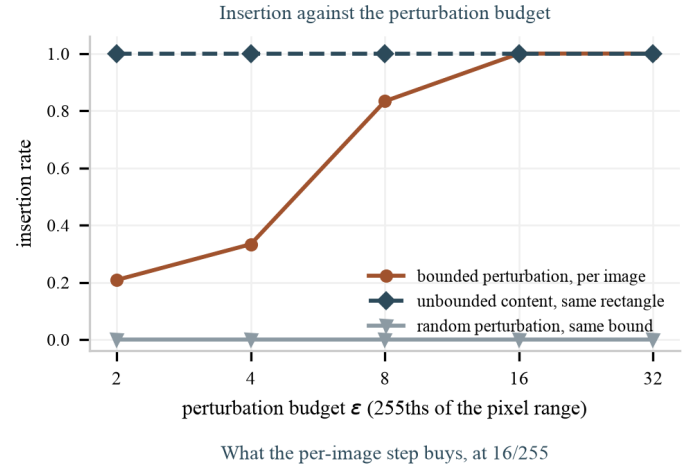


Fig. 29. Insertion rate against the perturbation budget, at a tenth of the frame, for a perturbation optimised for each image (24 held-out photographs per point). The dashed line is unbounded content in the same rectangle and the flat line is a random perturbation at the same budget, which is the control. The attack reaches ceiling by $16/255$ and is already at five in six by $8/255$, roughly three per cent of the pixel range. Table 8 gives the tool's rates with and without the per-image step.



Fig. 30. Image-specific perturbation on two held-out photographs containing no person. For each photograph the upper row gives the original and the attacked file, with the inserted person in red and the largest pixel change stated; the lower row outlines the perturbed region, because it is otherwise not locatable, and shows the perturbation alone amplified eight times. The perturbation adds to the photograph’s own pixels: the changed region keeps a structural similarity of 0.85 to the original and the whole file 0.98 (PSNR 38 dB), while the detector’s output changes.

Capability. The tool needs the target photograph and gradient access at the moment of attack, a strictly larger capability than pasting a fixed file. The two are separate threats: a reusable file whose content is foreign to the scene, and a bounded per-photograph change that is not reusable. Nothing measured here delivers both at once.

The photographic (stealth) patch. A third construction takes a 128×128 crop of a real photograph containing no person as its anchor and solves, with the universal-patch optimiser under an L-infinity bound from that anchor, a patch that stays a photograph. It is pasted at ten per cent of the frame at the centre, like the universal patch, and deployed by the same $1.58 \times$ bilinear upsample from the 128-pixel file; it is not optimised through the upsample. On the 39 held-out photographs of Section VI-C the anchor pasted unmodified fabricates in 0.000 (0/39, [0.000, 0.090]); with an 8/255 delta it fabricates in 0.974 (38/39, [0.868, 0.995]) and places the box on the patch in 0.974, with a structural similarity to the anchor of 0.81 and a total variation of 4.1 against 34.6 for the universal patch (mean absolute neighbour difference per pixel per channel, 0–255 units). At 16/255 it reaches 1.000. This is the digital form of the patches built to look photographic [10], and it is the family Section VIII-G meets. It was built on 2026-09-13, before any defence in Part B existed.

F. Delivery paths and adversary capability

This subsection is an argument about systems, not a measurement. A patch requires one capability: placing a rectangle of pixels into a frame that reaches the detector. The list below sets out systems where that holds and the access each needs. Two properties decide how much the patch’s appearance matters: whether a person ever looks at the image, and whether the adversary supplies the whole frame or only part of it.

- **Upload moderation or auto-tagging** — needs an account that can upload, anonymous on many services; a human rarely looks.
- **Automated inspection or counting from a fixed camera** — needs one chance to place an object or overlay in the field of view; seen only if the feed is watched.
- **Shared compositing asset** (watermark, logo, template) — needs write access to one asset in a repository, marketplace or CDN; seen on every image it is composited into.
- **Auto-labelling or dataset bootstrapping** — needs the ability to contribute images to a scraped or crowd-collected corpus; labels are generated, not reviewed.
- **Self-service claim or verification upload** — needs an authenticated customer account; seen sometimes, on dispute.
- **Camera or recorder burned-in overlay** — needs configuration access to the device; seen if anyone watches the feed.
- **Screen capture by an agent or assistant** — needs control of any rendered region on the victim’s screen; seen, but as ordinary page content.

The physical case, a patch printed and photographed, is outside this work. Nothing was printed, and no measurement here speaks to print reproduction, viewing angle, illumination or camera processing.

PART B — DEFENCE

VII. WHY NO DEFENCE INSIDE THE DETECTOR HOLDS

What this section shows. Nothing built from the detector’s own computation holds against an adversary who knows about it, and the reason is the threat model itself: the attacker has the model, hence the gradient of any quantity the detector produces with respect to every pixel.

Whatever statistic a defence watches, the attacker holds it steady while carrying out the attack. Stacking checks does not help, because all read the same picture; a second model helps only until transferability catches up. The section lists the defences built, what each hoped to see, and why each fell, and it explains the one exception: a check that reads the very number the attack must change, which cannot be satisfied without abandoning the attack.

The threat model of Section III is white-box: the attacker holds the weights, and that single fact decides this section. Training a network is gradient descent on a loss with respect to the parameters; attacking it is the same descent with respect to the input [2], [3], [4]. If it can be trained, it can be attacked, provided the statistic is differentiable or has a smooth surrogate, which is the condition every defence below meets. Every defence was built and then attacked by us with its full specification in the attacker's hands, the evaluation the field requires [22], [23]. Every experiment in this section runs cfgB; n and split are given with each number and in Table 3.

$$\text{training: } \theta_{k+1} = \theta_k - \eta \nabla_{\theta} J(\theta_k; x, y) \quad (19)$$

Training. J is the loss of the network on a picture x with its correct labels y. Training moves the weights theta a small step eta against the gradient of J with respect to the weights, and repeats. That gradient comes from one backward pass through the network.

$$\begin{aligned} \text{attacking:} \\ x_{k+1} = \Pi(x_k + \alpha \text{sign} \nabla_x J(\theta; x_k, y)) \end{aligned} \quad (20)$$

Attacking. The same J, the same backward pass, but the gradient is taken with respect to the picture instead of the weights, and the picture is moved up the loss instead of the weights down it, under a projection that keeps the change within budget. This is the whole of white-box attacking. If a model can be trained, it can be attacked, because both need exactly one thing: this gradient.

$$\nabla_x L = \left(\frac{\partial z}{\partial x} \right)^T \nabla_z L, \quad \frac{\partial z}{\partial x} \in \mathbb{R}^{(8400 \cdot 80) \times (3 \cdot 640^2)} \quad (21)$$

Where the attacker's gradient comes from. Any attack loss L in this paper is written on the logits z. The chain rule turns its gradient with respect to z into a gradient with respect to the pixels by multiplying with the transpose of the Jacobian of the detector, the matrix of how every logit depends on every pixel. The Jacobian is never formed explicitly; backpropagation applies it. It exists wherever the detector is differentiable, which is everywhere that matters.

A. Anything the detector computes is one more term in the loss

Each defence we tried, written as the statistic it reads. Every one is a function of the same pixels x, so its gradient is obtained exactly as in the previous equation.

$$\begin{aligned} s_{\text{probe}}(x) &= \max_{i,c} z_{ic}(x + \sigma \xi) - \max_{i,c} z_{ic}(x), \\ \xi &\sim \mathcal{N}(0, I) \end{aligned} \quad (22)$$

Noise-recovery probe. Add Gaussian noise of scale sigma and read how much the frame's strongest logit comes back. A suppressed frame recovers, a clean one does not. The attacker adds this to its loss and suppresses at the noisy frame as well as at the clean one.

$$\begin{aligned} s_{\text{anchors}}(x) &= |\mathcal{A}_i(x)| \\ \mathcal{A}_i(x) &= \{j : z_{jc}(x) > \tau, \text{IoU}(b_j, b_i) > 0.5\} \end{aligned} \quad (23)$$

Anchor agreement. Count the positions that report the same class above threshold with a box overlapping the reported one. The premise was that real objects gather many; this head assigns one position per object by design, so genuine detections have a count of one and there is nothing to separate.

$$\begin{aligned} s_{\text{crop}}(x) &= z_{ic}(x) - z_c(\text{crop}_{\rho}(x, b_i)), \\ s_{\text{flip}}(x) &= z_{ic}(x) - z_{ic}(\text{flip}(x)) \end{aligned} \quad (24)$$

Crop-and-re-query, and the flip gate. The first cuts out the reported box with a margin rho, resizes the crop, and reads how much the logit drops; the second flips the frame left to right and reads the same drop. Both are differentiable functions of x, the first through the crop and resize, the second through the flip, and one weighted term in the attacker's loss makes either drop vanish.

$$\begin{aligned} s_{\text{bank}}(x) &= \min_{k=1..K} z_{ic}(T_k(x)), \\ s_{\text{sub}}(x) &= \max_{j \in B} [z_{jc}(x + \sigma \xi) - z_{jc}(x)] \end{aligned} \quad (25)$$

The transform bank, and sub-threshold recovery. The bank demands that the detection survive all K transforms, so it reads the worst of them; K terms in the loss satisfy it. Sub-threshold recovery looks only at the band B of positions sitting just under the reporting line and asks whether noise lifts any of them; the attack leaves nothing there, it buries the removed object far below the band.

$$s_{\text{verify}}(x) = g_{\phi}(x)_c(b_i),$$

$$s_{\text{mask}}(x) = \min_{m \in \mathcal{M}} z_{ic}((1 - m) \odot x) \quad (26)$$

A second detector, and certified masking. The verifier reads a second model's score for the same class at the same box; it is differentiable in x and adversarial examples transfer between models, so it becomes a term like any other. Certified masking reads the lowest logit over a family $\mathcal{M}$ of masks placed so that some mask covers any possible patch; a genuine object also vanishes under the mask that covers it, so the statistic cannot tell them apart.

The mechanisms tried, and the logic of each. A **noise-recovery probe**: a perturbation tuned to one part in 255 is fragile, so add noise and watch the suppressed detections return; the attacker suppresses at the probe as well, at no extra cost. An **anchor-agreement field**: a real object should be seen by many anchors; this head assigns one anchor per object by design, so there is nothing to read. **Crop and re-query**: a fabrication depends on the whole frame, so crop the box and ask again; the second pass, not the crop, did the work, and on the held split the gate kept 0.798 of genuine detections against a bar of 0.80, so it was withdrawn before any adaptive attack. A **flip gate**: flip the frame and require the detection to survive; one term in the loss. A **bank of eight transforms**: eight terms, one constraint. **Sub-threshold recovery**: look for evidence just under the reporting line; the attack leaves none, burying the removed object far below it, and the gate fails before any attacker adapts to it. A **second detector**: transferability. **Certified masking**: sound, and blind to fabrication. Table 9 gives every number.

