RELAY SAMPLING: CROSS-LABEL WARM STARTS FOR SHORT-RUN JOINT ENERGY MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep energy-based models commonly rely on short-run samplers from random noise for training and generation, but their finite-step distributions remain dependent on the initialization and may cover only a limited subset of the model distribution. In joint energy models, each label defines a conditional energy surface, yet the label is typically used only as the final generation target. We introduce *relay sampling*, which also uses label conditions to diversify the path from noise to the target class. The sampler first follows a random auxiliary label to construct a structured intermediate state, which we call a *cross-label warm start*, and then passes this state to the target-conditioned sampler. This procedure replaces the single target-only path from noise with a mixture of auxiliary-conditioned finite-step paths, and we hypothesize that training with negatives from this mixture yields a model that covers more of each class. On ImageNet 64×64, models trained with relayed negatives reach substantially higher recall and much better FID than models trained with target-only negatives. On ImageNet 256×256, relay training of a ConvNeXt-Large model, in pixel space and from random noise, yields an FID of 3.05, below diffusion and autoregressive baselines that use 1.6 to 3 times as many parameters, while retaining robust classification, OOD detection, and calibration.

1 INTRODUCTION

Energy-based models (EBMs) define a distribution through an unnormalized energy function, a flexible and conceptually simple generative framework. Training them is known to be difficult: the gradient of the log-partition function is an expectation under the model, so each update requires MCMC samples from the current energy, which is costly and can destabilize training (Du & Mordatch, 2019; Nijkamp et al., 2020; Grathwohl et al., 2020; 2021). Because running these chains to convergence is impractical, modern image EBMs rely on short-run sampling, a small fixed number of steps from uniform noise (Nijkamp et al., 2019) or from a replay buffer of past samples (Du & Mordatch, 2019; Grathwohl et al., 2020), for both training negatives and generation. In this setting, energy-based models remain far behind GANs, diffusion models, and autoregressive models on large-scale datasets such as ImageNet. Later gains came from leaving it: combining a latent space with a generator (Hill et al., 2022) or with diffusion training (Guo et al., 2023), training a sequence of energies along a diffusion process (Gao et al., 2021; Zhu et al., 2024), learning the transport from noise by flow matching (Balcerak et al., 2025), or initializing the sampler from a jointly trained generator (Xie et al., 2018; Geng et al., 2026) or real images (Yin et al., 2026); of these, only the latent-space diffusion hybrid and the real-image initialization reach ImageNet 256×256.

Short-run sampling changes the role of the sampler: since the chain is not run to equilibrium, the generated distribution is not the model distribution but one determined by the initialization distribution and the finite-step transition. When the chain starts from random noise, this finite-step distribution can have limited coverage (Du & Mordatch, 2019; Nijkamp et al., 2019): the variance of random noise is spread uniformly over all directions, of which only a few distinguish natural images, so the starting points are nearly identical in those directions, and the same finite-step transition carries them to a similar set of modes. A target-conditioned sampler from random noise may therefore produce sharp samples for a class while occupying only a restricted subset of the class's modes. Our experiments on ImageNet 64×64 confirm this for the short-run SGLD sampler: it reaches a precision similar to that of the ADM diffusion model (Dhariwal & Nichol, 2021) but a much lower recall (0.48 against 0.63; Table 1).

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 address the limited coverage of short-run sampling from random noise in joint energy-based models. In these models a classifier with logits $f_\theta(x)$ defines the joint energy $E_\theta(x, y) = -f_\theta(x)[y]$ (Grathwohl et al., 2020), so the sampler runs under an explicit label, usually treated only as the final generation target. We observe that the label can also be used earlier in the sampling path. Before committing to the target label, the sampler can briefly follow an auxiliary label condition, drawn at random for each sample, producing a structured state from noise; this state is then relayed to a target-conditioned sampler, which reshapes it under the target label. We call this procedure *relay sampling* and the state a *cross-label warm start*. Since the finite-step distribution of a short-run sampler retains its dependence on the initialization, we expect relay sampling to reach more modes of the target class than target-only sampling: cross-label warm starts, spread over the target surface by the random auxiliary labels, start the target stage where random noise does not; Figure 1 illustrates the intuition.

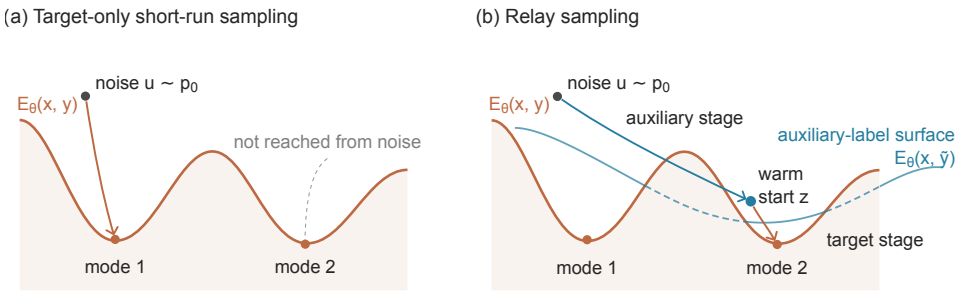


Figure 1: Schematic of target-only sampling and relay sampling on the energy surface of a target label y with two modes. (a) Target-only short-run sampling follows this surface from noise u and descends into mode 1; mode 2 is not reached. (b) Relay sampling first follows the surface of a random auxiliary label $\tilde{y}$, $E_\theta(x, \tilde{y})$, through the auxiliary stage to construct a cross-label warm start z, then relays z to the target surface $E_\theta(x, y)$, from which the target stage descends into mode 2.

During training, the sampler supplies the negatives; with its spread-out warm starts, relay sampling explores more of the energy surface, so contrastive training suppresses more false modes and shapes the energy of the target class more accurately. On ImageNet 64×64, models trained with relayed negatives reach a much better FID than models trained with target-only negatives, with substantially higher recall at a small cost in precision (Section 4.2). On ImageNet 256×256, a ConvNeXt-Large model trained with relay sampling, in pixel space and from uniform noise, reaches an FID of 3.05, below the ADM-G and LDM-4-G diffusion baselines (Dhariwal & Nichol, 2021; Rombach et al., 2022) and the autoregressive VAR-d16 (Tian et al., 2024), which use 1.6 to 3 times as many parameters (Section 4.5).

Our contributions are as follows.

