SILENT LOSS OF IMAGE CONDITIONING UNDER DIRECT FOOLING OBJECTIVES: A DIAGNOSTIC FOR AMORTIZED ADVERSARIAL ATTACKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Amortized adversarial attacks initially train a model to generate images that can fool a victim classifier, without requiring access to it at attack time. Previous works mention that models trained with an attack objective may collapse and generate similar perturbations across all images, while maintaining high attack success rate and perceptual quality. Our experiments reveal a novel form of this failure where the perturbation remains diverse unlike the earlier failure mode, yet when the generated perturbations are applied to an image other than the one it was generated from, the attack success rate still remains quite high. We introduce a shuffled-image test that compares a perturbation paired with the source image and a non-source image to measure this failure. We also evaluate a partial solution to this problem by guiding the generator with an attacker during an initial warmup phase. This strategy successfully preserves source-image specificity during the warmup phase, keeping perturbations image-specific. However, this strategy cannot consistently keep specificity high after warmup ends and the attack objective starts. Only eight of twelve full-recipe runs kept specificity high to the end. Finally, we test 57 third-party public amortized attack model checkpoints and find high specificity among them, which points to a recipe-specific finding within that one lineage rather than a general defect of amortization.

1 INTRODUCTION

Iterative adversarial attacks such as PGD (Madry et al., 2018) and Carlini-Wagner-style margin optimization (Carlini & Wagner, 2017) need multiple passes to the victim on every image they perturb, which becomes slow and impractical when such data must be produced in bulk, especially for adversarial training. Amortized attacks train a generator first, and then attack new images without needing any victim classifier access, moving computational cost from attack time to training time. GAP (Poursaeed et al., 2018), ATN (Baluja & Fischer, 2018), and AdvGAN (Xiao et al., 2018) are examples of this approach.

In our amortized training recipe, we first guide our 2–4 step generator with *AdvAD* (Li et al., 2024), a non-parametric diffusion-guided iterative attack, during a warmup phase before using any attack objective. The role of the teacher here is not to induce an attack. Our teacher-only ablation, with no attack objective, reaches only 1.46% attack success (`imitation_only`). The student's output is a perturbation confined to a fixed L_∞ norm (Section 3), the same bounded task PGD, APGD and Square attack solve, under different classifier access assumptions.

Our central finding is that amortized training can cause source-image specificity loss in two forms. The first is near-universal collapse onto a single shared perturbation, a failure NAT (Nakka & Alahi, 2025) describes qualitatively from visual inspection. The second, which we report here, leaves the perturbations diverse and nonetheless generic, so that a perturbation applied to an image other than the one it was generated from still remains highly likely to cause an attack. Attack success rate, perceptual quality and the step curve across all steps remain consistent, and universal energy, which measures how much perturbation energy lies in one shared component, stays low in the second form as the perturbation is diverse.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We also study a potential solution, in which guiding the generator with an attack framework like AdvAD keeps the image source-specific during warmup and, in many cases, beyond. We hypothesize that the teacher's role is instead to shift the initial local optimum so that optimization reaches a different basin, one that achieves source-image dependence. What we measure is consistent with this *basin-selector* role. We disable the attack objective during warmup to isolate this effect, varying only whether each image is paired with its own teacher trajectory or another image's, and it replicates at two training seeds. The mitigation is partial rather than reliable, as once warmup ends and the attack objective engages, specificity remains satisfactory in only eight of twelve full-recipe runs.

We evaluate on the complete ImageNet-1k validation set at 224×224 across five committed-in-advance training conditions, four of them at two or more seeds, against iterative and black-box baselines run at the same $8/255$ budget, and release the artifacts from which every guarded number can be recomputed. Two further measurements bound the failure rather than demonstrate it. Reproducing it on GAP's published generator, at three seeds per arm, shows it is not confined to our network. Auditing 57 released third-party checkpoints, which share one architectural lineage and were all trained with feature-space transfer losses rather than a direct fooling objective, finds no corresponding degradation, so the failure is not a general defect of amortized attacks. Neither measurement establishes which component of the recipe is responsible.

We make three contributions.

(1) We propose two forms of silent conditioning loss and a diagnostic that detects both.

(2) We identify that correct teacher pairing suppresses universal energy in warmup.

(3) We evaluate the teacher guided warmup as a partial mitigation.

2 Related Work

Iterative gradient-based attacks. FGSM (Goodfellow et al., 2015), PGD (Madry et al., 2018), and Carlini-Wagner-style optimization (Carlini & Wagner, 2017) are standard L_p-bounded baseline attacks. FGSM uses one gradient step, while PGD and CW use repeated gradient access at attack time.

Amortized and generative attacks. Amortized adversarial attacks such as ATN (Baluja & Fischer, 2018), AdvGAN (Xiao et al., 2018), and GAP (Poursaeed et al., 2018) train models to generate attacks at inference time, amortizing away the classifier access during inference.

For text attacks, White-to-Black (Gil et al., 2019) distills HotFlip attack into a BiLSTM-based character-edit predictor. CDA (Naseer et al., 2019), TTP (Naseer et al., 2021), SCGA (Jeong et al., 2026), dSVA (Wu et al., 2025), NAT (Nakka & Alahi, 2025), and PGA (Peng et al., 2026) extend this approach for better black-box transfer. SCGA stabilizes its generator with an averaged teacher model. Particularly, PGA uses a training schedule to prevent the generator from getting stuck in poor local solutions, and NAT reports the near-universal collapse qualitatively, but does not supply a functional diagnostic. Jang et al. (2019) *generate* diverse attacks for adversarial training. Our focus is how a diverse attack can still collapse into a generic one. Appendix H compares all six against lost source-image dependence.

Our `attack_only` ablation replicates the teacher-free GAP/ATN approach on our network. Holding the architecture fixed varies teacher losses and the attack objective together, so it tests a group of training recipes rather than teacher guidance alone. Appendix G runs the reverse experiment on GAP's published generator architecture, and under this teacher-free recipe, it fails image specificity consistently, unlike our teacher-guided recipe. This establishes that the failure appears on another architecture, not that the objective alone causes the outcome.

Diffusion-based adversarial generation. AdvDiff (Dai et al., 2024) and related methods use a diffusion model to generate unrestricted adversarial images, trading strong visual quality for high computational cost of multi-step denoising at attack time. Our teacher, AdvAD, is itself a diffusion-guided *attack*. AdvAD and the U-Net student are existing components, and our contribution is the diagnostic and empirical analysis, not a new teacher or generator architecture.

Robustness evaluation and AutoAttack. AutoAttack (Croce & Hein, 2020) establishes standard parameter-free reference points and RobustBench (Croce et al., 2021) publishes pretrained

defended classifiers for robustness evaluation. We use a RobustBench checkpoint as the teacher's adversarially-trained surrogate, and AutoAttack's APGD components as one of our baselines, run at a common budget.

3 THREAT MODEL

We attack a real source image x under an L_∞ budget ϵ in the untargeted setting. The objective is to produce x' with $\|x' - x\|_\infty \leq \epsilon$ such that a target classifier f satisfies $f(x') \neq f(x)$, without access to f's gradients or outputs at attack time. Our generator takes the dataset's ground-truth class label as an input, assuming it is known to the attacker. Training uses white-box surrogate classifiers, and the classifier we report attack success against is one of them, so the reported setting is white-box at training time followed by query-free inference. We do not claim attack success against unseen deployed victims, and transfer to classifiers outside the training ensemble is neither established nor optimized for here. PGD and APGD instead require live gradient access, and Square requires live score queries. The comparison shares a bounded, untargeted source-image task and one $8/255$ budget, not identical access rules. Attack success rate is considered on clean images x that are classified as being *correct* by the classifier, and every comparison is measured against that same set of clean images.

The student meets this budget by construction. It keeps a running perturbation $u^{(k)}$ that is updated at every step and only bounded through $\tanh$,

$$u^{(k)} = \begin{cases} 0, & k = 0, \\ u^{(k-1)} + G_\theta(x, z^{(k-1)}, y, s_k, \epsilon), & k \geq 1, \end{cases} \qquad z^{(k)} = \mathrm{clip}(x + r \tanh(u^{(k)}), -1, 1), \quad (1)$$

where G_θ is the student network with parameters θ, $s_k = (k-1)/\max(K-1, 1)$ is the normalized step index, and r is ϵ in the model's coordinate system. The final state $x' = z^{(K)}$ is the adversarial image. $|\tanh(\cdot)| \leq 1$, $\|z^{(k)} - x\|_\infty \leq r$ before clipping, and clipping onto the image cube is a projection onto a convex set containing x. The budget therefore holds at every step k, independently of the learned parameters θ. Floating-point budget checks use a small numerical tolerance.

We prefer the $\tanh$ parameterization rather than step-wise projection because it stays differentiable where a projection's gradient vanishes after saturation, and because it separates the budget from the step count.

4 METHOD

4.1 TEACHER AND STUDENT

The teacher is an AdvAD-style (Li et al., 2024) non-parametric, diffusion-guided attack. Appendix L discloses two implementation differences. We choose AdvAD because it supplies attack trajectories and intermediate states without training a separate generative teacher. We do not compare teacher families, so the results do not establish that AdvAD is preferable to PGD or another iterative teacher. Starting from the clean image, it takes $T = 50$ guided denoising steps, computing a normalized-gradient update against an ensemble of surrogate classifiers. It freezes an image once every ensemble member that was correct on its clean input is fooled. Adaptive per-image weights put more importance to the surrogate currently hardest to fool. The teacher is pretrained and kept frozen during student training.

The student, *TurboStudent*, is a compact three-scale U-Net (Ronneberger et al., 2015) with a global attention bottleneck, applied $K \in \{1, 2, 4\}$ times as in Equation 1. At each of the K steps it takes the clean image, the current perturbed state, the true class label, a normalized step index, and ϵ as input, and outputs an increment to an accumulated, $\tanh$-bounded perturbation. Training guides intermediate states towards both the next teacher step as well as the final output, so one- and two-step inference modes are not merely truncations of the four-step network. Appendix C shows the workflow and both generator architectures.

4.2 LOSS COMPOSITION

The student is trained to optimize the weighted sum of seven terms

$$\mathcal{L} = \lambda_{\text{atk}}\mathcal{L}_{\text{atk}} + \lambda_{\text{dir}}\mathcal{L}_{\text{dir}} + \lambda_{\text{mag}}\mathcal{L}_{\text{mag}} + \lambda_{\text{anc}}\mathcal{L}_{\text{anc}} + \lambda_{\text{spec}}\mathcal{L}_{\text{spec}} + \lambda_{\text{mdr}}\mathcal{L}_{\text{mdr}} + \lambda_{\text{perc}}\mathcal{L}_{\text{perc}}.$$

The *attack* loss $\mathcal{L}_{\text{atk}}$ lowers the surrogate's margin between the true class and the best competing class, which is what makes the output adversarial. Three *teacher-guidance* losses act only while the teacher is active. The direction loss $\mathcal{L}_{\text{dir}}$ matches the direction of the teacher's final perturbation, the magnitude loss $\mathcal{L}_{\text{mag}}$ matches how much of the budget the teacher uses on each image, measured as the average absolute size of its perturbation, and the anchor loss $\mathcal{L}_{\text{anc}}$ pulls each intermediate step toward the corresponding teacher state. Two *anti-collapse* losses are off during warmup and ramp in together with the attack loss. The specificity hinge $\mathcal{L}_{\text{spec}}$ requires an image's own perturbation to lower that image's margin by a fixed amount more than another image's perturbation does, and the universal-energy penalty $\mathcal{L}_{\text{mdr}}$ charges the fraction of perturbation energy carried by the batch mean. The *perceptual* penalty $\mathcal{L}_{\text{perc}}$ charges mean LPIPS (Zhang et al., 2018) above its target.

The attack term is zero during warmup, then ramps in. Table 6 gives all weights and schedules, and Appendix A states every term exactly. Appendix E records a consequence of guiding a $\tanh$-bounded output toward a teacher direction, namely that a smooth, gradient-aligned perturbation realizes only about a third of the margin decrease a signed one would at the same radius.

4.3 DIAGNOSTIC SCORES

Let $\delta_i = z_i - x_i$ be the perturbation of image i, where z_i is its final state $z^{(K)}$, let $\sigma(i) \neq i$ index a non-source image, and let $A(\delta)$ be the attack success rate after adding perturbations and clipping. Our specificity score (S) and universal energy (U) are

$$S = \frac{A(\delta)}{A(\delta_\sigma)}, \qquad U = \frac{\|\bar{\delta}\|_2^2}{N^{-1}\sum_i \|\delta_i\|_2^2}, \quad [\bar{\delta} = N^{-1}\sum_i \delta_i]. \tag{2}$$

For the within-class specificity S_{wc}, shuffled images are taken only from the images with the same class as source. Higher specificity means higher source-image dependence. High U indicates a shared component, but low U does not ensure high specificity.