TABLE 9. Every defence built from the detector, with what it read, how it did against an attacker that ignored it, and how it did against one that put it in the loss. AUC is directed, attacked over clean, 0.5 is chance; evasion is the share of attacks that succeed and pass the gate. Budgets 16/255 for fabrication, 1/255 for suppression. Split and n as stated.

defence	reads	attacker unaware	attacker aware
noise-recovery probe	max logit after added noise	AUC 0.958 on suppression, catch 0.90 at zero FA ($n = 20$, tuning)	caught 0.92 $\rightarrow$ 0.76 when the attacker suppresses at the probe too ($n = 25$); unbounded attacker caught 0.080
sub-threshold recovery	noise lift of the band just under the line	AUC 0.516 on selective removal ($n = 30$, tuning): fails unadapted	not needed
anchor agreement	anchors agreeing on the box	AUC genuine over fabricated 0.676 / 0.715 ($n = 14$ images, tuning): fails unadapted	not needed

defence	reads	attacker unaware	attacker aware
crop and re-query	logit drop on the re-queried crop	held: AUC 0.938 on the 16/255 field but genuine survival 0.798 < 0.80 bar ($n = 100$): withdrawn	not run
flip gate	logit drop after a horizontal flip	held ($n = 100$ PGD, 300 cheap): AUC 0.994 (16/255 field), 0.996 (4/255), 0.713 (universal patch), 0.704 (stealth); control 0.526	evasion 0.025 $\rightarrow$ 0.975 ($n = 40$, tuning)
bank of 8 transforms	worst logit over the bank	—	evasion 1.000 ($k = 1$), 0.933 (2), 0.800 (4), 0.867 (8); $n = 15$, tuning
second detector (Faster R-CNN R50-FPN, 41.8 M)	verifier's person score at the box	AUC 0.972 (16/255 field), 0.971 (4/255), 0.938 (universal), 0.864 (stealth); $n \leq 30$, tuning	held ($n = 200$): evasion 0.005 $\rightarrow$ 0.755 at 8.3 $\times$ wall-clock; stronger attacker 0.933 ($n = 15$)
certified masking (49 masks)	lowest logit over the mask family	catch 1.000 on 41 fabricated boxes; flags 1.000 of 129 genuine ($n = 40$, tuning)	not needed
quiet-output test	max logit against the threshold	catch 19/20 of total suppression, its own success count ($n = 20$, tuning); AUC 0.998	unbounded attacker caught 1.000 ($n = 25$); selective removal caught 0.038 ($n = 26$ successes)

Reading the table. Two defences fail before any attacker adapts to them, because the quantity they read does not separate attacked from clean frames; one was withdrawn when it missed its own false-positive bar on the held split. The three that separated well, the flip gate, the transform bank and the second detector, were each put into the attacker's loss and evaded at 0.755 to 0.975. Certified masking catches everything and flags everything. The quiet-output test is the one row whose adaptive column is not an evasion: it catches the unbounded attacker in full, and it covers total suppression only.

A defence inside the detector reads a quantity of the forward pass and gates on it: the response after a left-right flip, agreement across a bank of transforms, recovery under added noise, the number of anchors agreeing on a box, the answer to a re-query on a crop. Each is a differentiable function of the pixels the attacker writes; none of the defences built here needed a surrogate. The attacker adds it to the loss with a weight and descends. Where a defence is not differentiable, a smooth surrogate on the backward pass gives the same result [24]. This was established for classifiers in 2017 and 2018 [25], [24]; here it is reproduced on a detector.

The flip gate is the clean case. Against an attacker that ignores it, it separates attacked from clean frames at AUC 0.994 on 100 held frames (16/255 field). With the flip in the

loss, at the same budget on 40 tuning frames, evasion rises from 0.025 to 0.975 and the fabrication rate does not move. Eight transforms instead of one do not help: in a separate run on 15 tuning frames, evasion is 0.867 with eight against 1.000 with one, because eight views of one image are one constraint, the reason expectation over transformation defeats input transforms [20].

B. A second model is the same term with a different weight

$$\begin{aligned} L'(x) &= L(x) + \lambda s(x), \\ L''(x) &= L(x) + \sum_{k=1}^K \lambda_k s(T_k(x)) \end{aligned} \quad (27)$$

The attacker's answer to any of them. Take the attack loss L , add the defence's statistic s with a weight λ , and run the same projected descent. The descent now lowers both at once. A bank of K transformed copies adds K terms, but they are K functions of one picture, and the descent satisfies them together at little extra cost: many views of one image are one constraint, not many.

Asking a second, differently trained detector to confirm each box looks like an escape from the argument and is not. Its score is a differentiable function of the input, and adversarial examples transfer between models trained on the same task [2], [18]. The verifier is a Faster R-CNN with a ResNet-50 FPN backbone, 41.76 million parameters. On 200 held frames at 16/255 an attacker unaware of it succeeds 0.005 of the time (1/200); one that puts it in the loss succeeds 0.755 [0.691, 0.809], at $8.3\times$ the wall-clock; with more steps, restarts and noise over the verifier, 0.933 on 15 frames. A defence evaded three times in four does not hold. What it keeps is a price: the attacker's success falls from 0.985 to 0.755, a reduction of 0.230 [0.168, 0.292], not a wall.

C. What is ruled out by argument, and one experiment

Certified defences prove that no perturbation below a radius, or no patch below a size, changes the output; for detectors the certified constructions address the hiding attack by masking [12], [11]. Their patch form, mask every position and watch what disappears, cannot recognise fabrication: a real object disappears under a mask too. We built the sound version, 49 masks of 266 pixels at stride 64 so that some mask covers any 202-pixel patch, on 40 tuning frames: it flags every fabricated detection (1.000 of 41) and every genuine one (1.000 of 129), at 49 forward passes per frame. Adversarial training [4], the one style of defence that has survived re-evaluation and has been carried to detectors [13], [14], was not attempted: it needs the attack style in advance, it moves the decision boundary rather than removing it, and it retrains the detector, which this work holds fixed.

D. The one test that holds, and why

$$\begin{aligned} s_{\text{quiet}}(x) &= \max_{i,c} z_{ic}(x) - \tau, \\ L_{\text{van}}(x) &= 0 \Leftrightarrow s_{\text{quiet}}(x) \leq -\kappa \end{aligned} \quad (28)$$

Why the quiet-output test cannot be evaded. The test reads the frame's largest logit against the threshold. By the definition of the vanishing loss, that loss is zero exactly when every position's strongest logit is at least κ below the threshold, which is the same statement as the test firing. The attack has succeeded if and only if the test fires. There is no weight λ to choose: the statistic is the objective.

A suppression attack minimises the strongest class logit in the frame until it sits below the reporting threshold. A test that reads whether the strongest class logit is below the reporting threshold reads the attack's own objective: there is no direction of descent that satisfies the attack and not the test. On 20 tuning frames total suppression succeeded on 19 and the test fired on those 19, with its threshold set for zero false positives on that population; on 500 clean tuning frames it fires on 0.040 (20/500), of which 10 are frames the detector genuinely leaves empty, so its floor false-positive rate is the clean empty-frame rate. It catches 1.000 of an attacker with no budget at all (25/25) and costs one comparison. It does not extend. Remove one object and keep the rest and the frame is ordinary: that attack, a hinge that drives the target's logit below the line while a second hinge with weight one holds every other clean detection above it, projected gradient descent at step $\epsilon/16$ for 120 steps on 30 tuning frames with at least two detections, succeeds on 0.867 of frames at 1/255 (26/30) and is caught on 0.038 of those (1/26).

Everything else inside the detector reads a quantity the attacker can move for free. Section VIII reads one it cannot reach without knowing it is there: whether the pixels under a box are a photograph. A loss written on the detector's logits says nothing about that; a network that has only seen photographs does.

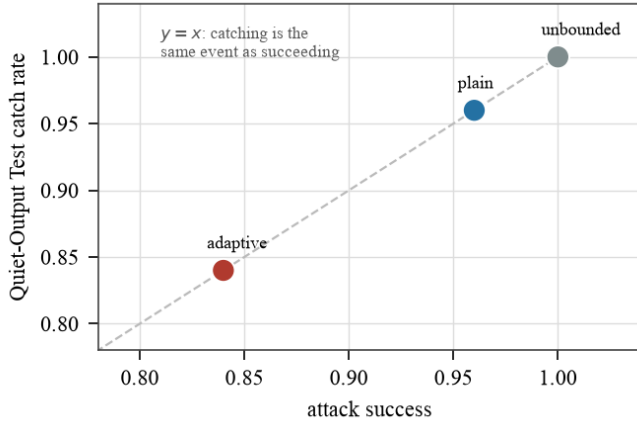


Fig. 31. Catch rate of the quiet-output test against attack success rate for three attacker strengths on 25 tuning frames at 1/255: plain projected gradient descent, the same attack that also suppresses at three noise probes (the attacker adapted to the noise-recovery probe of Table 9), and the same with no pixel budget. Every point lies on $y = x$: the attack succeeding and the test firing are the same event, so no attacker strategy separates them.

VIII. A DIFFUSION PRIOR

What this section shows. If the attacker can steer anything the detector computes, a defence must read something the detector never computes.

We chose whether the content inside a reported box is a photograph at all. A small network is trained on nothing but ordinary photographs to remove noise added step by step; a network that can do this has learned what photographs look like, and shown content that is not a photograph, its prediction error rises measurably. Placed beside the detector, it scores every reported box. The result is stated in full: the reusable patch, a synthetic pattern, stands out and is localised; the bounded full-frame perturbations are not separated from genuine detections; the patch cut from a real photograph is passed, and we explain why no training changes that. Every number is against attacks built without knowledge of the denoiser. The section closes with what the readout costs and what the network is and is not.

What follows reads a quantity the attacker's loss does not contain. We train a small network to remove noise from ordinary photographs. It never sees an attack, a patch, or the detector. Removing noise from a picture requires knowing what a picture looks like; at test time the network is shown the pixels inside each box the detector reports and its prediction error is measured. That error answers a question absent from the attacker's loss: are these pixels a photograph? The readout, a denoising residual read as an anomaly score, follows prior work on diffusion-based anomaly and out-of-distribution detection [15], [16]; what is measured here is its use per reported box, under an adversary, and the resolution it has.

The result, stated in full before the details. Four fabrication families meet the test on frames whose ground truth contains no person (VIII-E). The universal adversarial patch is is

separated and localised, and a zero-parameter texture statistic inside the box separates it nearly as well, because the patch is a texture anomaly. The two bounded full-frame perturbations are not separated from genuine detections: the 16/255 field reads as more natural than random noise at the same bound (VIII-F) and the 4/255 field sits at the resolution floor (VIII-C). The stealth patch is passed: the network is asked whether the region is a photograph, and it is (VIII-G). Every number is against attacks built with no knowledge of the denoiser.

A. The denoiser

Method. The network is a U-Net with 1,013,091 parameters. Its only job is to guess the noise we added to a photograph. Each training step takes a 128×128 crop of a COCO train2017 image [26] at native resolution, picks a noise level t at random from $\{0, \dots, 999\}$, draws noise $\epsilon \sim N(0, I)$, and mixes the two: $x_t = \sqrt{a_t} \cdot x_0 + \sqrt{1-a_t} \cdot \epsilon$. Here a_t is the running product of $1-\beta$ on a linear β schedule from $1e-4$ to 0.02 , so a larger t means more noise. The network sees x_t and t and guesses ϵ . The loss is the squared error of that guess: the training objective and noise schedule of a denoising diffusion model [27], unchanged. **That is the whole loss. No extra head, no regulariser, no detector in the loop, no adversarial example.** AdamW at learning rate $1e-4$, weight decay $1e-5$, 500 warm-up steps then cosine decay to zero, gradient clipping at norm 1.0, batch 16, and an exponential moving average of the weights at 0.999. The averaged weights are the model.