- We propose relay sampling, formulate its finite-step distribution as a mixture of label-conditioned finite-step distributions (Section 3), and propose, for training with relayed negatives, an amortization algorithm that caches warm starts across labels, so that a relayed negative costs only its target-stage steps (Section 3.1).
- We validate relay training on ImageNet 64×64: it raises recall and improves FID over target-only training at a small cost in precision, and the allocation between the auxiliary and target stages acts as a training-time control over the coverage–fidelity trade-off (Section 4.2).
- We show that relay training scales to ImageNet 256×256, where an energy-based model trained and sampled from random noise in pixel space is, to our knowledge for the first time, competitive with diffusion and autoregressive generators, while retaining, as a joint model, robust classification, OOD detection, and calibration (Section 4.5).
- We isolate the two factors that determine the coverage of short-run sampling (Section 2.2) and compare approaches acting on each. Acting on the initialization is far more effective than adding noise to the transition: relay sampling and an informative Gaussian initialization, both with the deterministic PGD sampler, each raise coverage and improve FID much more than short-run SGLD does. Between the two, relay sampling reaches a much higher recall at 256×256, provides a training-time control over the coverage–fidelity trade-off, and needs no data statistics (Section 4.3).

2 BACKGROUND

2.1 JOINT ENERGY-BASED MODELS

A joint energy-based model (JEM) reinterprets a classifier as an energy-based model over (x, y) (Grathwohl et al., 2020). Let $f_\theta(x) \in \mathbb{R}^C$ denote the logits of a C-class classifier. JEM defines the joint energy

$$E_\theta(x, y) = -f_\theta(x)[y], \tag{1}$$

with the marginal image energy obtained by marginalizing over labels,

$$E_\theta(x) = -\log \sum_y \exp(f_\theta(x)[y]) = -\operatorname{LogSumExp}_y f_\theta(x)[y]. \tag{2}$$

JEM trains the joint model by approximately maximizing $\log p_\theta(x, y) = \log p_\theta(y \mid x) + \log p_\theta(x)$. The first term is optimized by the usual cross-entropy loss of the classifier. The second term is trained with contrastive divergence on the marginal image energy: for a data image x^+ and a negative image x^- drawn by the sampler of Section 2.2, the EBM term is

$$\mathcal{L}_{\text{EBM}} = E_\theta(x^+) - E_\theta(x^-) = \operatorname{LogSumExp}_{y'} f_\theta(x^-)[y'] - \operatorname{LogSumExp}_{y'} f_\theta(x^+)[y']. \tag{3}$$

2.2 SHORT-RUN CONDITIONAL SAMPLING

Training and sampling EBMs require approximate samples from the current energy model. A JEM-style model admits two samplers: an unconditional one on the marginal energy of Eq. (2), and a conditional one on $E_\theta(x, y)$ with a label y drawn from the label distribution. JEM reports both for training and adopts the unconditional sampler (Grathwohl et al., 2020); we use the conditional sampler for both training negatives and generation, since relay sampling acts on the label condition. A common choice of transition is the update of stochastic gradient Langevin dynamics (SGLD) (Welling & Teh, 2011),

$$x_{t+1} = x_t - \eta \nabla_x E_\theta(x_t, y) + \sigma \epsilon_t, \qquad \epsilon_t \sim \mathcal{N}(0, I), \tag{4}$$

where each iterate is clipped to the valid input range after the update. Because exact sampling is intractable, modern EBMs typically run such updates for a small, fixed number of steps without convergence (Nijkamp et al., 2019; Du & Mordatch, 2019; Grathwohl et al., 2020).

Target-only sampling. Let $S^T_{\theta,y}(x \mid u)$ denote the T-step transition kernel of this sampler under label y, and $x = S^T_{\theta,y}(u)$ a sample drawn from it. The standard procedure, which we call *target-only sampling*, draws $u \sim p_0$, typically uniform noise, and runs all T steps under the target label. Its finite-step distribution, the *target-only distribution*, is

$$q^T_{\text{target}}(x \mid y) = \int S^T_{\theta,y}(x \mid u)\, p_0(u)\, \mathrm{d}u, \tag{5}$$

Coverage of short-run sampling. Under standard regularity conditions, Langevin-type chains approach the conditional model distribution $p_\theta(x \mid y)$ in the limit of vanishing step size, with the noise scale matched to the step size and the chains run long enough to mix (Roberts & Tweedie, 1996; Welling & Teh, 2011). In that limit the kernel $S^T_{\theta,y}(x \mid u)$ no longer depends on u, and by Eq. (5) neither does the sampled distribution. Practical short-run samplers operate outside this regime, with a noise scale far below the one matched to the step size and a fixed small number of steps, typically between 5 and 100 (Nijkamp et al., 2019; Du & Mordatch, 2019; Grathwohl et al., 2020), so the kernel keeps its dependence on u and the sampler acts as a finite-step generator rather than as a source of equilibrium samples (Nijkamp et al., 2019; 2020). The regions of a class that the generator covers are therefore determined by the two factors of Eq. (5), the initialization distribution and the finite-step transition. For short-run target-only sampling from random noise, the starting points are nearly identical across draws in the directions that distinguish natural images, so the transition carries the chains to a similar subset of the class and q^T_{target} covers only part of it (Du & Mordatch, 2019; Nijkamp et al., 2019); Table 1 confirms this for the short-run SGLD sampler, whose recall is far below that of a diffusion model at similar precision.

3 RELAY SAMPLING

We propose relay sampling to improve the coverage of short-run conditional sampling. Of the two factors of Section 2.2, it acts on the initialization rather than the transition: the target stage starts not from random noise but from a state reached under a different label,

$$u \xrightarrow[\text{label } \tilde{y},\, T_1 \text{ steps}]{\text{auxiliary stage}} z \xrightarrow[\text{label } y,\, T_2 \text{ steps}]{\text{target stage}} x. \tag{6}$$

Given a target class y, the sampler draws noise $u \sim p_0$ and an auxiliary label $\tilde{y} \sim \pi(\tilde{y})$, typically uniform over classes, and follows the auxiliary conditional surface for T_1 steps, $z = S^{T_1}_{\theta,\tilde{y}}(u)$, where S is the finite-step transition of Section 2.2. We call the intermediate state z a *cross-label warm start*. The state is relayed to the target-conditioned sampler (Figure 1(b)), which reshapes it into the target class over the remaining T_2 steps, $x = S^{T_2}_{\theta,y}(z)$.