5 EXPERIMENTAL SETUP

Dataset and splits. We evaluate on ImageNet-1k (Deng et al., 2009; Russakovsky et al., 2015) at size 224×224. We draw 20,000 images (20 images per class, balanced) from the official *training* split for teacher guidance, and evaluate on the complete official *validation* split (50 images per class, 50,000 images total). Attack success is reported only over images the target classifier already classifies correctly on the clean input (the "eligible" subset), following standard practice.

Teacher configuration. We run AdvAD at $\epsilon = 8/255$, guided by an ensemble of a standard ResNet-50 (He et al., 2016) and an adversarially-trained ResNet-50 (Engstrom et al., 2019). Appendix F gives the step count, guidance scale and remaining settings.

Student training. Appendix F lists principal hyperparameters. Training used a single RTX 5090 GPU and cost approximately 13.3 GPU-hours of active compute.

Fixed-budget baselines. We compare against PGD (Madry et al., 2018), a Carlini-Wagner-style (Carlini & Wagner, 2017) L_∞ margin attack (not the original L_2 Lagrangian formulation), DeepFool (Moosavi-Dezfooli et al., 2016), AutoAttack (Croce & Hein, 2020) (the two APGD components for efficiency), and Square Attack (Andriushchenko et al., 2020) at four query budgets. Every baseline runs at the same L_∞ budget the student runs at, $\epsilon = 8/255$, and we report achieved LPIPS and PSNR as measured outcomes of that budget. Fixing the radius rather than the achieved quality is what makes the Square rows a query-budget sweep, because calibrating quality per attack gives each row its own radius, and for Square that radius moves with the query count. Appendix B reports what the quality-calibrated alternative gives instead, and why we do not use it.

Table 1: Student attack success rate (ASR), image quality (LPIPS, lower is better, and PSNR in dB, higher is better) and time per image for each number of steps, on the ImageNet-1k validation split against ResNet-50. For 4 steps, the 95% confidence interval of the attack success rate, computed by resampling the 1000 classes, is [0.9719, 0.9786].

Steps	ASR	LPIPS	PSNR	ms/image
1	0.042	0.0064	43.39	2.9
2	0.798	0.0236	37.93	5.9
4	**0.9753**	0.0262	36.80	11.8

Table 2: All attacks use the same L_∞ budget, $\epsilon = 8/255$, on the same 1000 images (one per class). Access shows what each attack needs from the victim at attack time, which is gradients (white-box), output scores (black-box) or nothing. DeepFool stops as soon as an image is misclassified, so its distortion does not grow with ϵ (Appendix B).

Attack	Access	Victim queries/steps	ASR	LPIPS	ms/image
PGD-40	white-box	40 grad. steps	1.000	0.0499	18.6
CW-margin (L_∞)	white-box	40 grad. steps	0.999	0.0357	18.8
AutoAttack (APGD)	white-box	$\sim$100+ grad. steps	0.999	0.0620	52.1
DeepFool	white-box	40 grad. steps	0.335	0.0001	73.7
Square (100q)	black-box	100 queries	0.223	0.0248	19.7
Square (500q)	black-box	500 queries	0.349	0.0382	94.3
Square (1000q)	black-box	1000 queries	0.418	0.0449	177.8
Square (5000q)	black-box	5000 queries	0.486	0.0568	888.1
Ours (4-step)	**none at attack time**	**0**	**0.9710**	**0.0258**	**11.8**

6 RESULTS

The student beats the teacher at the same budget, but not at the same image quality. When both use the same $8/255$ budget, the 4-step student fools about 11 percentage points more images than the teacher, but it changes the image about twice as much by LPIPS. When the teacher is allowed a larger budget, so that its images change as much as the student's, the teacher fools more images than the student and still keeps clearly better PSNR (Appendix D). The student's advantage is speed and access. It is about eight times faster per image than the teacher and needs no access to the victim classifier at attack time. Appendix J estimates how many images must be attacked before this saving pays back the victim access used during training.

Image specificity. On the training images, the student's perturbations fool the classifier several times more often on their own source image than on another image of the same class. This is well above 2.0, the minimum specificity we require for a run to count as healthy, which we call the *floor*. So this student did not collapse into one shared perturbation that works the same on every image, which is the failure we worry about most, even in a recipe that already has two loss terms meant to prevent it.

Reading Table 2. We do not rank the rows, because the attacks need different kinds of access to the victim and are built for different goals. The white-box attacks fool slightly more images than our student. This is expected, because they compute 40 or more gradient steps against the victim for every image, while our student runs four forward passes and never touches the victim. But our student changes the image less than every white-box attack that matches its success. The only white-box attack that changes the image less, DeepFool, fools only about a third as many images. Square Attack is the closest comparison, since like our student it uses no victim gradients. At every query budget it fools about half as many images as our student or fewer, while changing the image about as much or more. This is also expected, because it can only query the victim's outputs.

7 ABLATION AND THE BASIN-SELECTOR HYPOTHESIS

A high attack success rate does not tell us whether a generator is healthy, so we ask what the teacher actually contributes. Our hypothesis is that the teacher is not something for the student to copy, but a *basin selector* during warmup. In other words, it steers the student toward a region of training where the attack objective can later be optimized without the student collapsing onto one perturbation that ignores the source image.

To test this, we compare the reported recipe with four variants. Each variant is identical to the reported recipe except for the change listed.

- **`full`** (reported above) uses the teacher during warmup, then adds the attack objective while the teacher stays active.
- **`attack_only`** removes the teacher, and with it the three losses that come from the teacher (direction, magnitude and anchor). The attack objective and the two anti-collapse losses stay active from the first epoch. This is the usual way amortized attacks are trained, so it also serves as a comparison point for that line of work.
- **`imitation_only`** sets the attack objective to zero for the whole run ($\lambda_{\text{atk}} = 0$), so the student only copies the teacher.
- **`warmup_only`** stops teacher guidance after epoch 31, once the attack objective has reached full strength, instead of keeping the teacher for the whole run as `full` does. It tests whether the teacher is needed throughout training or only at the start.
- **`shuffled_warmup`** is identical to `warmup_only`, except that during warmup each image is paired with the teacher trajectory of a *different*, randomly chosen image. It tests whether the teacher's signal is *specific to each image*, since the mismatched pairing breaks the link between an image and its own teacher trajectory.

7.1 RESULTS

Table 3 scores all five conditions on the same validation split. *Univ.* (universal energy) is the fraction of perturbation energy that lies in the average perturbation shared by all images. *Mean-δ ASR* is the attack success rate when that single average perturbation is added to every image. It shows how well the shared pattern attacks on its own, and it is not an additive decomposition of the original success rate, meaning it does not say how much of that success comes from the shared pattern. Two rows are deliberately not from seed 42, so that the table shows both ways a run can lose image specificity, not only the shared-pattern way. These are `warmup_only` at seed 43, where it degrades, and the teacher-free GAP generator of Appendix G at seed 44. The specificity numbers are not an effect of measuring on images the student was trained on. On 10,000 separate held-out images, every run keeps the same verdict, the healthy runs score slightly *higher*, and the degraded one stays below the floor (Appendix I).

Three runs carry a strong shared perturbation, and two lose specificity while their perturbations stay diverse. In `attack_only`, `shuffled_warmup` and seed-43 `full` (Table 13), the average perturbation alone fools roughly a fifth to a quarter of images, so a large part of these attacks is one *shared* pattern. Seed-43 `warmup_only` fools nearly as many images as the healthy runs, and its perturbations still work on most non-source images, while their shared average fools only a few percent. So the shared pattern is not what makes it fail. The GAP generator is the clearest example of this second form. It attacks about as strongly as seed-43 `warmup_only`, has even less universal energy than `full`, and still falls below the floor. Attack success, image quality and universal energy all look healthy for this run, and only the shuffled-image test reveals the problem.

Copying the teacher alone barely attacks, and mismatched pairing only has a clear effect during warmup. `imitation_only` fools almost no images, so copying the teacher alone produces essentially no attack in this recipe. The final results of `shuffled_warmup` and `attack_only` swap order at a second seed, so we do not rank them (Appendix N). What does repeat across seeds is the effect of pairing during warmup. This matches the simplified argument in Appendix Q, although it does not show that the argument's independence assumption holds for our actual implementation, which clips perturbations to each image.

Table 3: Each row is one training run, for the five training conditions and the two GAP variants (Appendix G). All rows use seed 42 except the two marked, which were chosen to show the second failure form. ASR, LPIPS and PSNR are measured on the full ImageNet-1k validation split. Spec. is *within-class* specificity, the measure we fixed in advance for judging runs. Spec., Univ. and mean-δ ASR are measured on the 20,000 training images. Arrows show which direction is better. Because replication is uneven (12, 3, 2, 2 and 1 runs for the five conditions, three per GAP variant), these are individual runs rather than averages. Four of twelve `full` runs fall below the 2.0 floor, and no ranking between conditions here holds at a second seed (Appendix N).

Condition	Seed	ASR ↑	LPIPS ↓	PSNR ↑	Spec. ↑	Univ. ↓	mean-δ ASR ↓
`full`	42	**0.9753**	0.0262	36.80	**6.49**	0.080	0.026
`warmup_only`	42	**0.9813**	0.0255	32.68	5.11	0.092	0.038
`warmup_only`	**43**	0.9056	0.0265	31.32	**1.28**	0.318	0.061
`attack_only`	42	0.9084	0.0275	32.00	2.44	0.309	0.227
`shuffled_warmup`	42	0.7940	0.0288	31.77	2.31	0.470	0.273
`imitation_only`	42	0.0146	0.0057	46.06	1.69	0.002	0.000
`gap_baseline`	**44**	0.9159	0.0240	32.20	**1.51**	0.039	0.009
`gap_warmup`	42	0.8660	0.0246	33.28	2.33	0.029	0.012

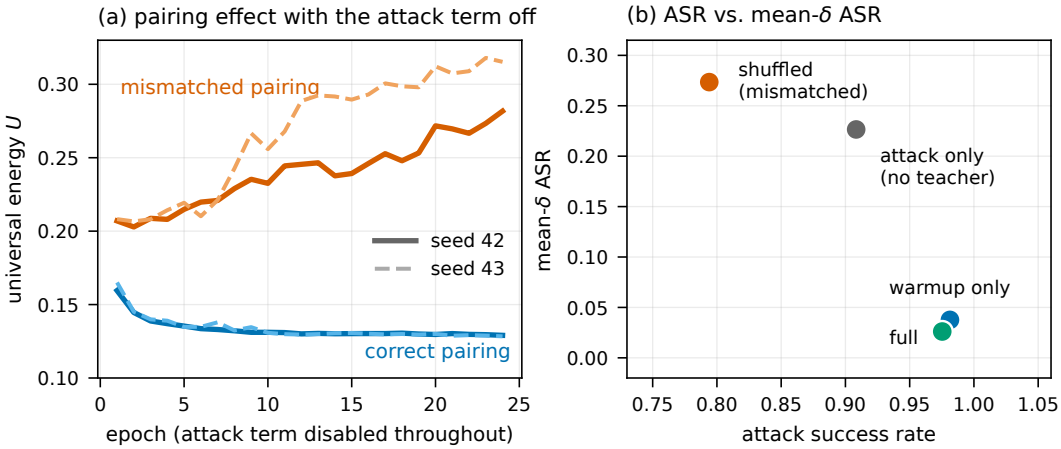


Figure 1: **(a)** Universal energy during warmup at both training seeds (solid lines are seed 42, dashed lines seed 43), with the attack objective switched off. The two conditions differ only in whether each image gets its own teacher trajectory, so the gap between them is caused by the pairing. **(b)** Attack success rates stay close across all four conditions (within 1.3× of each other), while the success of the single average perturbation varies by more than 10×.

Correct and mismatched pairing already differ during warmup, before the attack objective is switched on. In Figure 1(a), during the 24-epoch warmup, universal energy steadily *falls* for the correctly paired runs and *rises* for the mismatched runs, which end more than twice as high. The second training seed, drawn dashed, shows the same direction and a similar gap (exact values in Appendix Q). The attack objective is off during this whole period, so it cannot be the cause. This result also does not depend on what happens after the attack objective starts, a period in which we find that runs behave differently from one another.

The two anti-collapse losses alone do not prevent a shared perturbation. We had expected that removing the teacher would raise attack success at the cost of specificity. That prediction was wrong, and the seed-42 differences behind it do not hold at a second seed (Appendix N). What holds regardless of any ranking is that $\mathcal{L}_{\text{spec}}$ and $\mathcal{L}_{\text{mdr}}$ reach the same weights in `attack_only` as in `full`, yet a large shared pattern remains. Its universal energy is nearly four times that of `full`, and its average perturbation alone fools about a fifth of images. This comparison cannot separate the effect of the teacher losses from the effect of starting the attack objective later.