A network trained on that loss is a denoising score matcher [28]: at each noise level its best estimate of ϵ is, up to scale, the gradient of the log-density of noisy photographs [29]. Its error is a statistic of the photograph distribution as this finite network learned it, and that error is what every readout below reads. The training set is COCO train2017, 118,287 images, which is also the detector's training set: the denoiser never sees the detector, but the two share their training data. Validation loss is measured on a held-out portion of the same crops.

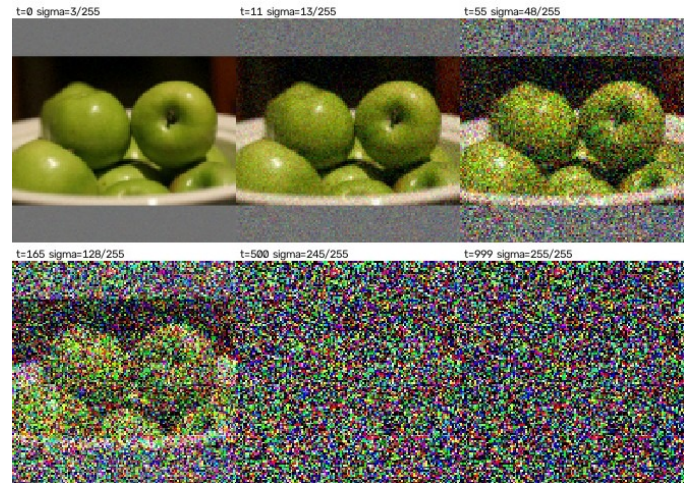


Fig. 32. One photograph at six points on the noise schedule. The probes used at deployment lie between $t=11$ and $t=165$, the second to fourth panels. At $t=999$ nothing of the image remains.

The U-Net has two downsampling stages by average pooling, a bottleneck with self-attention on a 32×32 grid, two upsampling stages by pixel shuffle with skip connections, and a sinusoidal encoding of t added in every block. All modules except attention total 946,787 parameters; attention is the remaining 66,304.

A second model serves as the comparison throughout: trained on whole images resized to 128×128 instead of native crops, for eight epochs instead of two. We call it the resize-trained baseline.

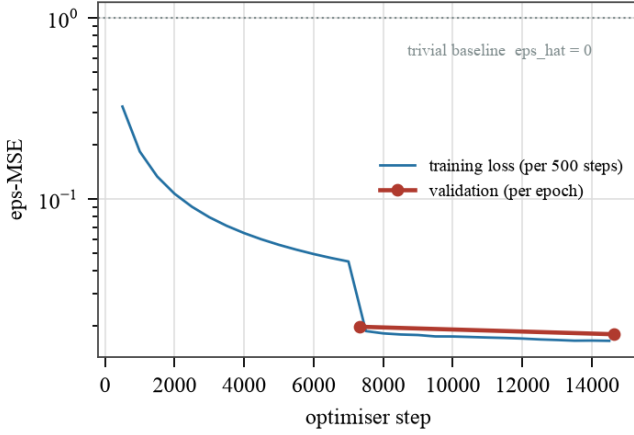


Fig. 33. Training loss, seed 42, two epochs of 7,319 steps of 16 crops each (118,287 images, the last partial batch dropped). The level shift at the epoch boundary is the first epoch's running mean, which includes the warm-up steps, giving way to the second's; the validation losses at the two epoch ends are 0.020 and 0.018, with no jump. A network that guessed zero noise every time would score 1.0.

Effect. Validation loss reached 0.020 after one epoch and 0.018 after two. Guessing zero noise scores 1.0 on the same batches. Two epochs is where the resolution floor stops moving quickly (VIII-C): on the baseline recipe the floor at the most sensitive probe falls from 3.64/255 after one epoch to 3.29 after two, 3.13 after three, 3.06 after four and 2.96 after eight. The second epoch delivers half of the eight-epoch gain, the third five per cent, the fourth two, and from the fifth on under one and a half per cent each.

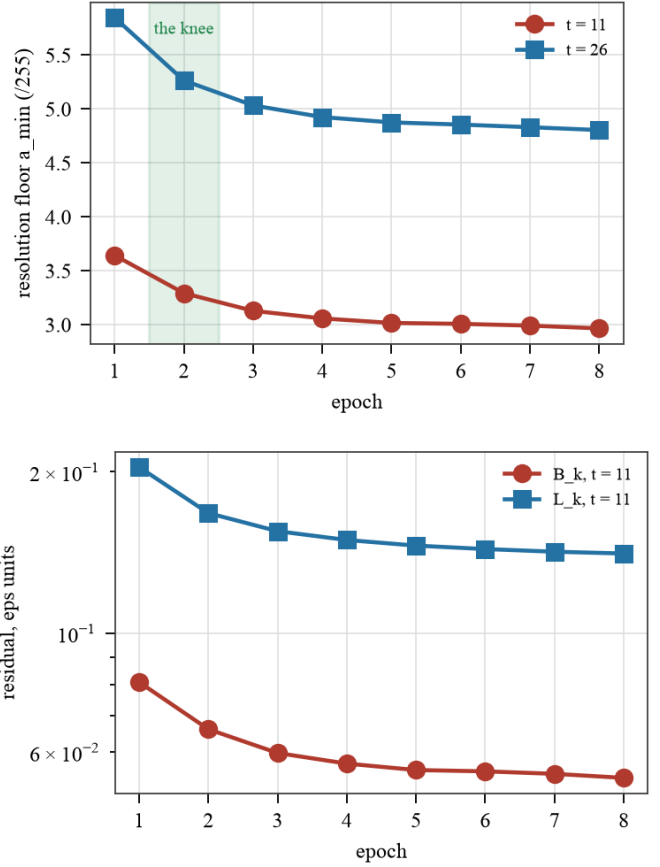


Fig. 34. Why two epochs. Top: the resolution floor over eight epochs of the baseline recipe, at two probes. The second epoch delivers half of everything epochs two to eight deliver together. Bottom: the prediction error and its draw-independent floor over the same run.

Brownian motion, forwards and backwards. The added noise is Brownian: each step of the schedule adds an independent Gaussian increment, so a photograph walked from $t = 0$ to $t = 999$ traces a discretised Wiener path from the picture to pure noise; with its shrinking drift the forward process is a variance-preserving Ornstein–Uhlenbeck diffusion driven by that motion. Training the network to predict the noise at every point of the path is training it to run the motion backwards: the reverse-time diffusion has a drift equal to the gradient of the log-density of noisy photographs, the score, and the noise predictor is that score up to a known scale [29], [28]. A network that has only seen photographs can therefore judge a region it has never seen, because it has learned, at every noise level, the direction towards a photograph and the distance from one.

The census of VIII-C uses that geometry to ask how much the eight noise levels tell the reader beyond one another. There the eight probes are eight points on one Brownian path, so their residual maps are correlated by construction: adjacent probes at 0.82, probes seven steps apart at 0.31. A one-step model built from the schedule alone predicts 0.72 falling to 0.10; the measured correlation sits above it at every lag, because the image-dependent part of the residual is shared

along the whole path. The squared-correlation estimate of independent maps among the eight is about two. The deployed readout draws each probe's noise independently and averages all eight; the per-probe separation of VIII-C shows that the universal patch is separated equally at every probe, so fewer probes would serve, but no such reduction is made here.

$$dx = -\frac{1}{2}\beta(t)x dt + \sqrt{\beta(t)} dW_t, \quad t \in [0, 1] \quad (29)$$

Forward diffusion, in continuous time. A photograph x is pushed along a Brownian path. The term with dW is Brownian motion: at every instant an independent Gaussian increment is added, scaled by the schedule beta. The drift term shrinks the picture slightly as it goes, so that the variance stays bounded. Run to t equal to one the picture is indistinguishable from pure Gaussian noise.

$$x_t = \sqrt{\tilde{\alpha}_t} x_0 + \sqrt{1 - \tilde{\alpha}_t} \varepsilon, \quad \tilde{\alpha}_t = \prod_{s=1}^t (1 - \beta_s), \quad \varepsilon \sim \mathcal{N}(0, I) \quad (30)$$

The same process, in the discrete steps used to train. After t of the 1,000 steps, the noisy picture is a known mixture of the original x_0 and one standard Gaussian draw epsilon. The mixing weight alpha-bar is the running product of one minus beta over the steps so far; beta rises linearly from 0.0001 to 0.02. Because the mixture is known in closed form, any noise level can be sampled directly without simulating the path.

$$dx = [-\frac{1}{2}\beta(t)x - \beta(t)\nabla_x \log p_t(x)]dt + \sqrt{\beta(t)} d\tilde{W}_t \quad (31)$$

The reverse-time process. Brownian motion run backwards is again a diffusion. Its drift contains one term the forward process does not: the gradient of the log-density of noisy photographs at noise level t , called the score. Everything else in the equation is known from the schedule. So the only thing a network has to learn in order to run the motion backwards, from noise towards a photograph, is the score at every noise level.

$$\varepsilon_\theta(x_t, t) \approx -\sqrt{1 - \tilde{\alpha}_t} \nabla_x \log p_t(x_t) \quad (32)$$

Noise prediction is score estimation. For the Gaussian mixture above, the score at x_t equals minus the noise that was added, divided by the noise scale. A network trained to predict the noise therefore predicts the score up to a known factor. This is the identity that makes a denoiser a model of what photographs look like.

$$\mathcal{L}(\theta) = \mathbb{E}_{x_0, t, \varepsilon} [\|\varepsilon - \varepsilon_\theta(x_t, t)\|^2] \quad (33)$$

$$x_t = \sqrt{\tilde{\alpha}_t} x_0 + \sqrt{1 - \tilde{\alpha}_t} \varepsilon$$

The single training loss. Draw a photograph, a step t , and a noise sample; mix them; ask the network for the noise; take the squared error. That is the entire objective. No attack, no detector, no adversarial example, and no second term enters it. The network learns which direction, at every noise level, points towards a photograph, and nothing else.

B. How it is scored

Threat model for verification. The attacker of Part A is unchanged: white-box on the detector, writing the file, the tensor or a bounded region. It knows a denoiser is present, because this paper says so, but does not hold its weights and does not put it in the loss. Every number in this section is therefore a measurement against attacks built without the denoiser, which by the standard of Section VII is the weaker test. The stronger one, an attacker that adds the residual to its objective, is the first experiment this paper does not contain (IX-C).

The target, images and scoring path are those of Sections II and IV, cfgB throughout. The detector is frozen; the denoiser neither modifies it nor communicates with it, and reads only the pixels inside the boxes it reports. This is detection, not purification: diffusion purification [30] denoises the frame and re-runs the detector, whereas here the frame is never touched. A random control at the same bound runs on the same frames and the directed AUC is never folded. The four attack arms are built per frame: the 16/255 and 4/255 full-frame fields by projected gradient descent from zero with a margin-maximising objective on the person logit, 120 steps at step $\max(\varepsilon/20, 0.3/255)$ on photograph pixels only, so every step carries gradient (these are not the regional attack of VI-E); the universal patch of VI-C and the stealth patch of VI-E pasted at the centre at ten per cent of the frame, as fixed files.