The relay distribution. Because each label indexes a different conditional surface and hence a different transition, target-conditioned sampling no longer begins from noise but from a mixture of structured warm-start distributions, one per auxiliary label, and the finite-step distribution of relay sampling is a mixture over auxiliary labels,

$$q^{T_1,T_2}_{\text{relay}}(x \mid y) = \sum_{\tilde{y}} \pi(\tilde{y}) \iint S^{T_2}_{\theta,y}(x \mid z)\, S^{T_1}_{\theta,\tilde{y}}(z \mid u)\, p_0(u)\, \mathrm{d}u\, \mathrm{d}z. \tag{7}$$

We hypothesize that training with negatives drawn from this mixture yields a model whose short-run samples cover more of each class than training with negatives from q^{T}_{target} of Eq. (5), since it lets training discover and suppress more false modes and shape the energy of the target class more accurately. The allocation (T_1, T_2), a free parameter of the procedure, should then act as a training-time control: a longer auxiliary stage spreads the warm starts more widely over the target surface and should increase coverage, and a longer target stage gives the chain more steps to reshape them into the target class and should increase fidelity. We test both in Section 4.2.

3.1 TRAINING WITH RELAY SAMPLING

Training an EBM requires a batch of negative samples at every optimization step (Eq. (3)). The target stage of a relayed path starts from a warm start that the auxiliary stage produces from noise, so relay sampling depends on the model generating valid warm starts from noise. Training therefore uses negatives from both target-only runs and relayed paths, so that the model learns to generate from both random noise and structured warm starts. We draw each negative from the mixture

$$q_{\text{neg}}(x \mid y) = \rho\, q^{T}_{\text{target}}(x \mid y) + (1-\rho)\, q^{T_1,T_2}_{\text{relay}}(x \mid y), \tag{8}$$

that is, a target-only run of T steps (Eq. (5)) with probability ρ and a relayed path (Eq. (7)) otherwise.

Amortization. Each sampler step is a gradient evaluation of f_θ with respect to the input, and these steps dominate the cost of training; under Eq. (8) a relayed negative takes $T_1 + T_2$ of them, against T for a target-only one. We observe that the T_1 auxiliary steps of the relayed branch need not be run separately: a target-only run of $T \geq T_1$ steps under its training label passes through $S^{T_1}_{\theta,y}(u)$ at step T_1, which is exactly the warm start used by the relayed branch. We therefore record this intermediate state in a cache $\mathcal{C}$ of image states and reuse it as a warm start for target-conditioned sampling under other labels, while the target-only run continues to its endpoint, which serves as the target-only negative. Algorithm 1 summarizes the procedure. It approximates Eq. (8) in two respects: the auxiliary label of a cached state is the training label that spawned it, so the training-label distribution takes the place of π, and a cached state was produced by parameters earlier than those that relay it, with a lag set by the cache size and the insertion rate ρ.

Sampling cost. Without the warm-start cache the expected cost per negative is $\rho\, T + (1-\rho)(T_1+T_2)$ sampler steps; with it the auxiliary steps are shared, and the cost drops to $\rho\, T + (1-\rho)\, T_2$, a saving of $(1-\rho)\, T_1$ per negative, while a relayed negative remains $T_1 + T_2$ steps from noise. For the relay configuration of our ImageNet 64×64 controlled comparison (Table 1; $\rho = 0.2$, $T = 150$, $T_1 = 120$, $T_2 = 60$), this is 78 steps against 174, a 2.2-fold reduction in sampler steps per negative.

Relation to standard replay buffers. Replay buffers are the standard mechanism for reducing negative-sampling cost in EBM training (Hinton, 2002; Tieleman, 2008; Du & Mordatch, 2019). A replay update draws a state from the buffer, applies a small number of sampler steps, uses the result as a negative, and reinserts it, so the buffer stores the state needed to continue a persistent Markov chain and the replayed samples are interpreted as approximate states of such chains. For class-conditional sampling in JEM, the chain runs under $E_\theta(x, y)$, so the natural implementation stores the joint state (x, y) and replays each image only under the label that produced it. The warm-start cache differs on both counts. It never reinserts sampler outputs: only states of target-only runs enter, so every entry is exactly T_1 steps from noise and no chain identity persists. And it stores only the image state, which is reused under other labels rather than the label that produced it. The cache is therefore a pool of noise-anchored cross-label warm starts, not a set of persistent chains. The difference also sets where each mechanism applies: a persistent chain is replayed under the condition that produced it, so the buffer applies only where conditions repeat, as class labels do; a cross-label warm start is consumed under another condition by design, so the cache also applies where conditions do not repeat, as with free-form text conditions.

Algorithm 1 Amortized relay training with a warm-start cache

Require: model f_θ, cache $\mathcal{C}$, minibatch $\{(x_i^+, y_i)\}_{i=1}^B$, noise distribution p_0, target-only run length T, auxiliary-stage length $T_1 \le T$, target-stage length T_2, target-only probability ρ

for $i = 1, \dots, B$ **do**
  Sample $r_i \sim \mathrm{Uniform}(0, 1)$
  if $r_i < \rho$ or $\mathcal{C}$ is empty **then**
    Sample fresh noise $u_i \sim p_0$
    Run target-only sampling from noise for T steps, $x_i^- \leftarrow S_{\theta,y_i}^{T}(u_i)$, recording the state after T_1 steps, $z_i \leftarrow S_{\theta,y_i}^{T_1}(u_i)$
    Insert z_i into $\mathcal{C}$, replacing the oldest entry if $\mathcal{C}$ is full
  else
    Sample z_i uniformly from $\mathcal{C}$
    Relay z_i to the current-label sampler: $x_i^- \leftarrow S_{\theta,y_i}^{T_2}(z_i)$; do **not** insert x_i^- into $\mathcal{C}$
  end if
  Use x_i^- in the generative loss
end for
Update θ using the supervised loss and the generative loss

4 Experiments

4.1 Setup

We conduct experiments on ImageNet 64×64 and 256×256 (Deng et al., 2009) with ConvNeXt models (Liu et al., 2022), ConvNeXt-Base at 64×64 for analysis of the method and ConvNeXt-Large at 256×256 for comparison with other generative models. We investigate two training procedures: target-only training, which draws every negative from noise, and relay training, which draws negatives by Algorithm 1 from the mixture of Eq. (8) with $\rho = 0.2$; $\rho = 0.4$ is slightly worse (Table 4).