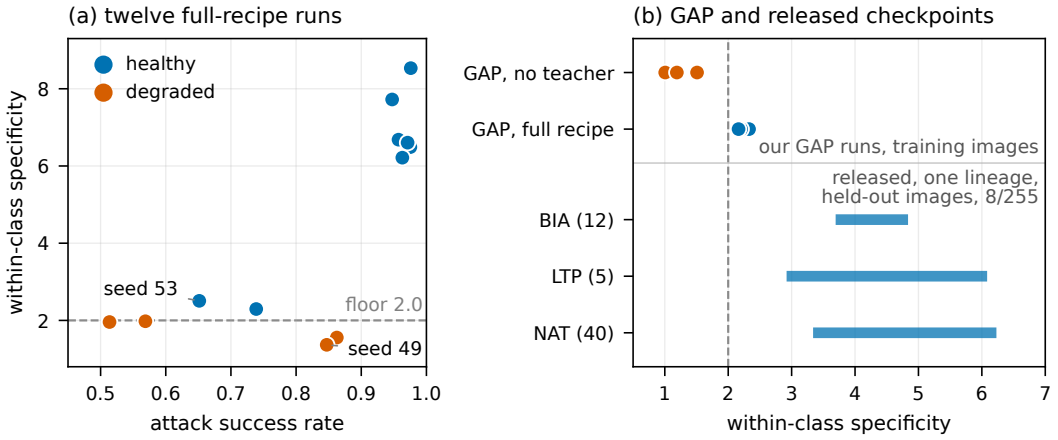


Figure 2: **(a)** All twelve `full` runs. Eight end above the 2.0 floor and four below it, several close to it, and higher attack success does not mean a healthier run (seed 49 against seed 53). **(b)** Within-class specificity of the GAP generator at three seeds per recipe, measured on the training images, and the range of each family of released checkpoints at $8/255$, measured on held-out images. The three families share one generator lineage, so they are not independent samples. Exact values are in Appendices N, G and I.

Only tests that move perturbations to other images tell the conditions apart. `full` and `warmup_only` are the same program until epoch 32 and still end differently, so we report `full` as our method but do not rank the two (Appendix N). The same pattern holds across the whole table, as Figure 1(b) shows. Attack success looks similar everywhere, and only the measures that apply perturbations to other images, either swapped or averaged, can tell a truly per-image attack from a generic perturbation that scores just as well. Appendix P repeats this comparison over every scored run, not just one seed. Appendix K supports it at a smaller scale, where specificity follows how much signal the teacher provides rather than how good an attacker the teacher is.

The failure also appears on GAP's published generator, but not in 57 released checkpoints. On our reimplementation of the GAP generator, training without the teacher falls below the floor at all three seeds, while our full teacher-guided recipe stays above it at all three, though only just (Figure 2b). So the failure is not specific to our network. The seed-44 teacher-free run is the clearest example. It attacks strongly within the quality limit and has less universal energy than our healthiest run, yet it still fools the classifier on most non-source images. This is the second failure form, where universal energy also looks healthy. Separately, none of the 57 released BIA, LTP and NAT checkpoints we measured falls below the floor at either budget, and even the least specific of them is well above it (Figure 2b). Together, these results are consistent with a failure that depends on the training recipe rather than a general defect of amortization, although neither identifies which part of the recipe is responsible.

The warmup effect does not guarantee a healthy final model, and four of twelve full-recipe runs lose specificity anyway. Across twelve `full` runs, three exploratory and nine committed in advance, eight end above the floor and four end below it (Figure 2a). The two groups are not clearly separated, since the weakest healthy runs are much closer to the floor than to the other healthy runs. We report this as a count and not as a rate, because twelve runs of one recipe on one architecture cannot tell us how often the failure happens in general. The count also depends on where the floor is set. A slightly lower floor would halve it, and a moderately higher one would add a fifth run. Attack success does not predict which side a run ends on. Seed 49 attacks far more successfully than seed 53, yet seed 49 fails the floor and seed 53 clears it. So the pairing effect that repeats during warmup does not always survive once the attack objective starts, and the protection is partial in the plain sense that it sometimes fails.

8 Discussion and Limitations

Results vary a lot between runs. Training at a second seed (Appendix N) changed attack success by up to 64 percentage points, and two identical programs drifted 44% apart once the attack objective started. The warmup pairing effect repeats across seeds but the final rankings between conditions do not, so we do not rank the ablation conditions by their final attack success. Headline numbers come from single runs, with confidence intervals computed over classes within each run. Two of the four predictions we made in advance turned out wrong, and we report them as misses. The controlled warmup comparison shows that teacher-image pairing has an effect in this recipe, but the experiments do not establish the causal mechanism behind the loss of specificity after warmup.

What the architecture results do and do not show. The GAP comparison changes the teacher losses and the attack schedule together, so it tests a whole recipe rather than teacher guidance alone. Three seeds per variant on one architecture is also a repeatable count, not a rate. The 57 released checkpoints come from one generator family and were only observed, not tested in a controlled experiment.

Scope. We study untargeted attacks only, and we have not tested targeted teachers. The victim classifier is one of the classifiers our training already uses, and PGD at the same $8/255$ budget still fools more images. Our claim is therefore amortization, meaning no victim access and low cost at attack time, not attack strength. We report no FID score or human study.

What the quality numbers do not show. LPIPS and PSNR measure how much the image changed, but they do not show that people cannot see the change, which we have not tested. The pixel budget also does not formally guarantee preservation of the class a person would see. Magnified difference images show a faint grid pattern, possibly caused by the three upsampling layers in the decoder, although we have not confirmed its cause. A repeating pattern of fixed strength may bother a human viewer more than average-based quality scores suggest. Both quality numbers are computed on the final adversarial images, with this pattern included.

Ethics statement

A fast adversarial attack that needs no access to the victim at attack time makes it cheaper to produce adversarial examples in large numbers, so it can be used both to help and to harm. The same cheap generation that helps red-teaming and robustness testing of deployed classifiers also lowers the cost of misuse against a system an attacker can query during training. Two points from our own results limit this risk. First, we only show attacks on a classifier that is inside the attacker's own training ensemble. We neither establish nor optimize for success against unseen deployed classifiers, so nothing here shows the method reaching a target the attacker could not already train against. Second, training is not cheap (about ten GPU hours per classifier family, Section 5), and building the teacher data needs white-box access to surrogate classifiers. In transfer settings those surrogates need not be the deployed target, so not needing queries at deployment does not remove the risk.

AI use statement

Generative AI tools (coding agents) were used substantially in this work, for implementing and debugging code, refining hypotheses, designing and running experiments, interpreting results, formulating and revising mathematical claims and proofs, checking literature, preparing figures, and drafting this manuscript. Verification includes checks against implementation code, saved artifacts and primary references, together with automated tests and consistency checks. We take responsibility for independently verifying the final content of this work.

Reproducibility statement

Section 5 describes the separate official data splits, and Appendices F and A give the main hyperparameters and the exact losses, with full configurations in the released code. The anonymized supplement contains the code, training logs, evaluation outputs and scripts that regenerate and check the reported numbers. Appendices Q and N show that the warmup effects repeat across seeds while the final rankings do not, and that fixing the settings and seeds does not guarantee the same result.

REFERENCES

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square Attack: A query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision*, 2020.

Shumeet Baluja and Ian Fischer. Learning to attack: Adversarial transformation networks. In *AAAI Conference on Artificial Intelligence*, 2018. doi: 10.1609/aaai.v32i1.11672.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, 2017.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, 2020.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. RobustBench: A standardized adversarial robustness benchmark. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Xuelong Dai, Kaisheng Liang, and Bin Xiao. AdvDiff: Generating unrestricted adversarial examples using diffusion models. In *European Conference on Computer Vision*, 2024. URL https://arxiv.org/abs/2307.12499.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

Logan Engstrom, Andrew Ilyas, Hadi Salman, Shibani Santurkar, and Dimitris Tsipras. Robustness (Python library), 2019. URL https://github.com/MadryLab/robustness.

Yotam Gil, Yoav Chai, Or Gorodissky, and Jonathan Berant. White-to-Black: Efficient distillation of black-box adversarial attacks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1373–1379, 2019. doi: 10.18653/v1/N19-1139. URL https://aclanthology.org/N19-1139/.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Yunseok Jang, Tianchen Zhao, Seunghoon Hong, and Honglak Lee. Adversarial defense via learning to generate diverse attacks. In *IEEE/CVF International Conference on Computer Vision*, 2019.

Jongoh Jeong, Hunmin Yang, Jaeseok Jeong, and Kuk-Jin Yoon. Improving black-box generative attacks via generator semantic consistency. In *International Conference on Learning Representations*, 2026.

Justin Johnson, Alexandre Alahi, and Fei-Fei Li. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.

Jin Li, Ziqiang He, Anwei Luo, Jian-Fang Hu, Z. Jane Wang, and Xiangui Kang. AdvAD: Exploring non-parametric diffusion for imperceptible adversarial attacks. In *Advances in Neural Information Processing Systems*, 2024.

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

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

Krishna Kanth Nakka and Alexandre Alahi. NAT: Learning to attack neurons for enhanced adversarial transferability. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 7582–7593, 2025.

Krishna Kanth Nakka and Mathieu Salzmann. Learning transferable adversarial perturbations. In *Advances in Neural Information Processing Systems*, 2021.

Muhammad Muzammal Naseer, Salman H. Khan, Muhammad Haris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-domain transferability of adversarial perturbations. In *Advances in Neural Information Processing Systems*, 2019.

Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli. On generating transferable targeted perturbations. In *IEEE International Conference on Computer Vision*, 2021.

Hongyu Peng, Xiang Yuan, and Gong Cheng. PGA: Prior-free generative attack for practical no-box scenario. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13296–13305, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Peng_PGA_Prior-free_Generative_Attack_for_Practical_No-box_Scenario_CVPR_2026_paper.html.

Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241, 2015. doi: 10.1007/978-3-319-24574-4_28.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Fei-Fei Li. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Shangbo Wu, Yu-an Tan, Ruinan Ma, Wencong Ma, Dehua Zhu, and Yuanzhang Li. Boosting generative adversarial transferability with self-supervised vision transformer features. In *IEEE/CVF International Conference on Computer Vision*, pp. 530–540, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Wu_Boosting_Generative_Adversarial_Transferability_with_Self-supervised_Vision_Transformer_Features_ICCV_2025_paper.html.

Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. In *International Joint Conference on Artificial Intelligence*, 2018.

Qilong Zhang, Xiaodan Li, Yuefeng Chen, Jingkuan Song, Lianli Gao, Yuan He, and Hui Xue. Beyond ImageNet attack: Towards crafting adversarial examples for black-box domains. In *International Conference on Learning Representations*, 2022.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

A LOSS AND SCORE DEFINITIONS

All images use the model's $[-1, 1]$ coordinates, so the perturbation radius is $r = 2\epsilon$. Let $\delta_i = z_i - x_i$ be the student's final perturbation, with z_i the final state $z^{(K)}$ of image i, and $\tau_i = t_i - x_i$ be the teacher's, h be the surrogate and $E = \{i \mid \arg\max_c h_c(x_i) = y_i\}$ its clean-correct subset. The true-versus-best-other logit margin is

$$m_h(v, y) = h_y(v) - \max_{c \neq y} h_c(v). \tag{3}$$

Lower margin favors an untargeted attack. We use the following equation for unit normalization that makes the direction and anchor terms below cosine similarities. $q(v) = v/\sqrt{\|v\|_2^2 + 10^{-8}}$. For a within-batch non-source image map σ, perturbed non-source image set, $\widetilde{z}_i = \mathrm{clip}(x_i + \delta_{\sigma(i)}, -1, 1)$. The implemented training terms are

$$\mathcal{L}_{\text{atk}} = |E|^{-1} \sum_{i \in E} \log(1 + \exp(m_h(z_i, y_i))), \tag{4}$$

$$\mathcal{L}_{\text{dir}} = B^{-1} \sum_i [1 - q(\delta_i)^\top q(\tau_i)], \tag{5}$$

$$\mathcal{L}_{\text{mag}} = B^{-1} \sum_i \rho_{0.05}\left(\frac{\|\delta_i\|_1 - \|\tau_i\|_1}{Dr}\right), \tag{6}$$

$$\mathcal{L}_{\text{anc}} = \frac{1}{B(P-1)} \sum_i \sum_{k=1}^{P-1} [1 - q(z_i^{(k)} - x_i)^\top q(t_i^{(a_k)} - x_i)], \tag{7}$$

$$\mathcal{L}_{\text{spec}} = |E|^{-1} \sum_{i \in E} [m_h(z_i, y_i) - m_h(\widetilde{z}_i, y_i) + \gamma]_+, \tag{8}$$

$$\mathcal{L}_{\text{mdr}} = \frac{D^{-1} \|B^{-1} \sum_i \delta_i\|_2^2}{\max((BD)^{-1} \sum_i \|\delta_i\|_2^2, 10^{-12})}, \tag{9}$$

$$\mathcal{L}_{\text{perc}} = [B^{-1} \sum_i \mathrm{LPIPS}(x_i, z_i) - p]_+. \tag{10}$$

Note that $\mathcal{L}_{\text{perc}}$ penalizes the average LPIPS over a batch, and checkpoint selection also uses average validation LPIPS, so neither limits the LPIPS of each individual image. Only the L_∞ radius of Equation 1 is guaranteed for every image. Here B is batch size, D the number of image coordinates, $[v]_+ = \max(v, 0)$, $\gamma = 0.25$, and $p = 0.03$. The smooth-L1 function is $\rho_\beta(v) = v^2/(2\beta)$ for $|v| < \beta$ and $|v| - \beta/2$ otherwise. For K student steps and M teacher anchor intervals, $P = \min(K, M)$ and $a_k = \mathrm{round}(kM/P)$, and $z_i^{(k)}$ is the state of image i after student step k, as in Equation 1. Anchor loss is zero when $P \leq 1$, and attack and specificity losses are zero when E is empty. When several training classifiers are enabled, each batch uses the next one in turn.