The readout was fixed in a protocol written on 2026-09-17, before the deployed model was trained and before any detector test. We cut the detector's 640×640 letterboxed frame into 25 native 128×128 tiles and probe each tile at eight noise levels, $t = 11, 17, 26, 38, 55, 79, 113, 165$, a geometric ladder in $\sigma = \sqrt{1-a}$ from 0.050 to 0.502, with one independent random draw each. We average the squared prediction error over the eight probes to get a full-resolution 640×640 map. The score of a reported box is the average of that map over the box's pixels. No per-image normalisation, no learned combination, no free parameter. Letterbox tiles are scored like any other; no reported box falls in the padding. A second, two-sided reading used in VIII-G scores the distance of a box's residual from the median residual of genuine boxes, which is a normalisation by that median and is named as such where it appears.

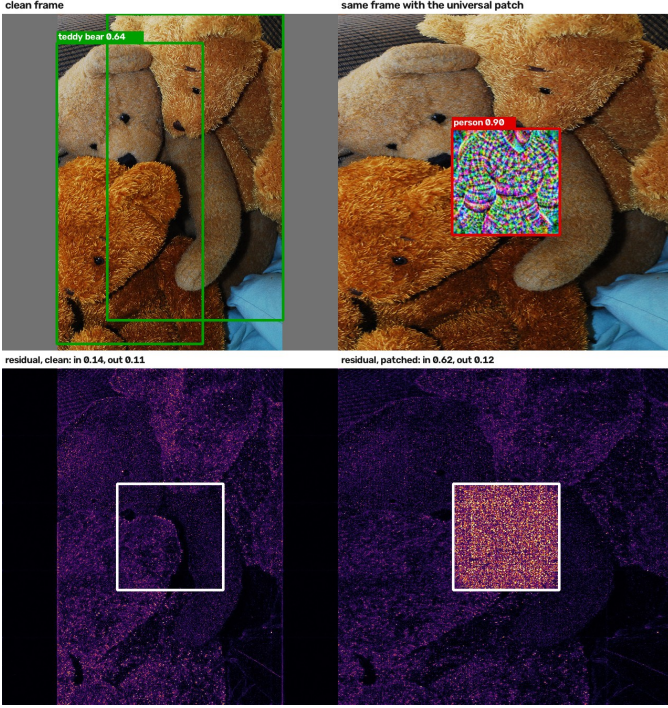


Fig. 35. The same tuning-split frame twice. Top left: clean, with the detector's own reports in green. Top right: with the universal patch pasted at the centre; the detector now reports a person, in red, where there is a teddy bear. Bottom: the denoiser's residual map for each frame, all eight probes averaged, on one shared colour scale, with the white square marking where the patch sits. The clean map is faint everywhere. Under the patch the map is bright inside the square and unchanged outside it: the mean residual inside the square rises from the clean value to more than four times it, as printed on the plate, while the outside barely moves. That inside-versus-outside contrast is the whole detector test, seen on one frame.

$$r_u(x) = \frac{1}{K} \sum_{k=1}^K \frac{1}{3} \sum_{ch=1}^3 e_{k,u,ch}^2 \quad (34)$$

$$e_k = \varepsilon_\theta(x_{t_k}, t_k) - \varepsilon_k$$

The residual map. Each probe k adds noise at its level t_k , asks the network to guess that noise, and records the squared error at every pixel u , averaged over the three colour channels. The map is the average over the K probes. Where the picture is a photograph the guess is good and the residual is small; where it is not, the guess is poor and the residual is large.

$$S(\text{box}) = \frac{1}{|\text{box}|} \sum_{u \in \text{box}} r_u(x) \quad (35)$$

The score of a reported detection. The residual map is averaged over the pixels inside the box the detector reported. That single number is compared, as directed AUC, between boxes the attack fabricated and boxes the detector reports on clean frames. There is no threshold, no normalisation per image and no learned combination.

$$K_{\text{eff}} = \frac{K^2}{\sum_{i=1}^K \sum_{j=1}^K \rho_{ij}^2} \quad (36)$$

How many independent probes the K probes are worth. When the K probes share one noise path their residual maps are correlated, with coefficient ρ between probes i and j . K_{eff} is the squared-correlation estimate of the number of independent maps among the K : it equals K when every off-diagonal ρ is zero and 1 when all maps are identical. It is measured in the census along a shared path (VIII-C) and comes to about two of eight. The deployed readout draws each probe's noise independently, so its probes share only the image-dependent part of the residual; K_{eff} is the census's statement about how much the eight noise levels tell the reader beyond one another, not a description of the deployed draw.

C. What the network learned

Method. A census on 200 held-out native crops, with 4 independent noise draws at each of the eight probes. These are the quantities that bound every readout that follows.

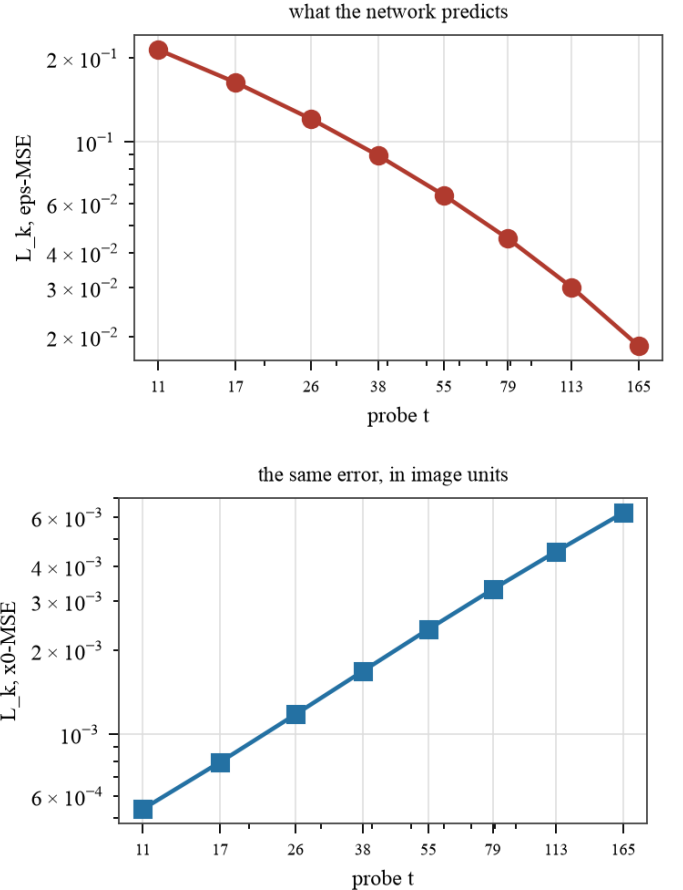


Fig. 36. Prediction error at each probe, in the units the residual is read in and in image units. The network works hardest at the low-noise end, where there is little noise to find.

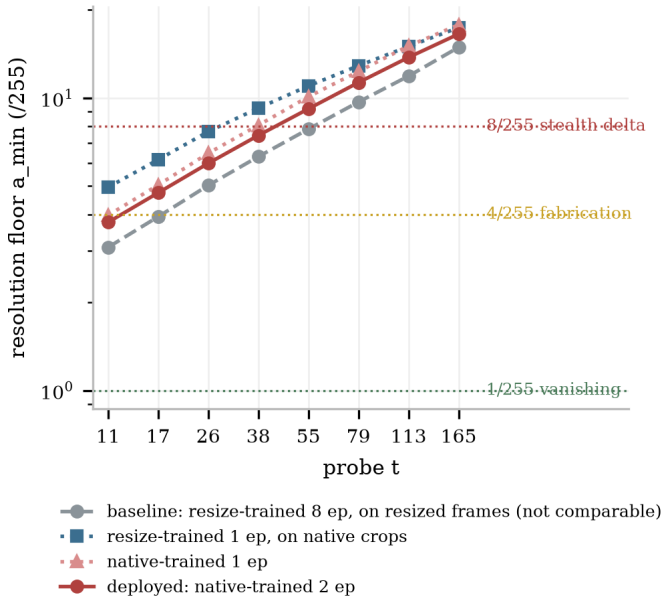


Fig. 37. The resolution floor: the smallest pixel change the residual can see at each probe, in the units of the network's input, with the budgets of the 1/255 suppression, 4/255 fabrication and 8/255 stealth attacks marked. Three models are measured on the same 200 native crops: the resize-trained network after one epoch, the native-trained network after one, and the deployed native-trained network after two. At equal training, native-resolution crops lower the floor at every probe, and the second epoch lowers it further. The grey line is the resize-trained baseline after eight epochs measured on resized whole frames, its own input domain; it is drawn for reference and is not comparable to the other three, because a native change arrives attenuated in a resized frame. The floor decides which attack families are visible at all.

Effect. The network's prediction is a conservative field (VIII-A). Its normalised Jacobian asymmetry is $1.1e-04$, the necessary condition for being the gradient of a potential, which the true score is; here it is measured, not assumed. Prediction error falls from 0.214 at $t=11$ to 0.018 at $t=165$. **The resolution floor** at the most sensitive probe is 3.76/255 for the deployed network; on the same native crops the resize-trained network after one epoch has 4.95/255 and the native-trained network after one epoch 3.97/255. The residual is local: its correlation length is 1.0–2.0 pixels, at the pixel grid's own resolution at the five lowest probes, so a high value at one pixel says almost nothing about a pixel two steps away. And along one shared noise path the squared-correlation estimate of independent maps among the eight probes is 2.05.

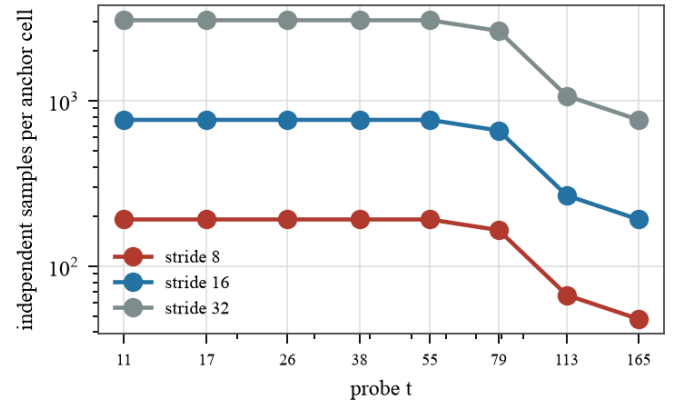
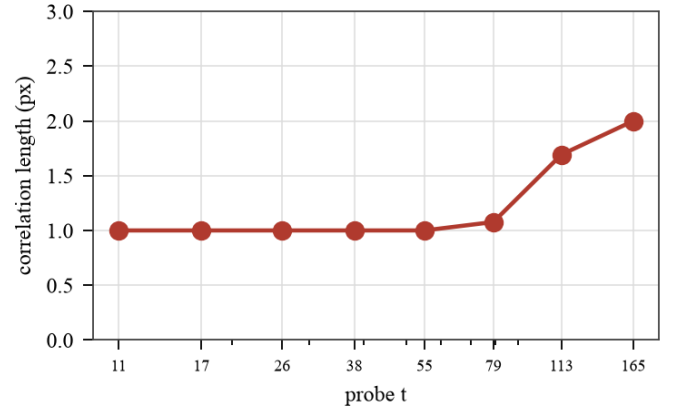


Fig. 38. Top: correlation length, one to two pixels across the probe grid. Bottom: the number of independent samples that gives per anchor cell at each of the detector's three strides. Fine pooling is justified.