Training recipe. Training is initialized from the adversarially trained robust ConvNeXt classifiers of Singh et al. (2023) and otherwise follows DAT (Yin et al., 2026): the classification branch continues adversarial training throughout to regularize the input gradient of the energy, and the generative term is a bounded binary cross-entropy loss between data images and generated negatives in place of the contrastive term of Eq. (3). Training hyperparameters are listed in Appendix A.1.

Sampler. Unless stated otherwise, negatives start from p_0, uniform noise over the input range. We consider two short-run samplers: the stochastic SGLD update of Eq. (4), with settings in Appendix A.1.4, and the deterministic projected gradient descent (PGD) sampler of DAT (Yin et al., 2026), which sets $\sigma = 0$ and replaces the gradient by its normalized direction, $x_{t+1} = x_t - \eta \nabla_x E_\theta(x_t, y)/\|\nabla_x E_\theta(x_t, y)\|_2$ (step sizes in Tables 3 and 4). We use PGD as the default sampler, since it injects no noise and thus isolates the effect of the initialization on coverage.

Test-time sampling schedule. For each training run, we sweep the test-time schedule and step size on every checkpoint and select the checkpoint and configuration with the lowest FID on 10K samples jointly; the sweep ranges are given in Appendix A.1.1 and the selections in Table 4. No test-time cache is used.

Evaluation metrics. All reported metrics are computed at the selected checkpoint and schedule from 50K generated samples with the same number of samples per class, with the ADM evaluation code (Dhariwal & Nichol, 2021) at 256×256 and with pytorch-fid and torch-fidelity at 64×64: FID (Heusel et al., 2017), Inception Score, and precision and recall (Kynkäänniemi et al., 2019), which measure fidelity and coverage respectively.

4.2 Relay Training versus Target-Only Training

Table 1: Target-only versus relay training on ImageNet 64; schedules in Table 4 (marked †). ADM from Dhariwal & Nichol (2021).

Training	FID ↓	Prec. ↑	Rec. ↑
Target-only			
Uniform + PGD	4.20	0.75	0.43
Uniform + SGLD	3.58	0.73	0.48
Info. Gaussian + PGD	2.35	0.73	0.56
Relay			
Uniform + PGD (89M)	2.40	0.72	0.58
ADM (296M)	2.07	0.74	0.63

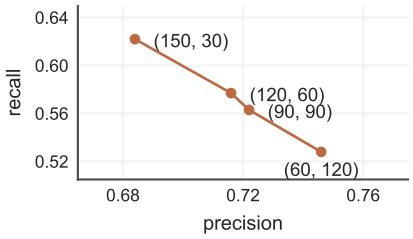


Figure 2: Training-time stage allocation: precision and recall of one model per split at $T = 150$, $T_1 + T_2 = 180$, labeled by (T_1, T_2).

Limited coverage of target-only sampling. The two target-only models of Table 1, trained from uniform noise with PGD and with short-run SGLD, reach a precision close to that of the ADM diffusion model but a recall far below it, and the stochastic sampler closes only a small part of the gap; their samples in Figure 7 are correspondingly confined to a few settings per class.

Relay training. We ask whether a model trained with relayed negatives covers more of the data than a model trained with target-only negatives, and at what cost in precision. The two uniform-noise PGD rows of Table 1 compare relay training at $(T;\, T_1, T_2) = (150;\, 120, 60)$ with target-only training at $T = 150$ under the same update rule, so the two differ only in whether negatives start from cross-label warm starts or from noise. Relay training raises recall and improves FID at a small cost in precision: FID improves from 4.20 to 2.40 and recall rises from 0.43 to 0.58, with precision at 0.72 against 0.75. Figure 7 shows samples from the four models of Table 1: the relay-trained model produces visibly more varied samples than either target-only model.

Relay allocation. Section 3 also predicts the direction of the allocation: increasing the auxiliary stage T_1 should raise coverage, and increasing the target stage T_2 should raise fidelity. We train one model for each of four splits of a 180-step relayed path, $(T;\, T_1, T_2) = (150;\, 150, 30)$, $(150;\, 120, 60)$, $(150;\, 90, 90)$, and $(150;\, 60, 120)$, all other settings fixed, and evaluate precision and recall (Figure 2, Table 4). Both move monotonically in the predicted directions, so the allocation acts as a training-time control over the coverage–fidelity trade-off.

4.3 Initialization versus Transition for Coverage

Relay sampling is one way to raise the coverage of a short-run sampler, and we compare it with the alternatives. Coverage is determined by the initialization distribution and the finite-step transition (Section 2.2), and each factor suggests one alternative: on the initialization side, an informative initialization that already spreads across the data, and on the transition side, a stochastic sampler that adds noise. We train one target-only model for each, from an informative Gaussian initialization fitted to the training images with PGD (Appendix A.1.3) and from uniform noise with short-run SGLD (Appendix A.1.4), and compare them with the uniform-noise PGD and relay models in Table 1.

Relay sampling and the informative initialization reach similar coverage and fidelity, the Gaussian with a slightly better FID and precision and relay sampling with a slightly higher recall. SGLD improves on the uniform-noise PGD model by much less than either, recovering about a third of their gain in recall and in FID at similar precision. In the short-run regime, acting on the initialization is thus more effective than adding noise to the transition. Relay sampling and the Gaussian also bring the EBM close to the diffusion model of Table 1: FIDs of 2.35 and 2.40 against 2.07 for ADM, with less than a third of its parameters.

The two initialization-side approaches are close at this resolution and differ in the following respects.

- *Coverage.* Relay training reaches a higher recall at 64×64 (0.58 against 0.56, Table 1) and a much higher one at 256×256 (0.53 against 0.45, Table 2, Figures 8 and 9), so it addresses coverage more directly.
- *Control.* The stage allocation is a training-time control over the coverage–fidelity trade-off (Figure 2); the Gaussian has no corresponding mechanism.
- *No data statistics.* Relay sampling needs no estimation pass over the data and no stored statistics, so training is self-contained and generation needs the model alone.
- *Cost.* The nominal sampler cost of relay training is more than three times that of the Gaussian, $T_1 + T_2 = 158$ steps per negative against $T = 46$ at 256×256 (Table 4), but with amortized training the expected cost per negative, $\rho T + (1 - \rho)T_2$ (Section 3.1), is about 67 steps against 46, about 1.5 times as many. At test time the selected schedules are (36, 25) steps for relay training and 20 for the Gaussian (Table 4); at deployment the auxiliary stage can be precomputed and stored, after which a sample costs its target stage of 25 steps against 20.