Does it attack? For an evaluation classifier f, let E_f be the clean images that f classifies correctly, and define

$$A_f(d) = |E_f|^{-1} \sum_{i \in E_f} \mathbf{1}\{\arg\max_c f_c(\mathrm{clip}(x_i + d_i, -1, 1)) \neq y_i\}. \tag{11}$$

So $A_f(\delta)$ adds every perturbation to the image it was generated from, and $A_f(\delta_\sigma)$ adds each one to a different image. The main text writes A for A_f, since f is always the ResNet-50 victim there.

Does it work better on its own image? The specificity of Equation 2 is the ratio of those two, $S = A_f(\delta)/A_f(\delta_\sigma)$. In the global version, the swapped-in perturbation can come from an image of any class. In the within-class version it always comes from an image of the same class, so only the source image changes, and the class label alone cannot explain the ratio. Both versions measure how much the correct pairing helps, not how visually similar the perturbations are. Training is not independent of this measure. The loss $\mathcal{L}_{\text{spec}}$ makes the same own-image versus swapped-image comparison, using the surrogate's margin instead of counting fooled images, so it is designed to raise S. What the diagnostic adds is that it counts actually fooled images in a separate evaluation, and conditions that use this loss at full weight still lose specificity (Section 7).

Is one shared perturbation responsible? The diagnostic also measures $A_f(\bar{\delta})$, the success rate when the single average perturbation of the batch is added to every image. This is the mean-δ ASR

reported in the tables. A shared pattern is only a working universal attack if that pattern fools the classifier by itself. A run can fail the specificity test with or without a strong shared pattern, and this second question tells those cases apart.

Numerical details. Universal energy U for a whole dataset uses all evaluated images, while the training loss computes it for each batch separately. Both add 10^{-12} to the denominator to avoid dividing by zero. When the swapped-image success rate is zero, the stored ratio also uses 10^{-12} as its denominator, so such values are placeholders rather than real measured ratios. If the own-image success is positive and the swapped-image success is zero, the ratio has no upper limit. If both are zero, or no image is classified correctly to begin with, specificity tells us nothing. A high ratio for a weak attack does not mean the attack is useful and image-specific. The 2.0 floor is a rule we chose for this study, not a universal line between generators that do and do not use their input image.

B WHY THE BASELINES ARE COMPARED AT A FIXED BUDGET

Before settling on a fixed budget, we tuned each baseline's L_∞ budget with an 8-step binary search that aimed to match the student's 4-step mean LPIPS.

Table 4: The same attacks when each one's budget is tuned to match the student's LPIPS, instead of all sharing one budget. Each row ends up with its own radius, none reached the search's target tolerance of 0.0015, and Square's radius shrinks as its query count grows, so part of its apparent query-budget trend is really a shrinking budget.

Attack	ϵ ($/255$)	ASR	LPIPS	ASR at $8/255$
PGD-40	5.12	1.000	0.0207	1.000
CW-margin (L_∞)	6.33	0.999	0.0217	0.999
AutoAttack (APGD)	4.15	0.999	0.0187	0.999
DeepFool	16.50	0.331	0.0001	0.335
Square (100q)	7.78	0.215	0.0235	0.223
Square (500q)	5.84	0.249	0.0178	0.349
Square (1000q)	5.72	0.295	0.0209	0.418
Square (5000q)	5.36	0.336	0.0229	0.486

This approach has three problems. First, the attacks no longer share a common budget. Second, the tuning did not work, since every row missed the search's own target tolerance, some by several times, and was accepted only by a fallback rule that allows any gap under half the target. The LPIPS an attack reaches comes out of a limited, random search over budgets, so there is no guarantee that it hits a chosen quality level. Third, Square's radius shrinks steadily as its query count grows, so the tuned sweep changes queries and budget together, in opposite directions. At a fixed budget, Square gains roughly twice as much from 100 to 5000 queries as the tuned version reports.

DeepFool is unaffected either way. It stops at the nearest point where the classifier's decision changes, whatever the budget, so it changes the image as little as possible.

C ARCHITECTURE AND ABLATION OVERVIEW

Figure 3 shows the whole pipeline, from generating teacher guidance offline, through training the student, to producing bounded perturbations at attack time. Surrogate classifiers are used in the first two stages but not in the third. Figure 4 shows the two generators used in our training experiments. Figure 5 summarizes the ablation's attack success, distortion and within-class specificity for the two reported seeds, and a missing condition has no second-seed result. Appendix R shows the same two held-out images under every method.

D TEACHER SUCCESS VERSUS IMAGE CHANGE

At its default setting ($\epsilon = 8/255$) the teacher changes the image much less than the student does, as measured by LPIPS, so comparing the two at their default settings is not fair. We therefore vary

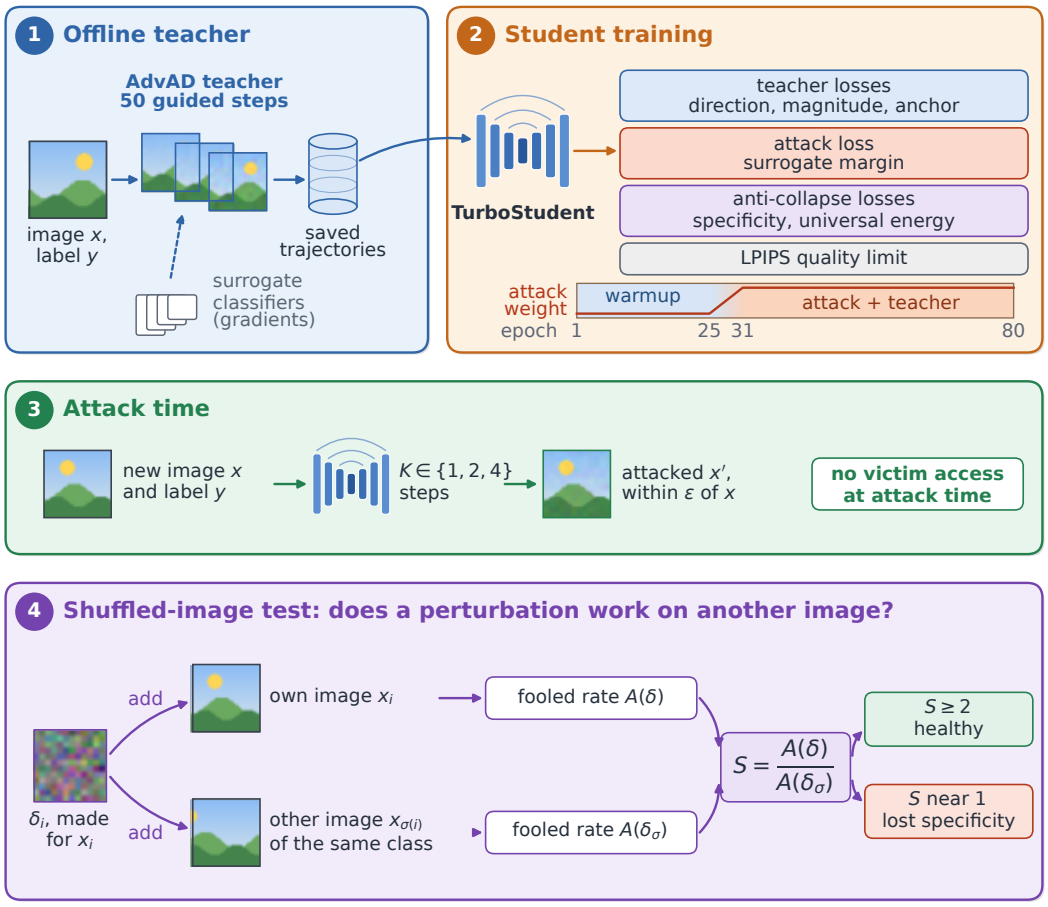


Figure 3: Generating teacher guidance offline, training the student, and producing bounded perturbations at attack time. The teacher and the training losses use surrogate classifiers, but attack time does not. The evaluation tests each perturbation on its own image and on a different image of the same class.

the teacher's ϵ and evaluate each setting on 5,000 validation images, balanced across classes. (The $\epsilon = 8/255$ row agrees with the full 50,000-image split to within half a percentage point, so this smaller sample is representative.)

Table 5: Teacher attack success and image quality as its budget grows.

ϵ	LPIPS	PSNR (dB)	ASR
$8/255$	0.01246	56.26	0.8603
$12/255$	0.01864	54.61	0.9445
$16/255$	0.02645	53.25	0.9829
$24/255$	0.03997	51.38	0.9983
$32/255$	0.05850	49.78	0.9998

At 4 steps, the student reaches a certain attack success for a certain amount of image change. This operating point is LPIPS 0.02619 at ASR 0.97534. Drawing a straight line between the $12/255$ and $16/255$ rows places the teacher at approximately 0.982 attack success at that LPIPS, which is above the student, and the teacher's PSNR is roughly 17dB better throughout this range. The rows are far apart and the teacher's curve bends, so 0.982 should be read as an estimate rather than a measured

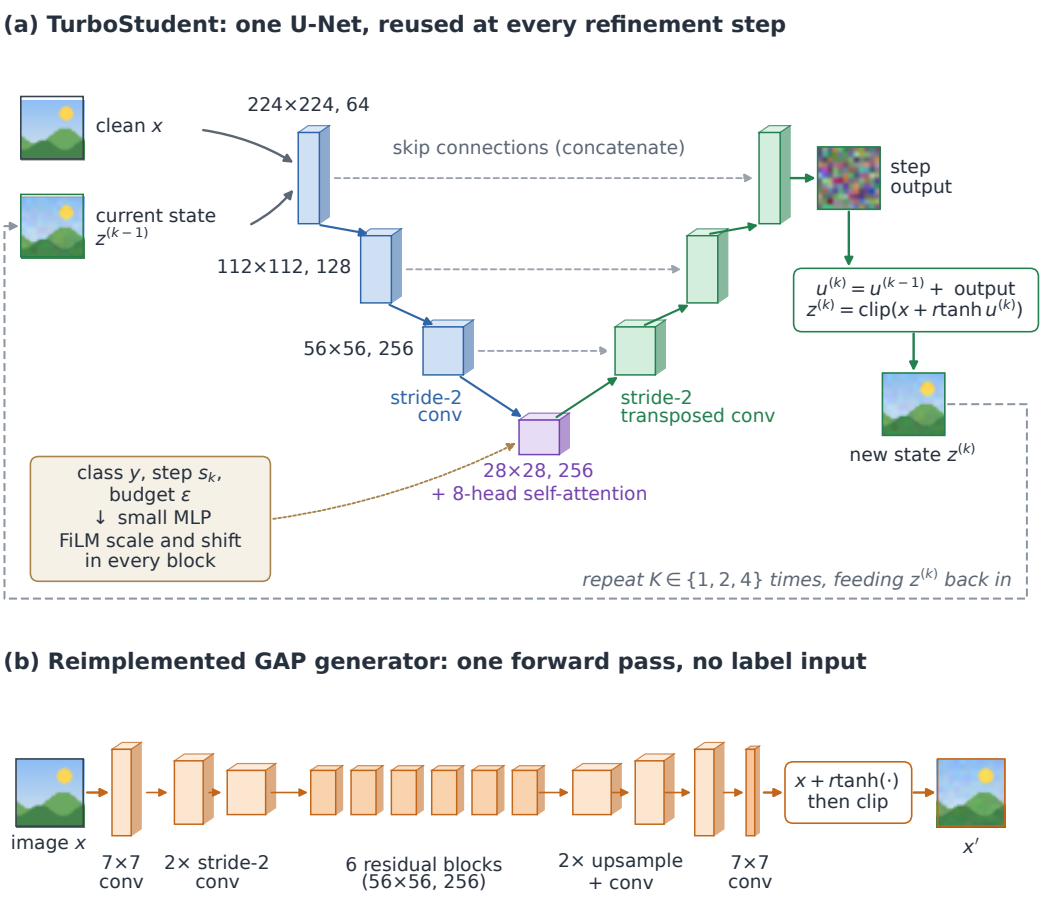


Figure 4: Block diagrams of the two generators. TurboStudent reuses one network for every refinement step, and its FiLM blocks, layers that adjust features using extra inputs, receive the class, the step and the budget. The reimplemented GAP generator runs once per image, with six residual blocks and resize-then-convolve upsampling. We do not claim either architecture as new.

value. The main conclusion, that the teacher is stronger when both change the image by the same amount, does not depend on its exact value.