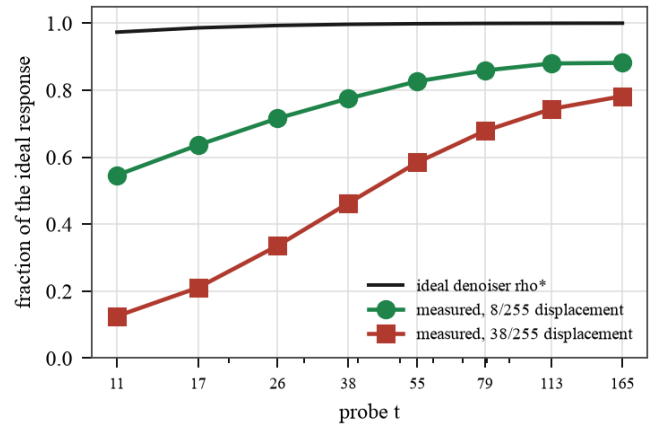


Fig. 39. Response to a known pixel change, against the ceiling an ideal denoiser would reach. A random-sign field of amplitude 8/255 or 38/255 is added to each of 100 crops; the response is the rise in the residual divided by the rise a perfect detector of that change would show, $\alpha\text{-bar}$ times the amplitude squared over sigma squared, so 1 is full response. The ceiling is what an ideal denoiser of a photograph manifold of thickness 1.5/255 would reach. The network recovers most of the ideal response at 8/255 and saturates at large amplitude: the calibration is nonlinear.

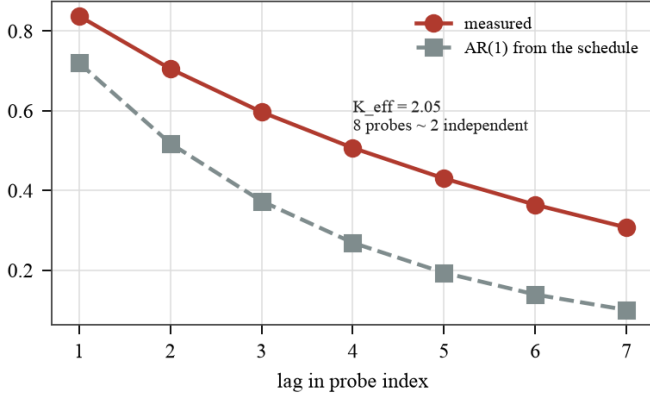


Fig. 40. How quickly the denoiser's error at one noise level stops resembling its error at another, along one Brownian path through the schedule. Horizontal axis: distance between two probes, in probe steps (adjacent probes are one step apart). Vertical axis: correlation of their residual maps, from 1 for identical maps to 0 for unrelated ones. Red: measured on 200 held-out crops. Grey: what a one-step model predicts from the noise schedule alone, 0.72 falling to 0.10. The measured correlation sits above the prediction at every lag, 0.82 for neighbours and 0.31 seven steps apart, because the part of the residual that depends on the image is shared along the whole path; the squared-correlation estimate of independent maps among the eight is about two.

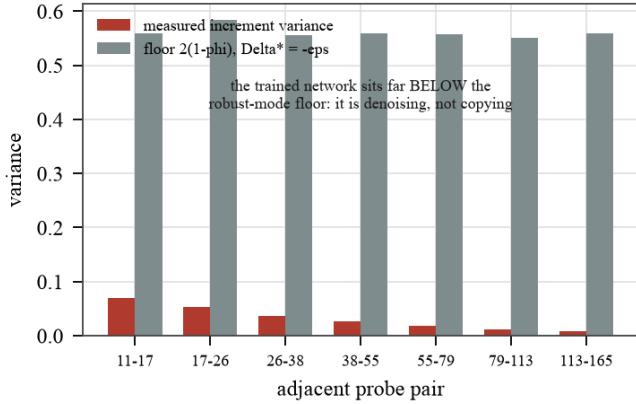


Fig. 41. Variance of the step between the residual maps of adjacent probes along the shared path, against the value a network that returned its input noise unchanged would produce, $2(1 - \phi)$ with ϕ the correlation of adjacent noise draws, about 0.56 at every lag. The trained network sits well below it, at 0.07 at the lowest probes and 0.008 at the highest, eight to seventy times smaller: most of the noise is removed and what changes between probes is small.

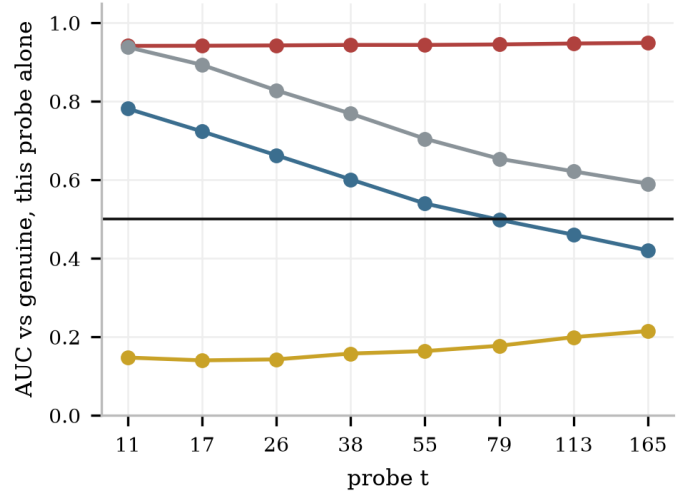


Fig. 42. Separation at each probe on its own, one-sided AUC against genuine boxes. The universal patch is separated equally at every probe. The 16/255 adversarial field and the random control at the same bound both fall as probe noise rises, and the adversarial field sits below the control at every probe. The deployed readout averages all eight probes; this figure shows that the universal-patch result does not depend on which are used.

$$a_{\min}(t) = \sigma_t \sqrt{\frac{B_t}{\tilde{\alpha}_t}}, \quad \sigma_t = \sqrt{1 - \tilde{\alpha}_t} \quad (37)$$

The resolution floor, and where it comes from. B_t is the network's own mean squared prediction error on clean photographs at probe t , the part of the residual that is there whether or not anything is wrong. A change of amplitude a in the photograph x_0 enters the noisy picture x_t scaled by the square root of α -bar, and shifts the noise the network is asked to recover by that amount divided by σ_t . Its squared contribution to the residual is therefore α -bar times a squared over σ squared, and it is resolvable only when that exceeds the floor B_t ; solving for a gives the expression. The floor scales with the noise level of the probe. At the most sensitive probe it is 3.76 grey levels, so a 4/255 change sits at the floor there and below it at every other probe.

$$\frac{\partial \varepsilon_{\theta}^{(v)}}{\partial x^{(u)}} = \frac{\partial \varepsilon_{\theta}^{(u)}}{\partial x^{(v)}} \quad \forall u, v$$

$$\Leftrightarrow \varepsilon_{\theta} = \nabla_x \Phi \quad \text{for some } \Phi \quad (38)$$

The conservative-field test. A vector field is the gradient of some scalar function Φ if and only if its Jacobian is symmetric. The true score is such a gradient, of the log-density, so symmetry is a necessary condition for the network to be a score model; it is not sufficient. The census measures the normalised asymmetry: for random sign vectors u and v it takes the difference between u -transpose $J v$ and v -transpose $J u$ by central differences, divides by the mean magnitude of u -transpose $J v$, and averages over 100 crops and the eight probes. The value is 1.1 times ten to the minus four. To that precision the network's prediction is a conservative field, which is what every statement above that rests on the score requires of it.

D. The photograph test

Method. Take a 128×128 native crop and replace a 40×40 square, ten percent of its area, with one of six kinds of content. Measure the directed AUC of the residual inside the square against outside, on 200 crops, at three pooling strides. Two of the six fillings are photographs. Four are not.

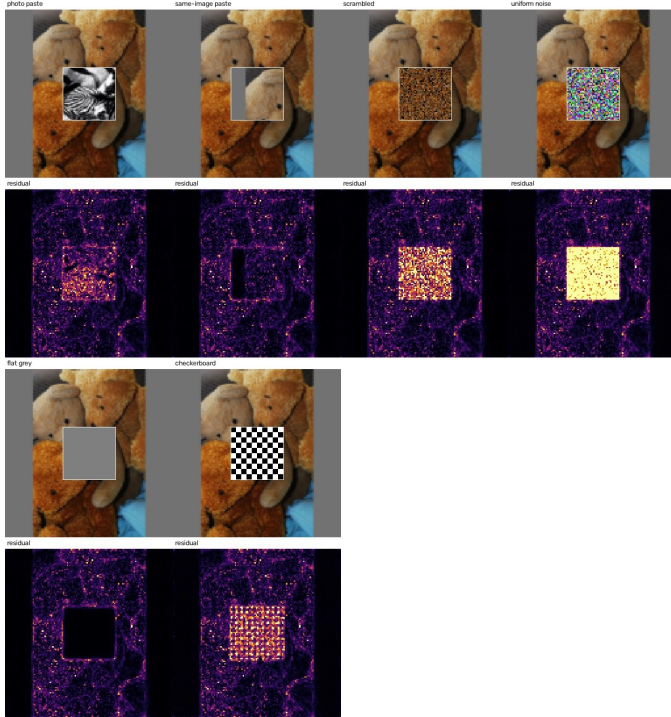


Fig. 43. The six fillings and the residual each produces. Photographic content is ignored. Structured and unstructured non-photographic content is flagged. Flat grey is flagged in the opposite direction, because there is nothing to denoise.

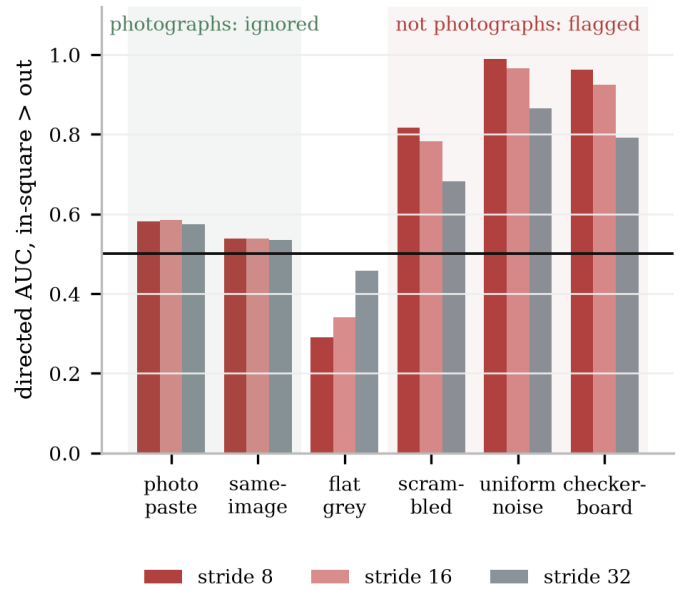


Fig. 44. The same six as directed AUC at three pooling strides, for the deployed network on 200 crops. The statistic is two-sided: too random and too regular both sit away from photograph statistics, on opposite sides.

Effect. A crop of another photograph scores 0.582 and a crop of the same image 0.537, both near chance: photographs are ignored. Scrambled pixels score 0.817, uniform noise 0.988, a 4-pixel checkerboard 0.961. Flat grey scores 0.291, far below chance.