The initialization used in training also determines what the energy can do in a downstream task: by short-run descent alone, the two models trained from uniform noise remove most of a salt-and-pepper corruption, whereas the Gaussian model removes much less of it (Appendix A.3).

4.4 Sampling Schedule Ablations

Training with relayed negatives (Eq. (8)) involves three step-count hyperparameters: the target-only run length T and the allocation (T_1, T_2) of the relayed path. To examine their effect, we train a model on ImageNet 64×64 for several schedules $(T;\, T_1, T_2)$, holding all other hyperparameters the same, and evaluate each checkpoint under the protocol of Section 4.1.

Training stability. We find that these step counts affect training stability: stable training requires the relayed path length $T_1 + T_2$ to exceed the target-only run length T by a bounded factor. In a stable run, FID decreases throughout training and then plateaus; in a collapsed run, FID improves at first, then the generated samples develop high-frequency artifacts and FID degrades. With $T_1 + T_2$ from $1.2\,T$ to $1.4\,T$, all ten runs train stably; outside this band the outcomes are mixed (Figure 3(a), Table 5). The band also held at 256×256: the ConvNeXt-Large model of Section 4.5 is trained at $T_1 + T_2 = 1.25\,T$ and trains stably. A possible explanation for the instability outside the band is a mismatch between the two kinds of negatives: at $T_1 + T_2 = T$ the relayed negatives, having spent T_1 steps under the auxiliary label, are weaker than the fresh ones, and at $1.5\,T$ and beyond they are stronger; in either case one population dominates the contrastive gradient and can cause instability.

Stage allocation. We find that the allocation between T_1 and T_2 affects FID strongly at a fixed budget. At $T = 150$ and $T_1 + T_2 = 180$, FID is minimized at the interior allocation $(T_1, T_2) = (120, 60)$, which assigns two thirds of the relayed path to T_1, and degrades toward both extremes, spanning 2.40 to 2.86 across the sweep (Figure 3(b)). The precision–recall trade-off of Figure 2 explains the interior optimum: T_1 raises coverage and T_2 raises fidelity, so precision and recall move in opposite directions across the same splits, and FID, sensitive to both, is lowest where they balance.

Schedule depth. We next ask how much the depth of the schedule matters. We compare three schedules, $(100;\, 80, 40)$, $(120;\, 100, 50)$, and $(150;\, 120, 60)$, which share $(T_1 + T_2)/T \approx 1.2$ and the allocation ratio $T_1/(T_1 + T_2) = 0.67$, and differ in their length (Figure 3(c)). FID improves from 5.18 at $T = 100$ through 3.03 at $T = 120$ to 2.40 at $T = 150$, so too shallow a schedule degrades quality sharply and the gains from further depth diminish toward saturation.

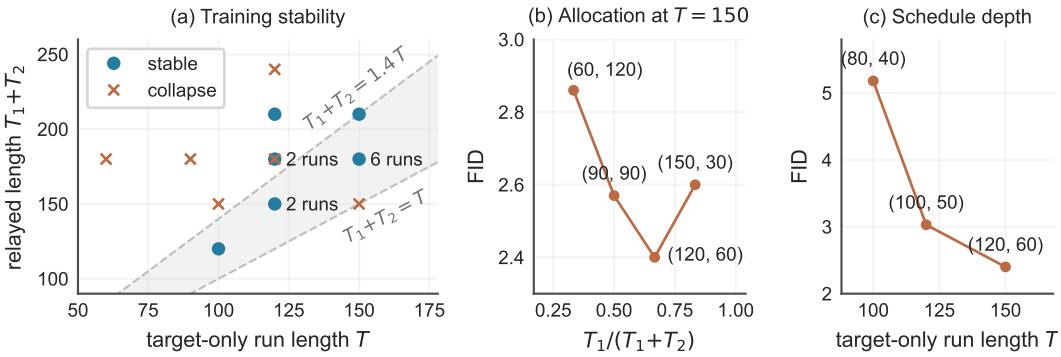


Figure 3: Effect of the sampling schedule on ImageNet 64×64. (a) Training stability of all completed runs as a function of T and T_1+T_2: every run in the shaded band between the dashed lines trains stably while outcomes outside it are mixed; Table 5 lists the allocations. (b) FID across allocations at $T=150$, $T_1+T_2=180$. (c) FID under proportional scaling of the schedule at $T_1/(T_1+T_2)=0.67$. In (b) and (c), point labels give (T_1, T_2).

4.5 Scaling to ImageNet 256×256

We apply relay training at scale, to a ConvNeXt-Large model on ImageNet 256×256, with the allocation $T_1/(T_1+T_2)=0.67$ that Figure 3(b) found best and $T_1+T_2=1.25\,T$; the step counts themselves are set as described in Appendix A.1.2 and listed in Table 4. Table 2 places the resulting model among other class-conditional generative models. With 198M parameters and an FID of 3.05, the model improves on the early representative of every family (BigGAN-deep, VQGAN, MaskGIT, ADM-G, LDM-4-G, and EGC) and is competitive with later ones: it improves on VAR-d16, GigaGAN, and LlamaGen-L, and falls short of StyleGAN-XL, DiT-XL/2-G, and MAR-B. Figure 8 shows uncurated samples of the model.

To our knowledge, our model is the first EBM trained and sampled from random noise in pixel space at this resolution, and the first such model to be competitive with the other families in Table 2: prior results in this setting stop at 128×128, with an FID of 43.7 (Du & Mordatch, 2019). The

Table 2: Class-conditional generation on ImageNet 256×256. Baseline numbers are from the source papers (DAT's precision and recall from its released checkpoint).