E SMOOTH VERSUS SIGNED PERTURBATIONS

Suppose the margin behaves linearly near the image, with gradient g. Under $\|\delta\|_\infty \le r$, the largest possible decrease is $r\|g\|_1$, reached by $\delta = -r\,\mathrm{sign}(g)$, which moves every pixel by the full amount. A perturbation that instead points along g itself reaches only $r\|g\|_2^2/\|g\|_\infty$. If the entries of g are independent standard Gaussians in dimension D (the ratio below does not depend on their scale), then using $\mathbb{E}\|g\|_2^2 = D$, $\mathbb{E}\|g\|_1 = D\sqrt{2/\pi}$ and $\|g\|_\infty \approx \sqrt{2\log D}$, the ratio of the two is

$$\frac{\|g\|_2^2/\|g\|_\infty}{\|g\|_1} \approx \frac{\sqrt{\pi}}{2\sqrt{\log D}},$$

which is about 0.34 at $D = 10^3$. So a smooth perturbation is not free. It costs attack strength at a fixed radius, and the teacher's guidance setting moves between these two extremes.

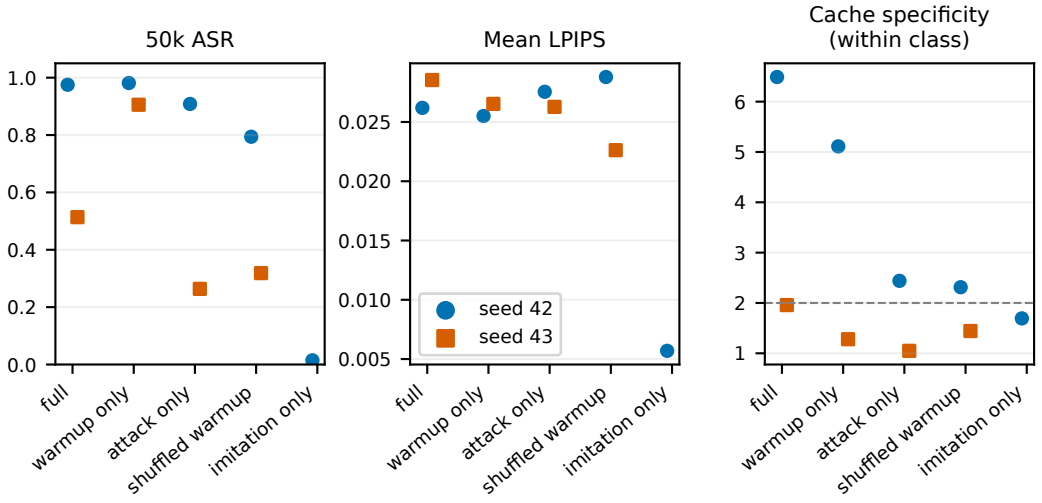


Figure 5: Overview of the ablation, regenerated from the saved evaluation results. ASR and LPIPS use the full validation split, while specificity uses the training images and same-class swapping. The dashed line is the 2.0 floor we fixed in advance. Each point is one training run, not an error bar.

F HYPERPARAMETERS

Table 6 lists the settings for the reported student. The ablation conditions change only the settings named in Section 7, and every other value is the same.

Table 6: Hyperparameters for the reported student. All values are the exact default or changed command-line settings used to produce the reported checkpoint.

Parameter	Value	Parameter	Value
Architecture	`turbo` (3-scale U-Net)	Batch size	8
Channels / blocks	64 / 2	Learning rate	2×10^{-4} (cosine)
Student steps (eval)	1, 2, 4	λ_{atk}	14.0
ϵ	$8/255$	λ_{dir}	2.5
Attack start epoch	25	λ_{mag}	1.0
Attack warmup epochs	6	λ_{anc}	1.0
Adversarial surrogate	ResNet-50	λ_{spec}	0.05
LPIPS target / gate	0.03 / 0.03	λ_{mdr}	0.02
Trajectory augmentation	horizontal flip	λ_{perc}	100.0

G A PUBLISHED GENERATOR UNDER THE SAME QUALITY LIMIT

`attack_only` follows the earlier generative-attack recipe, where a feedforward generator is trained to fool the classifier without a teacher, but it uses *our* architecture. This appendix asks whether the failure depends on that architecture, by moving both recipes onto a published one.

We reimplement the residual generator of Poursaeed et al. (2018) at its published size. We train it once with `attack_only`'s teacher-free loss and once with `full`'s complete teacher-guided recipe, using the same teacher data, epoch schedule, quality limit and checkpoint-selection rule. By design it runs only once per image, so it is scored at one step.

A controlled pair, repeated at three seeds. The two variants keep the published architecture, data, optimizer, number of epochs, quality limit and selection rule the same, but change a group of

loss terms together with their schedule. Before running the extra seeds we committed to the plan, which was seeds 43 and 44 of both variants, the same judging rule used for every other run in this paper, and three possible outcomes written down in advance. A healthy teacher-free run would have made us withdraw the claim about the training objective, and a degraded teacher-guided run would have shown that the protection is only partial on GAP, as it already is on `TurboStudent`.

Table 7: Both GAP variants at every seed we ran, judged by the rule fixed before the extra runs were launched, which calls a run degraded if its within-class specificity on the training images is below 2.0. ASR and LPIPS are on the full validation split, at the checkpoint chosen under the quality limit. Every run we launched appears here.

Arm	Seed	Epoch	ASR	LPIPS	Spec. (wc)	Verdict
teacher-free direct fooling	42	10	0.0395	0.0279	1.01	degraded
teacher-free direct fooling	43	5	0.1868	0.0261	1.19	degraded
teacher-free direct fooling	44	70	**0.9159**	0.0240	**1.51**	degraded
full teacher-guided recipe	42	80	0.8660	0.0246	2.33	healthy
full teacher-guided recipe	43	75	0.9115	0.0245	2.20	healthy
full teacher-guided recipe	44	80	0.8497	0.0234	2.16	healthy

By this rule, no teacher-free run clears the floor and every teacher-guided run does, although the guided runs clear it only narrowly. The seed-42 teacher-free ratio divides two attack rates of only a few percent, so on its own it says little. The evidence for repetition comes from the two teacher-free runs that attack well enough for the ratio to mean something, together with all three guided runs. The fifty-seven released checkpoints of Appendix I come from the same family of Johnson-style residual generators (Johnson et al., 2016). They match this generator's depth, width and parameter count to within 0.05%, but they are not identical, since they differ in normalization and upsampling. They are trained with feature-space transfer losses, and none of them falls below the floor. That observation agrees with the controlled comparison, but it is not a second controlled experiment.

How strong this evidence is. What the results support is that, on the same GAP architecture, the teacher-free recipe repeatedly comes with lost image specificity, while our full teacher-guided recipe repeatedly keeps it. This comparison changes the teacher losses and the attack schedule together, so it neither isolates teacher guidance nor confirms a causal mechanism, and three seeds of one architecture is a repeatable count rather than a rate. The fifty-seven released checkpoints fit the same recipe-level reading, but they are not independent controlled tests, because they differ in objective, data and training pipeline all at once, and we did not produce them.

The seed-44 teacher-free run is the clearest case in this paper of the failure going unnoticed. Its two earlier seeds were weak attacks, so a practitioner could have caught them by attack success alone. This one is not weak. It attacks more successfully than any teacher-guided GAP run while staying within the same quality limit, and it has *less* universal energy than even our healthiest `TurboStudent` run. Its perturbations are also nearly uncorrelated with one another pixel by pixel. This shows that they differ in pixel space, though it does not rule out a shared motif that appears at different positions. Yet its attack success drops by only about a third when each perturbation is added to a different image of the same class. So three measures, namely attack success, image quality and the universal-energy score this paper otherwise relies on, all look healthy on a run whose perturbation fools the victim on 62% of the wrong images. It is the failure of Section 7 reached from the opposite direction, with many different patterns that are still generic instead of one shared pattern. Only the shuffled-image test tells them apart.

Likewise, the seed-42 run's sizeable universal energy does not mean it has a working shared attack, because its average perturbation fools almost no images. The two teacher-free runs that do attack show the real failure instead, since their perturbations keep most of their attack success on other images of the same class.

The published generator has 7.85M parameters against `TurboStudent`'s 14.75M and runs one forward pass instead of four, so its rows in Table 3 do not compare networks of matched size with ours. The two GAP variants, which share one architecture, do.

Does the teacher-guided recipe carry over to this architecture? The guided variant uses `full`'s complete recipe, in which the teacher stays active after warmup, and we did not run a GAP version of `warmup_only`. We committed in advance to a conditional prediction based on this project's own rule of thumb, that an imitation cosine peak below ~ 0.08 means the warmup gives too little signal to select a good basin. The imitation cosine measures how closely the student's perturbation points in the teacher's direction. A peak at or above 0.08 by epoch 25 predicted specificity at or above 2.0, and a peak below 0.08 predicted specificity below 2.0.

The peak stayed below the threshold, and specificity cleared the floor anyway, within the quality limit. **The prediction failed**, and we report it as a failure rather than adjusting the threshold after seeing the result. The full recipe lifted this architecture's specificity from well below the floor to above it, exactly where our own rule predicted it would not. The rule is not wrong that a weaker imitation signal goes with worse basin selection, and the teacher-strength results in Appendix K point the same way. But 0.08 is not a reliable cutoff below which the teacher stops helping. On this architecture, a weaker signal than `TurboStudent` ever needed was still enough for a real improvement. Read narrowly, this is the paper's only test of the full recipe on an architecture other than `TurboStudent`. The prediction was made for seed 42 and we do not restate it for the extra seeds, but the outcome it got wrong holds, since all three guided runs clear the floor. This is evidence that the protection is not obviously specific to our own network, not evidence that it works broadly. One architecture at three seeds is evidence across architectures, not proof across architectures.

H Where this sits among generative attacks

Table 8 compares the closest related methods on the properties this paper cares about. We read each paper's described method rather than rerunning it, since their reported numbers come from setups that cannot be compared with ours. Generators that fall into poor, degenerate solutions are not new. Jang et al. (2019) observe a generator ignoring its noise input, reducing diversity for fixed image and other conditioning inputs, and PGA (Peng et al., 2026) targets degenerate solutions and poor local optima in a generative attack's own training. Closest of all, NAT (Nakka & Alahi, 2025) reports that BIA (Zhang et al., 2022) and LTP (Nakka & Salzmann, 2021) "consistently produce adversarial images with nearly identical patterns, regardless of the input image". That near-universal form is prior work, and we do not claim to have observed it first.

What we add is measurement rather than discovery. NAT reports this failure from looking at outputs and proposes no numeric test of whether perturbations depend on the input image. We supply one, comparing attack success on the source image with attack success on a swapped-in image, plus a within-class version that controls for the class label (Appendix M). We use it to show that attack success, image quality and the step curve all look healthy on runs it marks as degraded, including a second form, diverse but generic, that looking at outputs for shared patterns would not reveal.

I Released third-party generators keep source specificity

Every other specificity number in this paper is measured on a generator we trained ourselves. That leaves open the reading that only our network does this. It also leaves NAT's observation (Nakka & Alahi, 2025), that BIA (Zhang et al., 2022) and LTP (Nakka & Salzmann, 2021) "consistently produce adversarial images with nearly identical patterns, regardless of the input image", as the only statement about published generators, and that statement is based on looking at outputs. We therefore ran the shuffled-image test on every released, untargeted ImageNet generator that takes an image as input and that we could obtain. These are the twelve BIA checkpoints (four surrogates $\times$ three variants), the five LTP checkpoints (one per surrogate) and all forty NAT checkpoints (one per attacked neuron), fifty-seven in total. All three families share one architecture, the CDA-style residual generator with 7.84M parameters.