Consequence. The residual measures distance from photograph statistics in both directions. Sub-section VIII-F is a family that moves *toward* the photographic side; VIII-G is a family that was never on the other side.

E. The detector test

Method. The denoiser sits beside YOLO26 and scores every detection it reports (VIII-B). The test uses 25 frames of the tuning split whose ground truth contains no person and which carry at least one clean detection; the denoiser trained on train2017 and has never seen any val2017 image, and the split matters only because the detector attacks in this test use hyperparameters set on it. Any person reported under attack is fabricated by construction. Populations: genuine is every detection of any class on the clean frames (91 boxes, of which 2 are person-class reports the ground truth does not support; excluding them changes no AUC below by more than 0.003); fabricated is every person-class detection on the attacked frame, with no cap per frame; the control is every detection on the frame with 16/255 uniform noise added. Four attack families and the control: 27 boxes under the universal patch at ten per cent, 26 under the stealth patch, 75 under the 16/255 field (three per frame), 52 under the 4/255 field and 74 under the control. Two further arms added in a re-run with per-box logging: the universal patch at four and six per cent (24 and 26 boxes) and the stealth patch's anchor crop pasted unmodified, which fabricates 1 box in 25 frames. Every AUC carries a 95% interval from 2,000 bootstrap resamples of frames.

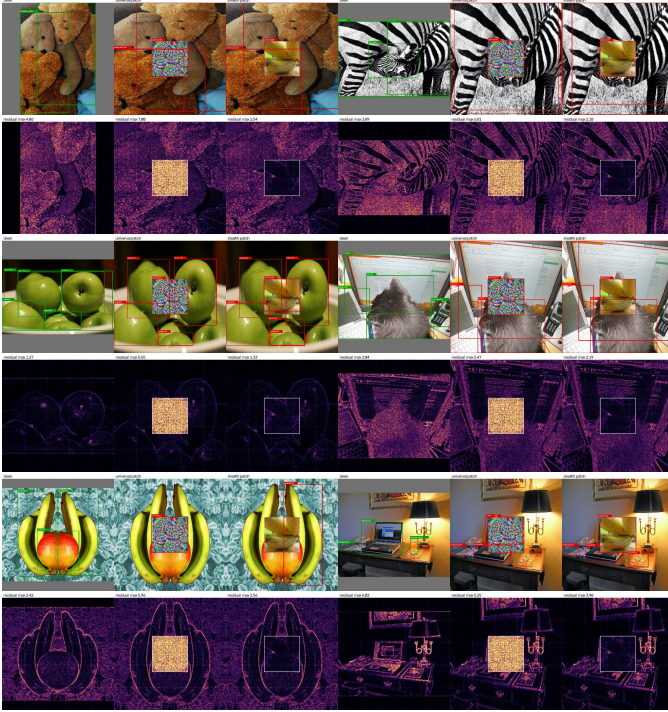


Fig. 45. Six frames, each shown clean, with the universal patch, and with the stealth patch; below each, the full-resolution residual map. The universal patch is the bright square. The stealth patch, outlined in white, cannot be told from its surroundings.

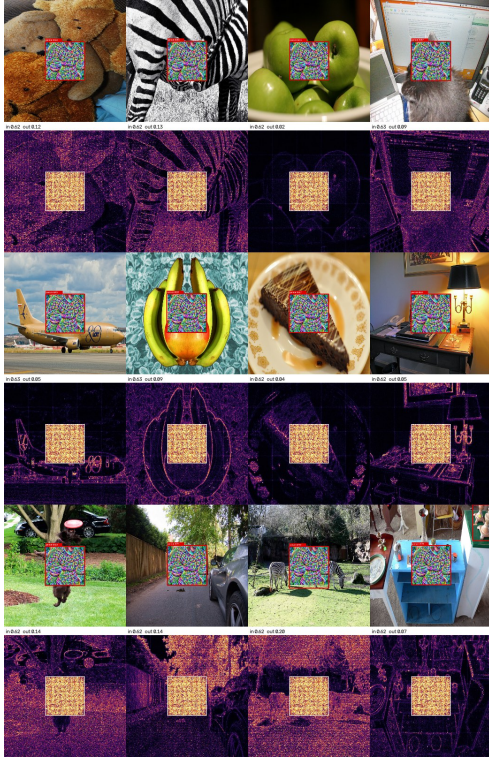


Fig. 46. Twelve more frames under the universal patch, with the residual inside and outside the patch printed on each. The ratio is never below three.

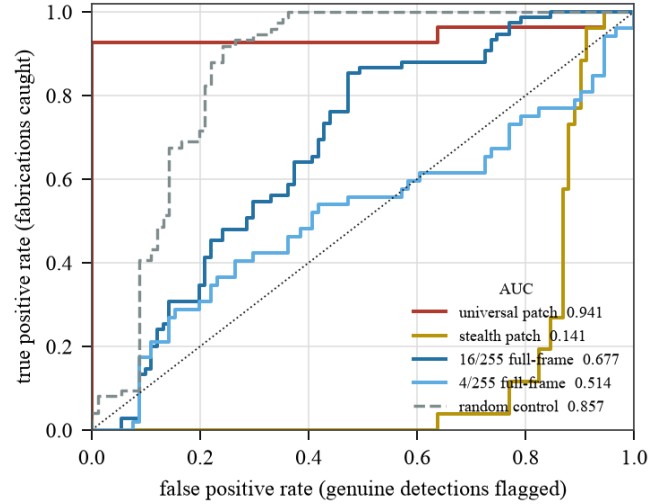


Fig. 47. Receiver operating characteristic per family. The universal patch is separated almost perfectly. The 16/255 field and the random control leave the diagonal in opposite directions.

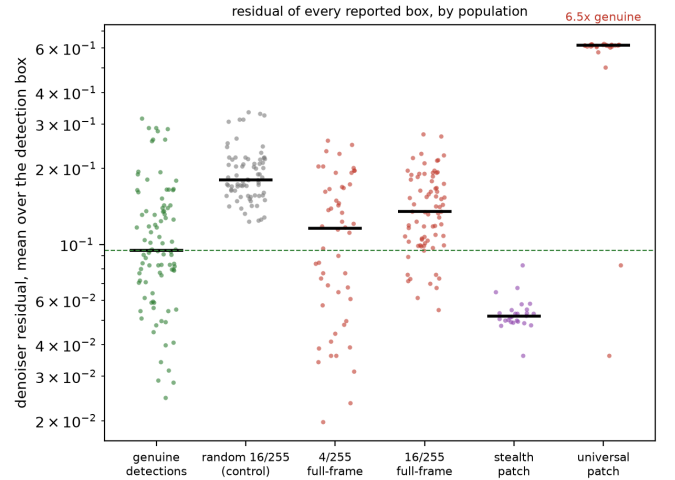


Fig. 48. Every detection, one point per reported box, with bars at the population median. Fabrications from the universal patch sit far above the genuine population.

Effect. The universal adversarial patch is separated at 0.941 [0.858, 1.000]. Its median residual is 0.616 against 0.095 on genuine detections, a factor of 6.5. It is localised, with a higher residual inside the patch than outside, on 1,000 of frames. Read two-sided, the AUC is 0.961. At four and six per cent of the frame, the areas at which the attack of VI-C reaches 0.897 and 0.974, the separation is 0.961 [0.885, 1.000] and 0.964, localised on every frame. Every fabricated patch box is larger than the median genuine box; against genuine boxes of that size class alone the ten-per-cent patch scores 0.947. The resize-trained baseline reaches 0.958 on the same frames after eight epochs.

Effect. A texture statistic does most of this. The same boxes scored by the mean absolute neighbour difference of their pixels, a zero-parameter total variation, give AUC 0.858 [0.741, 0.958] on the ten-per-cent patch and 0.959 at four per

cent; the variance of a Laplacian gives 0.643 and 0.928, the high-frequency share of the spectrum 0.686 and 0.945. The universal patch is a texture anomaly, and a texture statistic catches it. The denoiser adds eight points at ten per cent and nothing at four, and it fails on exactly the families the texture statistics fail on: total variation scores 0.555 on the 16/255 field, 0.473 on the 4/255 field and 0.072 on the stealth patch. What the denoiser is, that a texture statistic is not, is a calibrated model of photographs with a measured resolution floor (VIII-C) and a two-sided reading (VIII-D); what it is not is a necessity for this family.

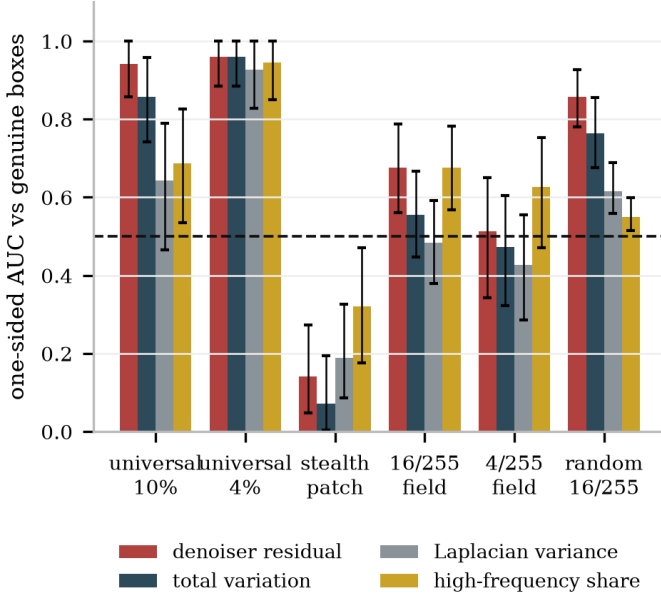


Fig. 49. The detector test scored four ways on the same boxes and frames: the denoiser residual and three zero-parameter texture statistics inside the reported box, total variation, Laplacian variance and high-frequency power fraction. Bars are one-sided AUC against genuine boxes, whiskers the 95% frame-bootstrap interval, one group per family; the dashed line is chance. The universal patch is separated by every statistic; the bounded full-frame fields and the stealth patch by none.

Effect. An operating point. A threshold on the residual set so that five per cent of genuine boxes are flagged (twelve per cent of clean frames carry at least one flagged box) catches 0.926 of the ten-per-cent patch boxes, 0.958 at four per cent and 0.962 at six, and 0.013 of the 16/255 field, none of the 4/255 field and none of the stealth patch. At one per cent of genuine boxes flagged (four per cent of clean frames) the patch rates are unchanged.

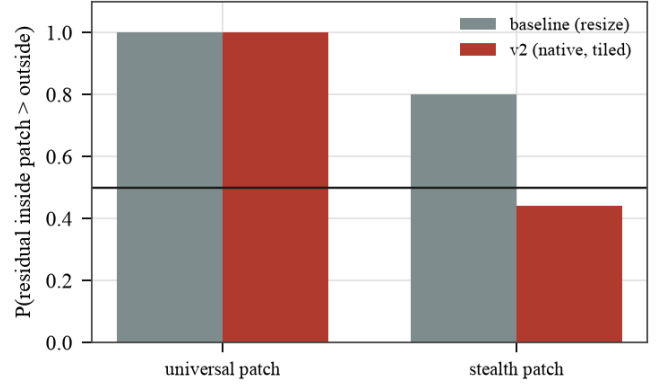


Fig. 50. Where the flagged pixels are. Under the universal patch the residual is higher inside the patch than outside on every frame, under both models.