Method	Params	Steps	FID ↓	IS ↑	Prec. ↑	Rec. ↑
GAN						
BigGAN-deep (Brock et al., 2019)	340M	1	6.95	203.6	0.87	0.28
StyleGAN-XL (Sauer et al., 2022)	166M	1	2.30	265.1	0.78	0.53
GigaGAN (Kang et al., 2023)	569M	1	3.45	225.5	0.84	0.61
Autoregressive and masked-token						
VQGAN (Esser et al., 2021)	1.4B	256	15.78	74.3	–	–
MaskGIT (Chang et al., 2022)	227M	8	6.18	182.1	0.80	0.51
LlamaGen-L (Sun et al., 2024)	343M	256	3.80	248.3	0.83	0.52
VAR-d16 (Tian et al., 2024)	310M	10	3.30	274.4	0.84	0.51
MAR-B (Li et al., 2024)	208M	256	2.31	281.7	0.82	0.57
Diffusion						
ADM-G (Dhariwal & Nichol, 2021)	608M	250	4.59	186.7	0.82	0.52
LDM-4-G (Rombach et al., 2022)	400M	250	3.60	247.7	0.87	0.48
DiT-XL/2-G (Peebles & Xie, 2023)	675M	250	2.27	278.2	0.83	0.57
Energy-based						
EGC (Guo et al., 2023)	543M	1000	6.05	231.3	–	–
DAT (ConvNeXt-L) (Yin et al., 2026)	198M	36	3.29	310.2	0.79	0.48
Informative Gaussian + PGD (ConvNeXt-L, ours)	198M	20	3.21	297.0	0.83	0.45
Relay sampling + PGD (ConvNeXt-L, ours)	198M	61 (36+25)	3.05	287.7	0.79	0.53

other two energy-based entries reach 256×256 by leaving this setting, and both have a worse FID: DAT keeps the pixel-space energy and the sampler-based training but starts its sampler from real out-of-distribution images rather than noise, for both training negatives and generation; EGC trains a diffusion-time-conditioned energy by score matching in the latent space of a pretrained autoencoder and samples by reverse diffusion. The informative-Gaussian initialization of Section 4.3, trained at this scale, reaches an FID of 3.21 against 3.05 and a recall of 0.45 against 0.53, so its samples cover less of the data, as the sample grids of Appendix A.5 also show.

We also evaluate the model on robust classification, OOD detection, and calibration (Appendix A.2). Like DAT, relay training preserves the robust classification of the classifier it starts from, at some cost in clean accuracy; the model also keeps most of its OOD detection ability and is much better calibrated. The informative-Gaussian model of Section 4.3, trained at this scale, performs similarly on all three tasks.

5 Related Work

Maximum-likelihood training of deep EBMs relies on approximate negative samples, obtained classically by contrastive divergence from data initializations (Hinton, 2002), by persistent chains (Tieleman, 2008), or by short-run MCMC from noise (Nijkamp et al., 2019; 2020). Replay buffers that store and continue previous chains are the standard amortization mechanism (Du & Mordatch, 2019; Grathwohl et al., 2020).

Beyond these MCMC-based schemes, a second line uses an auxiliary generator, either to replace MCMC altogether (Grathwohl et al., 2021) or to supply it with structured initial states, by cooperative learning (Xie et al., 2018; Geng et al., 2026), the divergence triangle (Han et al., 2019), or maximum-entropy generators (Kumar et al., 2019), or from a pretrained VAE (Xiao et al., 2021); informative initialization instead anchors the initial states to the data, drawing them from class-conditional Gaussians estimated on the training set (Yang & Ji, 2021). Relay sampling also replaces noise initialization with structured states, but draws them from the same conditional energy model under a different label, so no auxiliary generator is trained and no data statistics are estimated for the initialization. Cross-label states also appear in a test-time heuristic for sample diversity mentioned by Du & Mordatch (2019): at each round except the final ones, their sampler replaces the label of a long persistent chain by a random one with a small probability. Relay sampling instead makes one switch at a fixed step of a short-run chain from noise and is used for training rather than only at test time.

6 Conclusion

We introduced relay sampling, which uses the label condition of a JEM to diversify the initialization of short-run sampling: the sampler leaves noise under an auxiliary label and is relayed to the target label. As hypothesized, training with such warm starts as negatives covers more of each class: on ImageNet 64×64, relay training raises recall and improves FID over target-only training, and the stage allocation acts as a training-time control over the coverage–fidelity trade-off. At scale, relay training reaches an FID of 3.05 on ImageNet 256×256, a level not previously reported for EBMs trained and sampled from random noise in pixel space. With a suitable training recipe, EBMs thus train stably and are competitive with other generative models.

Beyond the method, our experiments show that the coverage of short-run sampling from noise is improved more effectively at the initialization than at the transition: relay sampling and a data-fitted full-covariance Gaussian, which spread the starting points by different means, raise coverage to nearly the same degree at 64×64, where both are competitive with a diffusion model at a third of the parameters, whereas adding noise to the transition gains much less. At 256×256, where only a low-rank Gaussian can be estimated, relay sampling reaches a much higher recall; it also needs no data statistics and can be advantageous in certain downstream tasks.

Limitations. Relay sampling needs a label to relay through, so it does not apply to unconditional models, and its warm starts are drawn from the other classes of the same model, so we expect the gain to shrink with few classes. All models follow DAT's training recipe, which regularizes the energy through adversarial training of the classification branch; transfer beyond this recipe and to conditions beyond class labels, such as text prompts, remains open.

AI USE STATEMENT

In this work, we used generative AI tools to implement methods and experiment code, to write data-preprocessing scripts, to provide feedback on the experimental design and on the framing of the claims, and to assist in interpreting and presenting results. We have not used generative AI tools to generate data, to formulate or prove mathematical claims, or to perform qualitative data analysis; the remaining tasks in the required-disclosure list are not applicable. Additionally, we used generative AI tools to draft and edit text, to write LaTeX and figure scripts and produce the plots, and to write scripts that launch and monitor experiments and collect their results. We have reviewed all AI-assisted work: code was verified by the authors, and all claims and interpretations were checked against the experimental results by the authors. We take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

The training procedure is specified in Section 3.1 and Algorithm 1, and the experimental setup, sampler, and selection protocol in Section 4.1. Appendix A.1 lists all training hyperparameters (Table 3) and every training run with its schedules and metrics (Table 4), and specifies the test-time sweep (Appendix A.1.1), the step schedule at 256×256 (Appendix A.1.2), the construction of the informative-Gaussian initialization (Appendix A.1.3), and the SGLD sampler (Appendix A.1.4); Appendix A.2 specifies the classification, calibration, and OOD protocols and Appendix A.3 the denoising protocol. The DAT numbers not reported in its paper are computed from its released checkpoint. The code and checkpoints are provided as supplementary material and will be released publicly upon publication.