The measurement uses the same code as every other specificity number here, not a separate reimplementation. Each generator's perturbations on the first 2,048 images of our held-out set go through the same routine that produces our own results, against the same ResNet-50 victim. The only new code converts between the two pixel-range conventions. We report two budgets throughout, each generator's published $10/255$ and our own $8/255$, because specificity is a ratio of two attack rates, and a budget difference could otherwise decide a comparison between methods. Before running

Table 8: The closest related generative attacks, compared on this paper's specific concerns. "Discusses collapse" asks whether the paper discusses any failure where its generator produces overly similar outputs. "Lost source dep." asks specifically about the failure this paper studies, in which a perturbation still works too well on an image other than its source, whether or not all perturbations collapse onto one shared pattern. "Qualitative" marks a paper that reports this failure from looking at outputs, without a numeric test for it.

Work	Image-cond.	Discusses collapse	Lost source dep.	Teacher guidance	Univ. diagnostic
GAP (Poursaeed et al., 2018)	Yes	No	No	No	No
Jang et al. (2019)	Yes, plus noise	Yes (noise)	No	No	Diversity loss
SCGA (Jeong et al., 2026)	Yes	No	No	EMA teacher	Reliability
dSVA (Wu et al., 2025)	Yes	No	No	No	No
NAT (Nakka & Alahi, 2025)	Yes	Yes (input-agnostic)	Qualitative	No	Visual only
PGA (Peng et al., 2026)	Yes	Yes (local optima)	No	No	No
This work	Yes	Yes	Quantified	Paired trajectory	Three diagnostics

anything we wrote down three possible outcomes. BIA and LTP below the floor with NAT above it would confirm NAT's observation with numbers. NAT below the floor too would mean that variety *across* generators is not the same as source-image dependence *within* one. Every generator clearing the floor would mean the failure is specific to a direct fooling objective, and this paper's framing would have to narrow.

Table 9: Fifty-seven released third-party generators under this paper's test, on our held-out images against ResNet-50. "Within-class spec." is the range of within-class specificity (Appendix M), and "Below floor" counts generators under 2.0. Univ. is the median universal energy. Not one generator falls below the floor at either budget. For comparison, on the same images, victim and code at $8/255$, our healthy `full:42` scores above the top of every released range, and our four collapsed runs all fall below a floor that no released generator comes near.

Method	Budget	n	ASR (med.)	Within-class spec.	Below floor	Univ.
BIA	10/255	12	0.690	2.97–4.36	0/12	0.039
BIA	8/255	12	0.500	3.69–4.84	0/12	0.039
LTP	10/255	5	0.810	2.58–5.77	0/5	0.053
LTP	8/255	5	0.623	2.92–6.08	0/5	0.052
NAT	10/255	40	0.770	2.99–5.56	0/40	0.068
NAT	8/255	40	0.583	3.34–6.23	0/40	0.066

The third outcome is the one that happened, and it is the one that works against us. Zero of fifty-seven released generators fall below the 2.0 floor at either budget, and the lowest within-class ratio anywhere in the set is 2.58. These generators are clearly image-specific. We report this as a count and not as a rate, because these fifty-seven are not fifty-seven independent samples. Forty are NAT checkpoints that differ only in which neuron they attack, all three families use one CDA-style architecture, and their training pipelines descend from one another. The number of truly independent training setups here is closer to three than to fifty-seven, and a confidence interval over the checkpoint count would wrongly present that as repeated evidence. The silent loss of specificity this paper studies is therefore *not* a property of feedforward attack generators in general. On this evidence it is associated with training one with a direct objective to fool the victim, which our `attack_only`

and `shuffled_warmup` conditions do and BIA, LTP and NAT do not, since all three are trained with feature-space transfer losses and none with a direct victim-fooling term. The main text's framing is narrowed accordingly, because this measured set does not support extending the claim to the published literature.

The same measurement bears on NAT's visual observation, though it does not settle it. One numeric version of "nearly identical regardless of the input image" is the mean pairwise cosine similarity between perturbations of different images, which is 1 for identical perturbations and close to 0 for unrelated ones. On BIA and LTP's own released checkpoints it is 0.013–0.072 (median 0.046), and across all fifty-seven it never exceeds 0.134, whereas our own collapsed runs reach 0.892 on the same measure, images and victim, several times higher than anything released. We state what this does and does not show. It does show that these perturbations are nearly unrelated to one another pixel by pixel at the tested budget, and that they still depend on their source image. It does not rule out a shared visual texture. Cosine compares flattened perturbations pixel by pixel after the budget is applied, so two perturbations carrying the same motif at different positions score near zero, and NAT's figure looks at the raw generator output rather than the bounded perturbation we measure. Looking at the images and measuring how they attack answer different questions, and ours is the question a practitioner checking specificity would ask. NAT's own forty generators sit slightly *higher* on both cosine and universal energy (medians 0.067 against 0.046) than the BIA and LTP generators they are compared with, although every one of them clears the specificity floor comfortably and the ranges overlap. We read this as the two sets being indistinguishable on these measures, not as a reversal.

Compared with our own students on the same held-out images, the separation is complete. Of the ten of our runs with held-out results, the six healthy ones sit above the floor and the four collapsed ones below it, while no released generator comes near it. So the test splits our own runs into two groups while placing every measured released checkpoint in the healthy group, which is the main result of this appendix. The test does not simply flag everything, and what it flags is specific to how we train.

Because these runs were also measured on their own training images, this pair answers an objection that the single-seed version could not. A collapsed run might simply have memorized the images it trained on, giving perturbations that look image-specific there and generic everywhere else. It does not. Across those ten runs, held-out global specificity stays within about fifteen percent of the training-image value, and the collapsed runs stay within a few percent of it, so they are just as degenerate away from the training set as on it. Universal energy is essentially unchanged in every case.

Two limits apply to the comparison. The third-party generators take an image as input like ours, which is what makes the test usable on them at all, but they take no class label, whereas our students do. The two kinds of swap still differ for them, because the image itself carries class information. What changes is only that they have no separate label input through which class information could leak, so the within-class version answers a narrower question for them than for us. We report both anyway. Also, the comparison rows for our own students use the full 10,000-image held-out set rather than its first 2,048 images. The floor verdicts do not change because of this, but the samples are not identical.

J When amortization pays off

Our remaining efficiency claim is amortization, meaning that the student needs no gradients and no queries from the victim at attack time. This invites an obvious objection. Training the student needed a lot of victim access, so the saving may only be moved rather than removed. This appendix estimates the victim access used to build the teacher data and to train one 80-epoch recipe, and asks after how many attacked images that total becomes smaller than what per-image attacks spend. Validation, tuning, final evaluation, other seeds and computation that does not touch the victim are left out, so this is not an estimate of total experimental cost or time.

We count only `resnet50`, the victim. The teacher also queried `robust_engstrom_r50`, which costs real computation but is not access to the victim, so we do not count it. Leaving it out makes our own total smaller, which we state openly. Costs are counted in forward passes, with a backward

pass counted as two forward passes. For each image, the teacher spends one forward pass to check the clean prediction, then one guided gradient per step plus one forward pass for its early-stop check, over $20{,}000$ images and 50 steps. Every image pays for all 50 steps, because early stopping only freezes a fooled image and does not remove it from the loop. Training is counted only for the epochs in which the attack term is active (epochs 26–80, 55 of 80), because before that the training loss does not call the classifier. Each such training batch makes three calls to the victim, two of them with gradients. That totals 4.02M forward passes for the teacher data and 6.93M for training, 10.95M in all.

Table 10: Estimated victim cost per image for each attack, and the number of images after which the student's total victim access becomes the smaller of the two.

Attack	Victim cost per image	Crossover (images)
PGD-40 / CW-margin	120	91,250
Square, 1000 queries	1,000	10,950
Square, 5000 queries	5,000	2,190

The student pays its victim cost once, before the first image is attacked, while each competitor pays again for every image. So each competitor's total overtakes the student's exactly once (Table 10). Against PGD-40 that happens within two ImageNet validation sets. Square costs use the allowed query budgets rather than the number of queries actually made. APGD is omitted here because its number of calls depends on the target classes, restarts and how many images are still unfooled, not on a fixed 100 gradient steps.

Three caveats apply. Counting a backward pass as two forward passes is an assumption, but counting it as one moves every crossover by less than a factor of two, all in the same direction, so the ordering does not depend on it. The estimate assumes one student serves all N images against the same family of victims, which is the setting we claim and not a general one. And PGD has no training cost at all, so below the crossover it is simply the cheaper choice. The argument here is about where the lines cross, not about the student being cheaper everywhere.

K Teacher strength at a smaller scale

The experiments in Section 7 change whether the teacher is present and whether it is correctly paired. They do not change how much signal the teacher provides, and the basin-selector idea also makes a prediction about that. What should matter is how strong the teacher's guidance is, not how good an attacker the teacher is.

An earlier, smaller-scale phase of this project happens to bear on this. Three teacher settings were run on a 10-class subset at 128×128 with an earlier version of our code. They differ in how the teacher attacks, since one freezes each image the moment it is fooled, so it changes images as little as possible, while another always runs all of its steps. They also differ in how many surrogate classifiers they attack, and as a result in how much of the perturbation budget they use. The student architecture, loss and data are the same for all three.

Table 11: Stronger teacher signal gives a better student, but the teacher's own attack success does not predict this. Budget use is the teacher's mean $|\delta|/r$, the average fraction of the allowed change it actually uses. Specificity here uses *global* swapping on the training images, not the within-class version used elsewhere in the paper, so these values can be compared within this table but not with other tables. Univ. is universal energy. The two teachers that fool almost every image (0.9998) produce the worst and the best student on both measures.

Teacher	Own ASR	Budget use	Specificity	Univ.	Student ASR
1 surrogate, early stop	0.9998	0.045	4.03	0.0827	$0.8847^{\dagger}$
2 surrogates, early stop	0.8632	0.161	7.17	0.0327	0.9499
1 surrogate, full budget	0.9998	0.221	7.72	0.0137	0.9499

Specificity rises and universal energy falls steadily as the teacher uses more of its budget, with a sixfold difference in universal energy between the two extremes. The teacher's own attack success tells us nothing here, since the two teachers that fool almost every image sit at opposite ends. The teacher with the weakest signal is also the only one whose student misses the quality limit (†, slightly over the 0.03 LPIPS target), so the damage is not limited to specificity. Everything gets worse together, which is what the basin-selector idea predicts when the warmup gives too little signal to select a good basin.

We report this as supporting evidence from a different angle and at a different scale, not as part of the experiments we committed to in advance, and we do not overstate it. It is only three runs. The settings differ in more than budget use, so budget use is the variable that lines up with the outcome, not one we changed on its own. The dataset, image size and code version also differ from the ImageNet-1k results in the main text. Testing teacher strength properly would need a teacher sweep at full scale, which we have not run.

L Implementation differences from the reference teacher

AdvAD (Li et al., 2024) runs its reverse diffusion with the denoiser fixed to the initial noise, so its predicted clean image (`pred_x0`) stays exactly equal to the input image until guidance changes it, and the whole process reduces to adding up a perturbation. Our implementation tracks that perturbation directly, instead of carrying the noisy state and recomputing `pred_x0` at every step. This gives the same result mathematically, but it lets us enforce the L_∞ budget exactly in pixel space, instead of the reference implementation's approximate enforcement in noise space. It does not change the guidance direction, which is the reference method's own normalized ensemble gradient throughout.

A second, separate difference does change how the guidance is spread across steps, so the projection change is not the only difference. Rewriting the reference update in our accumulated form introduces an extra weighting factor that grows without limit as the noise level approaches zero. If kept, it would give the single largest of our $T = 50$ steps 26.5% of the total push (computed directly from our implementation, not from the reference schedule). That would turn the method's smooth, coarse-to-fine schedule into what is effectively a few-step one. We drop it on purpose. Projecting onto the total budget still enforces the same final limit, but how the budget is spread across steps differs from the reference schedule. This changes the size of each step, not the guidance direction at any step, but it is a real, deliberate change and we state it as one rather than treat it as an unimportant detail.

M Specificity with within-class swapping

The student receives the true class label, so a globally shuffled swap, which pairs each perturbation with a random other image in the batch, can mix up class specificity with image specificity. An image from a different class may simply be easier or harder to attack, for reasons that have nothing to do with whether the perturbation is image-specific. Every specificity number in this paper therefore uses the stricter within-class swap, which pairs each perturbation with a different image of the *same* class. This appendix compares the two.

Table 12: Specificity under global and within-class swapping, on the training images, for both training seeds. The within-class version is always stricter, as expected. The ordering of conditions stays the same, although one borderline verdict changes (`full` at seed 43, discussed below).

	Seed 42		Seed 43	
Condition	Global	Within-class	Global	Within-class
`full`	7.29	6.49	2.17	1.96
`warmup_only`	5.77	5.11	1.32	1.28
`attack_only`	2.77	2.44	1.11	1.05
`shuffled_warmup`	2.53	2.31	1.55	1.44

Every condition scores somewhat lower under the stricter test, and the ordering stays the same, so the main claims do not depend on which swap is used. One verdict does depend on it. `full` at seed 43 clears the 2.0 floor under global swapping but falls just below it under within-class swapping, which is why we count it as degraded. No other run in this two-seed comparison comes nearly as close to the floor under either swap, and the next closest is about seven times farther away. So in this comparison it is the one case where the exact floor value, and not only the choice of swap, decides the verdict.