Consequence. Section VII explains why nothing read from a detector holds this line. Measured, the detector's own output statistic caught 0.000 of this family at its zero-false-positive threshold (20 frames); the best reading of its forward pass, the flip gate, reached 0.713 on 305 boxes of 292 held frames before an adaptive attacker broke the gate on the other families; the second detector of Section IV-B, Faster R-CNN with 41.76 million parameters, reached 0.938 on 30 frames. The result here comes from a network of 1.01 million parameters, forty-one times smaller than that second detector.

F. Full-frame perturbations read as more natural than noise

Method. The 16/255 image-specific perturbation and the matched random control at the same bound, on the same frames, under the same readout. We score each against genuine detections and compare the two with each other.

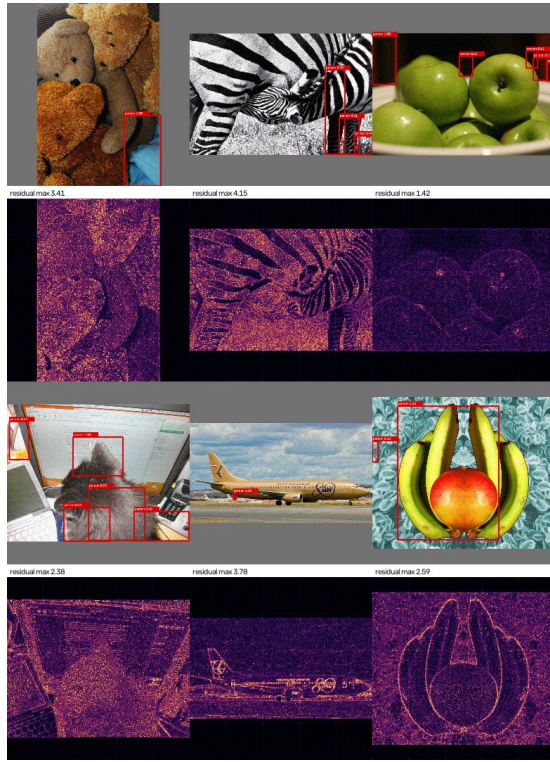


Fig. 51. Six frames under the 16/255 adversarial perturbation, each with its residual map. The map shows that the frame has been touched. It shows nothing localised.

	universal patch	stealth patch	16/255 full-frame	4/255 full-frame	random CONTROL
v2 one-sided	0.94	0.14	0.68	0.51	0.86
v2 two-sided	0.96	0.58	0.56	0.63	0.83
baseline one-sided	0.96	0.39	0.35	0.47	0.50
baseline two-sided	0.98	0.27	0.52	0.58	0.50

Fig. 52. The detector test as one grid. Each cell is the directed AUC: the chance that a box the attack fabricated scores higher than a genuine box. Green near 1.0 means caught; yellow near 0.5 is a coin toss; red below 0.5 means the fabricated box looks more like a photograph than a real one does. Rows: the deployed network (v2) and the resize-trained baseline, each read one-sided (a higher residual is suspicious) and two-sided (any departure from the genuine median is). Columns: the four attack families and, set apart, the random-noise control at 16/255. Read across a row: the universal patch is caught by every reading, the stealth patch is red or yellow for all of them, the full-frame perturbations sit near chance, and the control column is what plain noise does to each reading. 91 genuine and 180 fabricated detections on 25 tuning-split frames, the denoiser having never seen any of them.

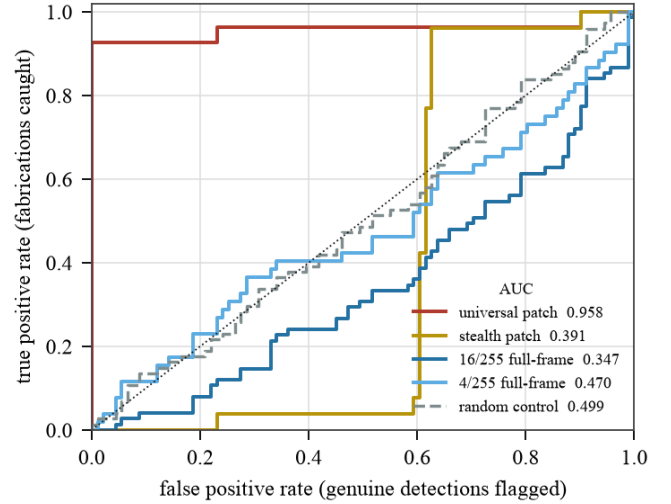


Fig. 53. Receiver operating characteristic per family under the resize-trained baseline, on the same frames and boxes as Fig. 47. The universal patch is separated as before. The random control now sits on the diagonal: at this model's higher floor, 16/255 noise is below what it resolves, and the adversarial field at the same bound falls below chance.

Effect. Random noise at 16/255 raises the residual on a genuine detection to AUC 0.857: the network sees that the frame carries non-photographic texture at a level no camera produces. The adversarial perturbation at the same bound scores 0.677 [0.560, 0.787], **lower than the random control by 0.18** (control interval [0.780, 0.928]; the two intervals touch, so the gap is suggestive at this n and is confirmed by the same ordering at every one of the eight probes). Under the resize-trained baseline the same field scores 0.347 against a control at 0.499: below chance, in the same direction. Fig. 42 shows the adversarial field below the control at every one of the eight probes. The 4/255 family scores 0.514: chance, below the floor.

Consequence. An adversarial perturbation is not white noise: the denoiser finds it more photograph-like than random noise of the same size, a real, ordered, two-sided signal. But it separates the perturbation from noise, not from a clean frame. A deployment that thresholds the residual alone flags a noisy camera before it flags this attack. The readout this family would call for, the residual's slope across probes rather than its mean, is not implemented here (IX-B).

G. The stealth patch is a photograph

Method. The stealth patch of VI-E: a crop of a real photograph carrying an 8/255 optimised delta, the digital form of the patches built to look photographic [10]. We measure the texture of the patch region and of the scene it replaces, as total variation (mean absolute neighbour difference per pixel per channel, 0–255 units), and the residual inside and outside the patch, and do the same for the universal patch and for the anchor crop pasted with no delta on the same frames. Patches are stored at 128×128 and deployed at 202×202 , a $1.58 \times$ bilinear upsample that smooths them, so the texture on the

plate is for the patch as deployed, not for the file measured in VI-E. The stealth patch was built six days before any part of this defence existed.

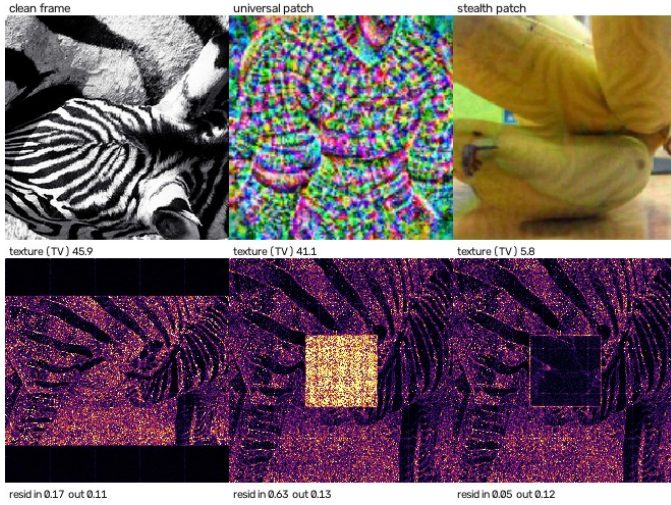


Fig. 54. One region under three conditions, the clean frame, the universal patch and the stealth patch, each with its texture and the residual it produces. The universal patch raises the residual inside the region four-fold and leaves the surroundings unchanged. The stealth patch leaves it below its surroundings.

Effect. The stealth patch scores 0.141 [0.048, 0.272], well below chance, and is localised on 0.440 of frames [0.280, 0.640], which is chance. Fig. 54 prints the texture and the residual for one region under three conditions. Across the 25 frames as deployed, the universal patch region has a total variation of 29.3, the stealth patch 3.4 and the anchor crop with no delta 2.4; the median residual inside the region against outside is 0.049 against 0.052 for the stealth patch and 0.036 against 0.052 for the unmodified crop. The region reads as a photograph because it is one: a crop upsampled 1.58 times is smoother than the scene it replaces, its residual sits below its surroundings with or without the delta, and the 8/255 delta adds a little texture without changing that. The detections it fabricates score *lower* than genuine ones.

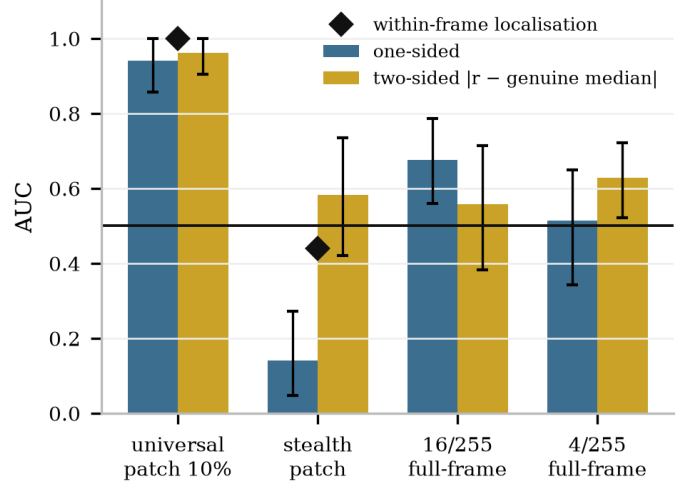


Fig. 55. One-sided against two-sided readout, deployed network, with the 95% frame-bootstrap interval on each bar. Scoring [residual – genuine median] costs nothing on the universal patch and does not recover the stealth patch: its two-sided AUC is 0.582 [0.420, 0.735], its departure from the genuine median no larger than a genuine box’s own, while one-sided it sits below genuine boxes. Diamonds: within-frame localisation.

Consequence. This is a structural limit, not a tuning failure. An oracle asked whether a region is a photograph answers yes for an attack built from a photograph. No threshold, training budget or architecture inside this design changes that answer. Catching this family needs a different question, whether the content is plausible where it is, and that is a different oracle.

H. Cost

Each probe costs 25 forward passes of the 1.01-million-parameter denoiser (VIII-A) on 128×128 tiles, and the deployed readout uses eight probes, 200 passes per frame. The detector itself has 2.41 million parameters in its fused graph and runs once at 640×640. Latency was not measured; the count is a size estimate, not a timing. Fig. 42 shows the universal-patch separation equal at every probe, so a deployment could use fewer, but that reduction is not made or measured here. Training is two epochs, roughly ninety minutes on one Apple GPU; Fig. 34 shows why more is not needed.

IX. DISCUSSION

What this section shows. The two halves of the paper, read together. Part A found that controlling this detector is cheap for anyone holding the model.

Part B found that no check built from the detector’s own computation holds against that adversary, for a reason that applies to any trainable model, and that one check built on something else, whether the pixels look like a photograph, separates the reusable patch, fails on the bounded perturbations and is passed by the photographic patch. The section states what that defence is and is not: a measured result against attackers unaware of it, not a guarantee; a result a plain texture statistic nearly matches; and a model that will itself be

attacked once its weights are known, which is why it is built to be replaced.