REFERENCES

Michal Balcerak, Tamaz Amiranashvili, Antonio Terpin, Suprosanna Shit, Lea Bogensperger, Sebastian Kaltenbach, Petros Koumoutsakos, and Bjoern Menze. Energy matching: Unifying flow matching and energy-based models for generative modeling. In *Advances in Neural Information Processing Systems*, 2025.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. MaskGIT: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

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

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Yilun Du and Igor Mordatch. Implicit generation and modeling with energy-based models. In *Advances in Neural Information Processing Systems*, 2019.

Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P. Kingma. Learning energy-based models by diffusion recovery likelihood. In *International Conference on Learning Representations*, 2021.

Cong Geng, Xue Han, Ye Yuan, Qiang Hu, Xin Huang, Ruiqiao Bai, and Junlan Feng. Latent-guided cooperative energy-based models. In *International Conference on Machine Learning*, 2026.

Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *International Conference on Learning Representations*, 2020.

Will Grathwohl, Jacob Kelly, Milad Hashemi, Mohammad Norouzi, Kevin Swersky, and David Duvenaud. No MCMC for me: Amortized sampling for fast and stable training of energy-based models. In *International Conference on Learning Representations*, 2021.

Qiushan Guo, Chuofan Ma, Yi Jiang, Zehuan Yuan, Yizhou Yu, and Ping Luo. EGC: Image generation and classification via a diffusion energy-based model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22952–22962, 2023.

Tian Han, Erik Nijkamp, Xiaolin Fang, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Divergence triangle for joint training of generator model, energy-based model, and inferential model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

Mitch Hill, Erik Nijkamp, Jonathan Mitchell, Bo Pang, and Song-Chun Zhu. Learning probabilistic models from generator latent spaces with Hat EBM. *Advances in Neural Information Processing Systems*, 35:928–940, 2022.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10124–10134, 2023.

Rithesh Kumar, Sherjil Ozair, Anirudh Goyal, Aaron Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models. *arXiv preprint arXiv:1901.08508*, 2019.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, 2019.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. In *Advances in Neural Information Processing Systems*, 2024.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986, 2022.

Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent non-persistent short-run MCMC toward energy-based model. In *Advances in Neural Information Processing Systems*, 2019.

Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. On the anatomy of MCMC-based maximum likelihood learning of energy-based models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Axel Sauer, Katja Schwarz, and Andreas Geiger. StyleGAN-XL: Scaling StyleGAN to large diverse datasets. In *ACM SIGGRAPH 2022 Conference Proceedings*, 2022.

Naman D. Singh, Francesco Croce, and Matthias Hein. Revisiting adversarial training for ImageNet: Architectures, training and generalization across threat models. In *Advances in Neural Information Processing Systems*, 2023.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Tijmen Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 1064–1071, 2008.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *International Conference on Machine Learning*, 2011.

Zhisheng Xiao, Karsten Kreis, Jan Kautz, and Arash Vahdat. VAEBM: A symbiosis between variational autoencoders and energy-based models. In *International Conference on Learning Representations*, 2021.

Jianwen Xie, Yang Lu, Ruiqi Gao, and Ying Nian Wu. Cooperative learning of energy-based model and latent variable model via MCMC teaching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Xiulong Yang and Shihao Ji. JEM++: Improved techniques for training JEM. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Xuwang Yin, Claire Zhang, Julie Steele, Nir Shavit, and Tony T. Wang. Scalable energy-based models via adversarial training: Unifying discrimination and generation. In *International Conference on Learning Representations*, 2026.

Yaxuan Zhu, Jianwen Xie, Ying Nian Wu, and Ruiqi Gao. Learning energy-based models by cooperative diffusion recovery likelihood. In *International Conference on Learning Representations*, 2024.

A Experimental Details and Additional Results

A.1 Training Setup, Hyperparameters, and Runs

Table 3 lists the training hyperparameters of the two models, Table 4 lists every evaluated training run with its training length, the test-time steps selected by the protocol of Section 4.1, and its metrics, and Table 5 lists all runs of the stability study with their outcomes.

Table 3: Training hyperparameters of the default configuration, uniform-noise p_0 with the PGD sampler, at 64×64 and 256×256. The informative-Gaussian and SGLD variants are described in Appendix A.1.3 and Appendix A.1.4; test-time step sizes are in Table 4.

	ImageNet 64×64	ImageNet 256×256
Backbone	ConvNeXt-Base	ConvNeXt-Large
Optimizer, learning rate, batch size	AdamW, 2×10^{-4}, 200	AdamW, 2×10^{-4}, 200
Learning-rate schedule	constant	constant
Weight decay	0.005	0.005
Training length (images seen)	3.5M	2.5M
Initialization p_0	uniform noise over $[0, 1]$	uniform noise over $[0, 1]$
Sampler step size η (training)	0.5	3
Cache size $\lvert\mathcal{C}\rvert$	20,000	20,000
Target-only probability ρ	0.2	0.2
Classifier AT: norm, budget ϵ_{clf}, steps, step size α_{clf}	ℓ_2, 1.875, 6, 0.5	ℓ_2, 7.5, 6, 2

Table 4: Evaluated training runs on ImageNet 64×64 (ConvNeXt-Base) and ImageNet 256×256 (ConvNeXt-Large). The first column gives the training procedure and initialization; the sampler is PGD and $\rho = 0.2$ unless stated. The training schedule is T for target-only and $(T;\, T_1, T_2)$ for relay training, at 256×256 the values at the selected checkpoint (Appendix A.1.2); images seen is the training length at the selected checkpoint (total training lengths in Table 3); the test schedule and test step size η are the configuration selected by the protocol of Section 4.1 (Appendix A.1.1; for SGLD, η scales the unnormalized gradient and the noise scale is in Table 6). Metrics are reported at the selected checkpoint; rows marked † are the runs of Table 1.