N A SECOND TRAINING SEED

Four of the five conditions in Table 3 were retrained at a second seed, changing only the seed and where the results are saved. `imitation_only` has one run and is left out of Table 13, since it barely attacks, so there is no final result for a second seed to test. `full` has since been run at twelve seeds (Table 15) and `warmup_only` at three, so the number of seeds differs between conditions, and no comparison between conditions here assumes equal repetition. The teacher data, the evaluation images, the schedule and the code are identical, and all runs were scored on the same 50,000 validation images.

Table 13: The same conditions at two training seeds, giving the per-run values behind Table 3. Spec. is *within-class* specificity on the training images, the measure we fixed in advance and use throughout, and Univ. is universal energy. Every seed-43 run here passes the quality limit, so none of them would be rejected by the method's own selection rule.

	Seed 42			Seed 43		
Condition	ASR	Spec.	Univ.	ASR	Spec.	Univ.
`full`	0.9753	6.49	0.080	0.5137	1.96	0.555
`warmup_only`	0.9813	5.11	0.092	0.9056	1.28	0.318
`attack_only`	0.9084	2.44	0.309	0.2637	1.05	0.881
`shuffled_warmup`	0.7940	2.31	0.470	0.3186	1.44	0.689

Runs split apart when the attack objective starts, and the seed still matters. `full` and `warmup_only` differ by one setting, which ends imitation after epoch 31 and cannot have any effect before epoch 32. Up to epoch 31 they are the same program with the same seed, run as separate processes.

Table 14: Universal energy for two runs that are the same program up to epoch 31. The attack objective starts at epoch 26.

	Seed 42		Seed 43	
Epoch	`full`	`warmup_only`	`full`	`warmup_only`
25 (attack off)	0.129	0.129	0.129	0.129
26 (attack on)	0.805	0.825	0.561	0.787
31	0.519	0.517	0.975	0.676

At both seeds the pair agrees almost exactly until the attack objective starts. At seed 42 they then drift apart and come back together, within a fraction of a percent by epoch 31. At seed 43 they end up 44% apart, with `full` falling into the fully collapsed state, where every perturbation is nearly the same one, while `warmup_only` does not. So the start of the attack objective comes with much larger run-to-run variation. Two identical programs with one seed take clearly different paths, and a difference of that size between conditions after this point cannot be credited to the condition. The seed still matters, and more than any condition. At seed 43 all four conditions ended degraded, including both runs of this pair, while at seed 42 neither teacher-guided condition did. Keeping the teacher after warmup does not rescue a run that has already degraded, and `full` at seed 43 recovers only partway, ending with about seven times more universal energy than at seed 42 and just below the floor. Appendix O supports this epoch by epoch with four more geometric measures on the saved checkpoints.

The final ordering does not repeat. `attack_only` attacks more successfully than `shuffled_warmup` at seed 42 but less successfully at seed 43. The reversal is not limited to attack success, since at seed 43 `attack_only` is also the more collapsed of the two on both specificity and universal energy. We therefore claim no final ordering between these two conditions. The simplified argument in Appendix Q does not give one either, because it only says that a constant solution is optimal under independence and does not rank final results. Once both conditions have collapsed toward one shared perturbation, which of them collapsed further is not stable, and comparing two collapsed runs by attack success only compares noise.

A rising step curve does not prove per-image behavior. `warmup_only` at seed 43 attacks strongly, its success rises with the number of steps the way a healthy run's does, and it passes the quality limit, yet its perturbation still fools the classifier on most of the *wrong* images. Attack success, image quality and the step curve all look healthy while the run is close to universal. Only the shuffled-image test tells them apart. Two independent, cheap measures agree here, and both are wrong.

What does repeat is the warmup result. The pairing comparison of Appendix Q repeats at both seeds, in size and direction, with no attack term active during that period. Final rankings among these conditions do not repeat, although some comparisons after warmup can still repeat, as the GAP recipe pair shows.

Additional seeds. `warmup_only` was retrained once more (seed 44), reaching ASR 0.9790, within-class specificity 4.27 and universal energy 0.130. That run is healthy, and much closer to seed 42 than to seed 43's collapse. `full`, where the collapse claim matters most, was then retrained at more seeds under a plan registered in advance. The plan fixed the seeds, the judging rule (within-class specificity below 2.0 on the training images means degraded) and the requirement to report every launched run. The planned study is complete. All nine of its seeds ran, none was excluded, and together with the three earlier runs that makes twelve scored runs. The analysis cutoff is seed 53, the last planned seed.

Table 15: Scored `full` runs through seed 53, judged by the rule registered in advance for the planned runs. Seeds 42–44 were exploratory. ASR is on the full validation split at the checkpoint chosen under the quality limit, and specificity is within-class specificity on the training images.

Seed	ASR	Spec. (wc)	Univ. energy	Verdict
42	0.9753	6.49	0.080	healthy
43	0.5137	1.96	0.555	degraded
44	0.9572	6.68	0.137	healthy
45	0.5688	1.98	0.539	degraded
46	0.8625	1.56	0.305	degraded
47	0.9760	8.53	0.070	healthy
48	0.7388	2.29	0.448	healthy
49	0.8470	1.37	0.342	degraded
50	0.9630	6.22	0.131	healthy
51	0.9474	7.72	0.113	healthy
52	0.9711	6.60	0.136	healthy
53	0.6514	2.51	0.445	healthy

Eight of twelve are healthy, so the exact two-sided 95% interval on that fraction is $[0.349, 0.901]$. This interval is exact only if every run is treated as an independent coin flip with the same odds. The first three runs were exploratory and the other nine were planned. It is not a precise estimate of how often collapse happens in general. The study was designed to turn a single anecdote into a repeatable count, not to estimate a probability.

The final results do not show two clearly separated groups, since the closest healthy and degraded runs are only 0.32 ratio units apart. Seeds 48 and 53 sit just above the floor, far below the other healthy runs, and they fill the gap that the first six runs seemed to show. Seed 49 attacks far more successfully than seed 53, yet seed 49 fails the floor while seed 53 clears it, so higher attack success does not mean a better specificity verdict. We keep the registered 2.0 floor, but two degraded runs lie just below it and two healthy runs only modestly above it, so a 1.9 floor would leave two degraded

runs rather than four, and a 2.5 floor would give five. The count therefore depends on where the floor is set, and we report it as a count with these margins rather than as a clean split into two groups.

O CHECKPOINT GEOMETRY AS THE ATTACK OBJECTIVE STARTS

Table 14 shows where the runs split using one measure, universal energy, taken from the training logs on the full training set. Here we test the saved checkpoints of the same two seed-43 runs directly, on the CPU and with no victim gradients, using 128 held-out images. We add four geometric measures that do not appear elsewhere in this paper. These are the pairwise cosine similarity between perturbations, the share of variation explained by the single strongest direction in the set of perturbations (the first principal component), the effective rank, which roughly counts how many independent directions the perturbations use, and each image's distance to the average perturbation.

Table 16: An independent geometric check of the seed-43 split, on 128 held-out images, computed on the CPU. Univ. here is universal energy measured by this check itself (compare it with the training-time value in Table 14 at the same epochs). Rank is effective rank out of 128, and Dist. is the mean distance of each perturbation to the sample's average perturbation.

	`full`		`warmup_only`	
Epoch	Univ. / Cosine	Rank / Dist.	Univ. / Cosine	Rank / Dist.
20	0.037 / 0.032	39.3 / 4.27	0.040 / 0.040	39.5 / 4.09
25	0.033 / 0.028	41.2 / 4.41	0.025 / 0.017	34.5 / 4.95
30	0.966 / 0.965	10.2 / 3.72	0.710 / 0.694	13.0 / 11.17
35	0.972 / 0.971	8.4 / 3.17	0.483 / 0.488	29.0 / 14.26

Three of the four new measures agree closely with universal energy and with each other. Cosine similarity follows universal energy almost exactly, effective rank drops roughly fourfold exactly where `full` gets stuck near the fully collapsed state, and the distance of each perturbation to the average falls the same way. That drop is what collapse means geometrically, with every perturbation sitting near one shared point. Between epochs 30 and 35 `warmup_only` recovers on all three, more than doubling its effective rank, while `full` keeps sinking. This supports the epoch-31 result in Table 14 with a different calculation on a separate set of images.

The fourth new measure, the share of the first principal component, agrees only in direction, since at epoch 30 `warmup_only`'s value is close enough to `full`'s that the two only separate clearly at epoch 35. We see this as a weakness of the measure rather than a contradiction. Once perturbations cluster around a shared point, the centered data it is computed from also shrinks toward zero, so it becomes a ratio of two small, noisy numbers. We report it because one of the five measures, counting universal energy, agreeing only loosely is part of the answer this appendix set out to find. The check uses only 128 images, so we do not report its own specificity ratios, because with so few images an own-image versus swapped-image ratio is dominated by random noise.

P AGREEMENT BETWEEN MEASURES ACROSS ALL SCORED RUNS

The scatter plot in the main text uses one seed. Figure 6 plots the same relationships across all 26 independently scored runs, a fixed set made up of `full` seeds 42–53, every completed ablation condition and both GAP recipes. The set is an explicit list rather than whatever happens to be saved, so the figure and its caption cannot drift apart.

Panels (a), (b) and (d) repeat, across all runs, claims that the main text makes from individual runs, and panel (c) is the one new check, that the global and within-class versions of the swap test agree. Universal energy detects a shared pattern, but it shows neither whether that pattern attacks nor whether diverse perturbations still work on the wrong image. Mean-δ ASR answers the first question and the shuffled-image test answers the second, so together they cover both forms.

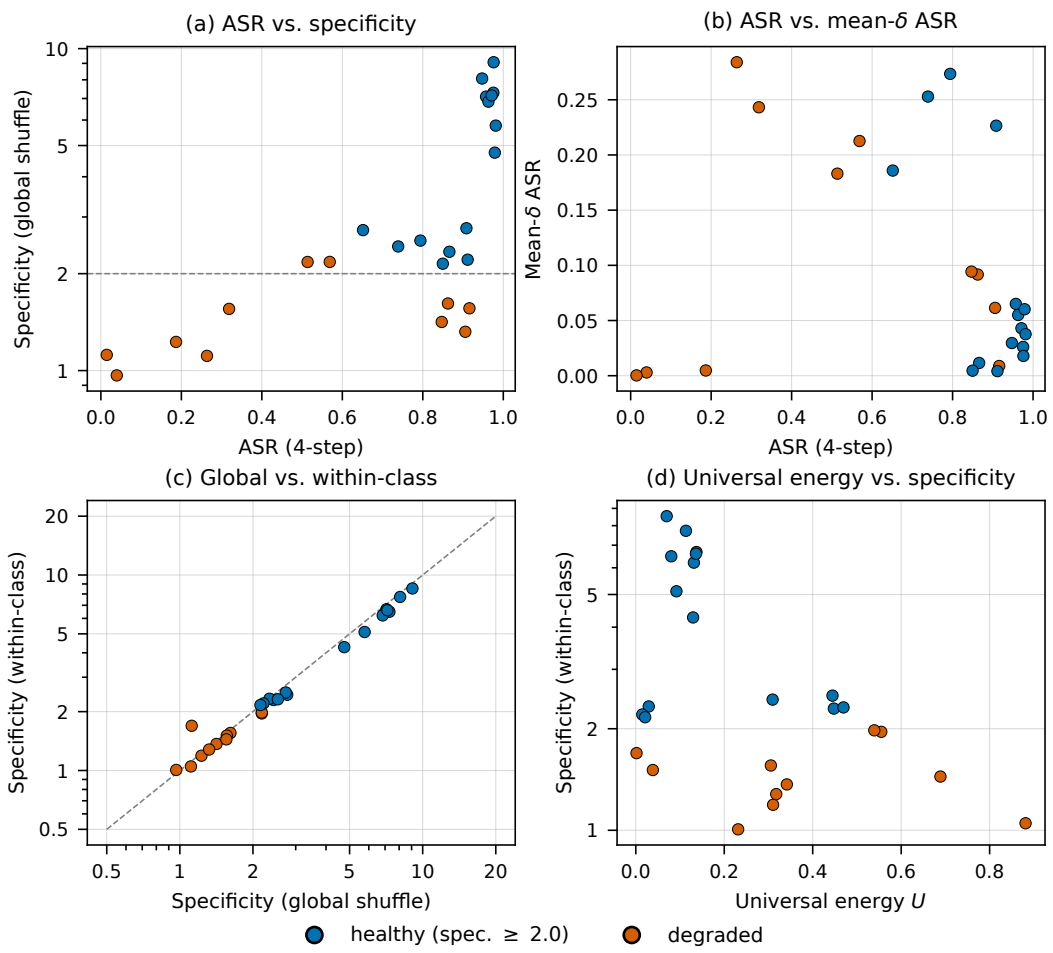


Figure 6: Agreement between measures across all 26 scored runs in the set. Color shows whether a run's within-class specificity is above or below the 2.0 floor, the stricter measure that controls for class (Appendix M). In (a) and (b), attack success does not order runs by either measure. Two runs with nearly the same attack success end on opposite sides of the floor, and the same two runs differ almost fourfold in the success of their average perturbation. In (c), global and within-class specificity agree closely. In (d), universal energy misses the diverse but generic form, since `gap_baseline:44` is degraded despite having less universal energy than the healthy `full:42`.