A. Summary of measured exposure

Controlling YOLO26-nano is inexpensive, and the margin is wide. Whole-image suppression costs a change below one grey level per pixel and survives lossless delivery as a file; it does not survive JPEG at the qualities tested, whereas the universal patch does. Integrity is reachable throughout and consistently dearer: injecting a chosen class at a chosen place costs roughly an order of magnitude more budget than removing everything, and displacing a box while holding its label and confidence sits between the two. Confidentiality was not evaluated.

B. Limitations and open questions

The following are absent from this work, and their absence is not a result.

- **Transfer.** Every attack assumes the adversary holds this exact checkpoint. Whether they carry to a model trained privately and never published [18] is unmeasured, and it is the assumption on which the practical reach of all these results depends. Only YOLO26-nano was attacked; the argument of Section VII is general, the measurements are not.
- **Operating point and input size.** Every attack is priced at confidence 0.25 and 640 pixels, the library's defaults. A stricter or looser confidence threshold, or a larger input, changes the budget a suppression needs; neither was varied.
- **Delivery.** JPEG survival of the full-frame attack was measured at two budgets and three qualities, one seed; the 8-bit rounding of a clean frame was not scored as a control.
- **Few-pixel perturbations.** We change every pixel slightly, or one rectangle heavily. How few individual pixels suffice is unasked.
- **Query-only attacks.** Every attack here uses gradients, which require the model internals. What remains available to an adversary who can only submit images and read answers [17], at a matched forward-pass budget, is unmeasured.
- **For patches: class, placement, transfer and the physical world.** Every patch targets the person class. Patches are pasted at the centre of the frame, which is the most forgiving position available; a corner or an edge is untested. Whether a patch built here works against another checkpoint is untested. The delivery ladder used one patch per arm on 15 photographs. And nothing was printed or photographed, so no statement in this paper bears on a physical patch.
- **For the box attack.** Displacement is reported in pixels, not as a fraction of box size, and success is not stratified by object size.
- **For the defence.** One denoiser seed and one training recipe were deployed; the baseline differs from it in two variables at once. The detector test used three patch areas, one target class and 25 frames. The denoiser was trained and tested on

COCO only. The residual's slope across probes, the readout the full-frame family would call for, was not implemented. Latency was not measured. And the attacker who knows the denoiser is there and puts it in the loss was not run, which is the subject of the next subsection.

C. What the defence is, and is not

Three things bound it, and the first is a limit, not a shortfall. The photographic patch of VI-E defeats the denoiser inside the stated threat model and with no knowledge of it: a region cut from a photograph is a photograph, its residual sits below its surroundings with or without the optimised change, and no threshold, training budget or architecture inside this design changes that, because the question is answered correctly. Second, on the family the denoiser does separate, the universal patch, a zero-parameter texture statistic inside the box separates it nearly as well: the patch is a texture anomaly, and the denoiser's contribution over that baseline is eight points at ten per cent of the frame and nothing at four. What the denoiser is, that the statistic is not, is a calibrated model of photographs with a measured resolution floor; what it is not is a necessity for that family. Third, the two bounded full-frame perturbations are not separated from genuine detections at all.

It is not a proof, and by the standard this paper applies to every other defence it is evaluated at the weaker level: every number in Section VIII is against attacks built without the denoiser in the loss. It has not been tested against an attacker holding its weights, nor against one who trains a substitute from the public recipe and transfers. Like any model it is a defence until those weights are exfiltrated or the network is reverse-engineered; at that point Section VII applies to it. The denoiser is differentiable: if it can be trained it can be attacked, and it will be. The adaptive attack, with the residual added to the fabrication loss, is the next measurement, and the one that decides what the defence is worth.

X. CONCLUSION

Under white-box access, attacking an object detector reduces to gradient descent on the input tensor: the procedure that trained the model, pointed at the picture.

Applied to YOLO26-nano it prices every objective an operator cares about, and the prices are low. A change of one part in 255 in every pixel empties the frame on most photographs and survives lossless writing to disk, though not JPEG; a chosen class appears at a chosen position for a few parts in 255 more; one patch file, optimised once, invents a person on photographs it has never seen and survives compression; a crop of a real photograph carrying a small change does the same while remaining a photograph. The attacker needs the published checkpoint, the target photograph for the per-photograph attacks, and nothing else.

Detection from inside the detector fails, and the failure is structural. Every statistic of the forward pass is a differentiable

function of the pixels the attacker writes, or has a smooth surrogate; to a gradient-aware adversary it is one more minimisation term, satisfied by a few more steps of the same descent. Stacking such terms does not multiply the constraint, because views of one image are one constraint. A second model is the same term under another name and falls to transferability, leaving a computational tax rather than a barrier. The certified patch form cannot distinguish an object that disappears under a mask from a fabrication that does. The single test that holds reads the attack's own objective, so evasion is failure; it covers one attack and costs one comparison. The argument is not specific to YOLO26, only the measurements are: it is what white-box means, and it is why a decade of detection-based defences on classifiers fell to the same procedure.

Verification therefore requires a quantity outside the detector's computation. Denoising score matching supplies one: a network trained only on clean photographs, by learning to reverse Brownian noise, measures how far the pixels under a reported box lie from the photographic distribution. The reusable patch, which is not a photograph, is separated from genuine detections and localised, and a zero-parameter texture statistic separates it nearly as well; the bounded full-frame perturbations are not separated; the photographic patch is passed, and no training changes that, because the question is answered correctly. The readout is 200 passes of a one-million-parameter network per frame; its latency was not measured.

The defence is bounded by the argument that produced it. The denoiser is a differentiable network; once its weights are exfiltrated, Section VII applies to it and it becomes one more term in the loss. Every number here is against attacks that did not know it existed. It trains in under two hours, so a compromised checkpoint can be replaced by an independently trained one; whether an attacker's own independently trained denoiser transfers to the operator's, and so how much replacement buys, is the measurement that has not been made. The next measurements are that one and the attacker who puts the residual in the loss. Until they are taken, the claims are these: the attack surface is priced, the class of defences that cannot hold is characterised, and the one that separates a patch is built, matched by a texture statistic, and bounded. An operator should run the quiet-output test today, which covers total suppression and nothing else; treat a frame-reading defence as a cost to the attacker that must be measured per defence, as it was here for one; and deploy the denoiser, or the texture statistic it barely beats, knowing what each is.

Disclosure. The target is a public checkpoint and no deployed system was attacked. No vendor was notified, because the findings describe properties of white-box access to any public detector rather than a defect specific to the vendor's code; the suppression-free head is attacked and defended on the same terms as the default one. Patch files, the per-photograph tool and attacked images are not released;

every result file carries the hash of the code that produced it. The author declares no competing interests and received no funding for this work.

REFERENCES

- [1] R. Sapkota, R. H. Cheppally, A. Sharda, and M. Karkee, "YOLO26: Key architectural enhancements and performance benchmarking for real-time object detection," arXiv:2509.25164, 2025.
- [2] C. Szegedy et al., "Intriguing properties of neural networks," in Proc. ICLR, 2014.
- [3] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in Proc. ICLR, 2015.
- [4] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in Proc. ICLR, 2018.
- [5] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 1369–1378, doi: 10.1109/ICCV.2017.153.
- [6] K.-H. Chow, L. Liu, M. E. Gursoy, S. Truex, W. Wei, and Y. Wu, "TOG: Targeted adversarial objectness gradient attacks on real-time object detection systems," arXiv:2004.04320, 2020.
- [7] D. Wang et al., "Daedalus: Breaking nonmaximum suppression in object detection via adversarial examples," IEEE Trans. Cybern., vol. 52, no. 8, pp. 7427–7440, Aug. 2022, doi: 10.1109/TCYB.2020.3041481.
- [8] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," arXiv:1712.09665, 2017.
- [9] S. Thys, W. Van Ranst, and T. Goedemé, "Fooling automated surveillance cameras: Adversarial patches to attack person detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), 2019, pp. 49–55.
- [10] Y.-C.-T. Hu, B.-H. Kung, D. S. Tan, J.-C. Chen, K.-L. Hua, and W.-H. Cheng, "Naturalistic physical adversarial patch for object detectors," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 2021, pp. 7848–7857.
- [11] C. Xiang, S. Mahloujifar, and P. Mittal, "PatchCleanser: Certifiably robust defense against adversarial patches for any image classifier," in Proc. 31st USENIX Secur. Symp., 2022, pp. 2065–2082.
- [12] C. Xiang and P. Mittal, "DetectorGuard: Provably securing object detectors against localized patch hiding attacks," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS), 2021, pp. 3177–3196, doi: 10.1145/3460120.3484757.
- [13] H. Zhang and J. Wang, "Towards adversarially robust object detection," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 2019, pp. 421–430.
- [14] P.-C. Chen, B.-H. Kung, and J.-C. Chen, "Class-aware robust adversarial training for object detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2021, pp. 10420–10429.
- [15] J. Wyatt, A. Leach, S. M. Schmon, and C. G. Willcocks, "AnoDDPM: Anomaly detection with denoising diffusion probabilistic models using simplex noise," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), 2022, pp. 649–655.
- [16] M. S. Graham, W. H. L. Pinaya, P.-D. Tudosi, P. Nachev, S. Ourselin, and M. J. Cardoso, "Denoising diffusion models for out-of-distribution detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), 2023, pp. 2948–2957.
- [17] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square Attack: A query-efficient black-box adversarial attack via random search," in Proc. Eur. Conf. Comput. Vis. (ECCV), 2020, pp. 484–501.
- [18] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples," arXiv:1605.07277, 2016.

- [19] E. B. Wilson, “Probable inference, the law of succession, and statistical inference,” *J. Amer. Statist. Assoc.*, vol. 22, no. 158, pp. 209–212, 1927.
- [20] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, “Synthesizing robust adversarial examples,” in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, PMLR vol. 80, 2018, pp. 284–293.
- [21] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 1765–1773.
- [22] N. Carlini et al., “On evaluating adversarial robustness,” *arXiv:1902.06705*, 2019.
- [23] F. Tramèr, N. Carlini, W. Brendel, and A. Madry, “On adaptive attacks to adversarial example defenses,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 1633–1645.
- [24] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,” in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, vol. 80, 2018, pp. 274–283.
- [25] N. Carlini and D. Wagner, “Adversarial examples are not easily detected: Bypassing ten detection methods,” in *Proc. 10th ACM Workshop Artif. Intell. Secur. (AISec)*, 2017, pp. 3–14, doi: 10.1145/3128572.3140444.
- [26] T.-Y. Lin et al., “Microsoft COCO: Common objects in context,” in *Proc. ECCV*, 2014, pp. 740–755.
- [27] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 6840–6851.
- [28] P. Vincent, “A connection between score matching and denoising autoencoders,” *Neural Comput.*, vol. 23, no. 7, pp. 1661–1674, 2011.
- [29] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, “Score-based generative modeling through stochastic differential equations,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.
- [30] W. Nie, B. Guo, Y. Huang, C. Xiao, A. Vahdat, and A. Anandkumar, “Diffusion models for adversarial purification,” in *Proc. 39th Int. Conf. Mach. Learn. (ICML)*, PMLR vol. 162, 2022, pp. 16805–16827.