Q THE WARMUP PHASE ON ITS OWN

Simplified motivation. Let a target τ be independent of all inputs ξ given to the generator, and let the loss $\ell(d, \tau)$ of an output perturbation d not depend on ξ in any other way. If $R(d) = \mathbb{E}_\tau[\ell(d, \tau)]$ has its minimum at d_*, then over all unrestricted generators G the average loss satisfies $\mathbb{E}_\xi R(G(\xi)) \geq R(d_*)$, with equality for the constant generator $G(\xi) = d_*$. So reusing one perturbation on every image is then an optimal solution. This does not assert uniqueness or say that the optimum is an average, and it says nothing about which solution training actually reaches. It is also not a description of our actual warmup. Our code shuffles the teacher perturbations and clips them around the recipient image, which brings back a dependence on that image, and the perceptual loss also depends on that image. The smooth-L1 magnitude loss need not estimate an average either. We therefore test the effect of pairing with experiments, here and in Section 7, rather than relying on this simplified optimum.

This experiment isolates pairing with the attack term turned off. `warmup_only` and `shuffled_warmup` have identical settings apart from the one that shuffles teacher pairing. The other setting they share, which ends imitation at epoch 31, has no effect before that epoch, and

both used the 80-epoch schedule, so their learning rates match epoch by epoch. Epochs 1–24 are therefore a test of pairing alone, with the attack weight held at zero.

Table 17: Universal energy (the energy of the batch's average perturbation divided by the average energy of each perturbation) during the warmup, with the attack term off throughout. Correct pairing pushes it down and mismatched pairing pushes it up, as the simplified argument suggests. Both training seeds are shown, and the effect repeats in direction and in size.

Epoch	1	4	8	12	16	20	24
Seed 42							
correct (`warmup_only`)	0.160	0.137	0.132	0.130	0.130	0.130	0.129
mismatched (`shuffled_warmup`)	0.207	0.208	0.229	0.246	0.246	0.272	0.282
Seed 43							
correct (`warmup_only`)	0.165	0.139	0.132	0.130	0.130	0.130	0.128
mismatched (`shuffled_warmup`)	0.208	0.214	0.243	0.289	0.293	0.312	0.315

The mismatched run also lines up worse with its teacher target, since its teacher cosine ends the warmup at roughly a third of the correctly paired run's. It is being pulled somewhere, but it follows the given trajectory less closely. These measurements do not show what the optimal solution of our actual loss is.

An unplanned repeat. `full` and `warmup_only` also have identical settings before epoch 31 but were run as separate processes. So any difference between them in that period comes only from run-to-run nondeterminism, meaning small random differences in computation even with the same seed. The period covers 30 epochs, including the first six after the attack objective starts, and the two parts behave differently. While the attack term is off, the runs stay close, differing by a few percent at most and only in the first two epochs. Once the attack term starts, the difference grows several times over within two epochs, most of all on the specificity loss.

This contrast is the useful part, and it goes against the reassuring reading. The training loop is nearly reproducible while it only imitates and much less so once it attacks, which is the setting every reported number comes from. At seed 42 the pair came back together afterwards. At a second seed it did not. The same two programs split at epoch 26 and followed different paths (Table 14), so the caution was justified, and more than the first-seed numbers alone could show. (`imitation_only` cannot serve as a third copy. It runs 31 epochs, so its learning-rate schedule has a different length and its learning rate at a given epoch differs from the other two.)

R QUALITATIVE EXAMPLES

Figures 7 and 8 show the same two held-out images under every method. From a fixed set of images balanced across classes (selection seed 17092026), we take the first pair from the same class that ResNet-50 classifies correctly, without choosing for attack success or appearance. These examples only illustrate outputs and are not the measurement over the whole dataset, because visual variety cannot show specificity.

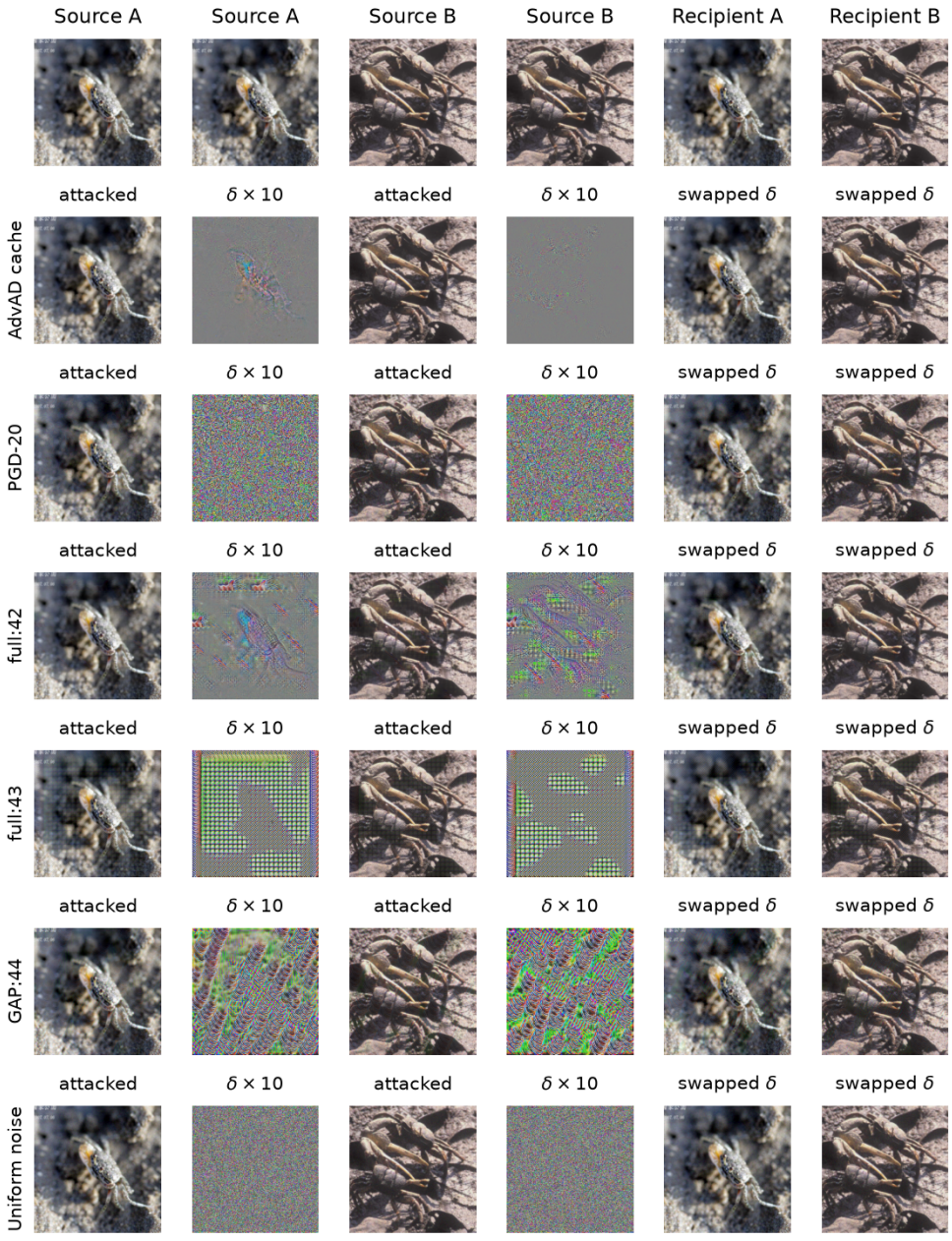


Figure 7: The teacher, an illustrative PGD, healthy and degraded students, teacher-free GAP and uniform noise, all on the same images. Every output uses an $8/255$ pixel budget. Perturbations are shown as neutral gray plus 10 times their pixel difference, on one fixed scale, and images are saved as lossless PNG. The right columns swap perturbations between these two same-class images. PGD here uses 20 margin steps, step size $2/255$ and no random start, so it is not the baseline of Table 2. The teacher output is loaded from the held-out set and projected back into the budget. Random noise is a visual reference, not a competing attack. The run labels come from measurements over the whole dataset, not from these two images.

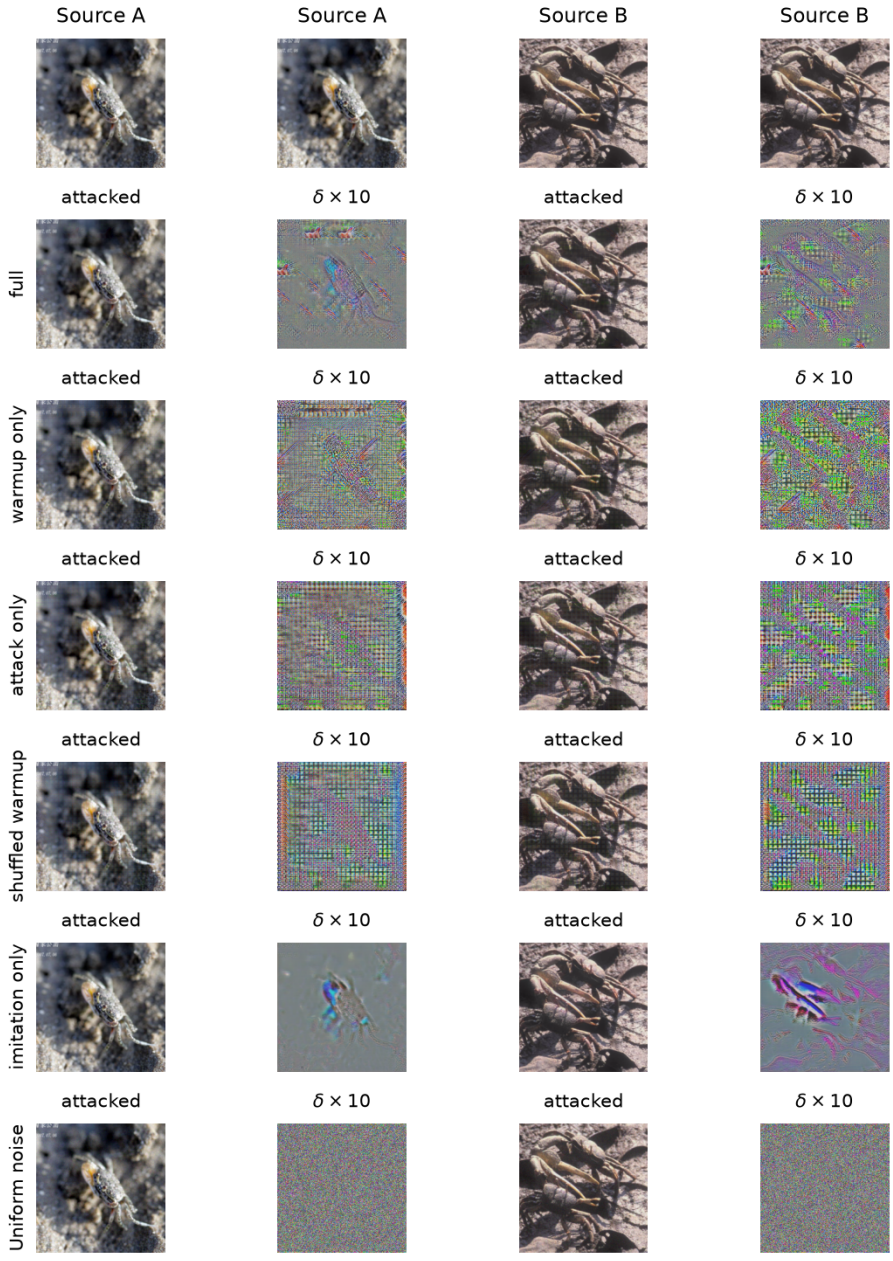


Figure 8: All five seed-42 training conditions on the same held-out images, plus uniform noise. Each row shows attacked images and perturbations magnified $10\times$, drawn the same way and with the same budget as Figure 7. All student outputs use their selected checkpoints and four refinement steps. The generators output perturbations, and no extra noise is added at attack time.