Training	Training schedule	Seed	Images seen	Test schedule	Test η	FID ↓	Prec. ↑	Rec. ↑
ImageNet 64 × 64								
Target-only, uniform†	T=150	1	2.3M	T=156	0.5	4.20	0.75	0.43
Target-only, info. Gaussian	T=40	1	3.2M	T=22	1.0	3.18	0.70	0.59
Target-only, info. Gaussian†	T=50	1	3.2M	T=28	1.0	2.35	0.73	0.56
Target-only, info. Gaussian	T=60	1	2.2M	T=31	1.0	2.39	0.75	0.54
Target-only, uniform, SGLD†	T=150	1	3.1M	T=174	3.0	3.58	0.73	0.48
Relay, uniform	(100; 80, 40)	1	1.0M	(20, 16)	2.0	5.18	0.63	0.60
Relay, uniform	(120; 100, 50)	1	2.4M	(24, 20)	2.0	3.03	0.69	0.59
Relay, uniform	(150; 60, 120)	1	3.5M	(18, 34)	2.0	2.86	0.75	0.53
Relay, uniform	(150; 90, 90)	1	3.5M	(24, 28)	2.0	2.57	0.72	0.56
Relay, uniform†	(150; 120, 60)	1	3.5M	(28, 24)	2.0	2.40	0.72	0.58
Relay, uniform	(150; 120, 60)	0	3.5M	(30, 22)	2.0	2.44	0.72	0.58
Relay, uniform, ρ=0.4	(150; 120, 60)	1	3.5M	(28, 22)	2.0	2.66	0.73	0.56
Relay, uniform	(150; 150, 30)	1	2.2M	(36, 16)	2.0	2.60	0.68	0.62
ImageNet 256 × 256								
Target-only, info. Gaussian	T=46	1	1.0M	T=20	8.0	3.21	0.83	0.45
Relay, uniform	(125; 105, 53)	1	2.1M	(36, 25)	8.0	3.05	0.79	0.53

Table 5: Stability runs of Figure 3(a) on ImageNet 64×64. Images seen is the length at which each run was stopped. Stable denotes no observed collapse over that length; for collapsed runs, the entry gives the training length after which FID began to deteriorate, in all cases within 1M images.

Training schedule $(T;\, T_1, T_2)$	Seed	$(T_1{+}T_2)/T$	Images seen	Outcome
$(60;\, 60, 120)$	0	3.00	0.4M	collapse (after 0.3M)
$(90;\, 90, 90)$	0	2.00	0.5M	collapse (after 0.2M)
$(100;\, 80, 40)$	1	1.20	3.5M	stable
$(100;\, 100, 50)$	0	1.50	0.6M	collapse (after 0.4M)
$(120;\, 100, 50)$	0	1.25	1.3M	stable
$(120;\, 100, 50)$	1	1.25	3.5M	stable
$(120;\, 120, 60)$	0	1.50	2.6M	stable
$(120;\, 120, 60)$	1	1.50	1.0M	collapse (after 0.7M)
$(120;\, 120, 90)$	0	1.75	2.6M	stable
$(120;\, 120, 120)$	0	2.00	0.5M	collapse (after 0.2M)
$(150;\, 100, 50)$	1	1.00	1.0M	collapse (after 0.8M)
$(150;\, 150, 30)$	0	1.20	2.6M	stable
$(150;\, 150, 30)$	1	1.20	3.5M	stable
$(150;\, 120, 60)$	0	1.20	3.5M	stable
$(150;\, 120, 60)$	1	1.20	3.5M	stable
$(150;\, 90, 90)$	1	1.20	3.5M	stable
$(150;\, 60, 120)$	1	1.20	3.5M	stable
$(150;\, 150, 60)$	0	1.40	2.6M	stable

A.1.1 Test-time schedule sweep

The sweep of Section 4.1 runs over the step count T for target-only models and the stage lengths (T_1, T_2) for relay models, in both cases jointly with the step size, over $\eta \in \{0.5, 1, 2\}$ at 64×64 and $\eta \in \{2, 4, 8\}$ at 256×256; Table 4 lists the selections. The SGLD model is outside this search: it keeps the step size $\eta = 3$ of its unnormalized gradient and the noise scale of its training (Appendix A.1.4) and is swept over T only.

A.1.2 Training step schedule at 256×256

Because a schedule sweep as in Section 4.4 is costly at this resolution, we adjust the step counts during training rather than fixing them: we hold the allocation at the ratio $T{:}T_1{:}T_2 = 120{:}100{:}50$ and raise T whenever the AUC of the negative energy between the real and the generated images exceeds 0.3, so that the negatives keep a fixed difficulty as the model improves; the threshold is slightly below the value of about 0.4 at which the relay model of Table 1 settles late in training. Both 256×256 models, relay training and the informative-Gaussian initialization, use this controller with the same threshold, so their step counts are set by the same rule. The relay model is at $(T;\, T_1, T_2) = (125;\, 105, 53)$ at its selected checkpoint and the Gaussian model at $T = 46$.

A.1.3 Informative Gaussian initialization

The informative initialization of Section 4.3 follows Yang & Ji (2021), with one class-independent Gaussian in place of their per-class estimates, fitted to the ImageNet training set and kept fixed throughout training. Both estimates use the training images resized to a short edge of 256 and center-cropped, downsampled to 64×64 for the 64×64 models, and scaled to $[0, 1]$; initial states are drawn at the model's resolution and clipped to $[0, 1]$. At 64×64 we take the empirical mean and full covariance of the 12,288-dimensional vectors. At 256×256 a full covariance is out of reach (d^2 entries, 155 GB, and an $O(d^3)$ factorization), so we estimate a low-rank-plus-diagonal 256×256 Gaussian, $\hat{\Sigma} = LL^\top + \mathrm{diag}(d)$ with $L \in \mathbb{R}^{d\times 2048}$: the leading 2,048 eigendirections of the empirical covariance are obtained by a randomized Nyström approximation and the diagonal residual restores the marginal variance of every coordinate; the low-rank part explains 91.6% of the empirical variance, the factors take 1.6 GB, and initial states are $x_0 = \mathrm{clip}_{[0,1]}(\mu + Lz + \sqrt{d} \odot \epsilon)$ with independent standard normal z and ϵ. At 64×64 we train this baseline at three run lengths, $T \in \{40, 50, 60\}$, and report the run with the best FID ($T = 50$; all three in Table 4).

A.1.4 SGLD SAMPLER

The SGLD comparison of Section 4.3 trains and samples with the stochastic update of Eq. (4): the unnormalized gradient, a fixed noise scale $\sigma > 0$ added at every step without annealing, and the same per-step clipping to the input range. As in the short-run samplers of prior work, the noise scale is decoupled from the step size (Du & Mordatch, 2019; Grathwohl et al., 2020). Everything else follows the target-only training of Section 4.1 and Table 3. Table 6 lists the two runs: one at the noise scale of prior work, $\sigma = 0.005$ (Du & Mordatch, 2019; Grathwohl et al., 2020), which collapsed late in training, and one at $\sigma = 0.0025$, which we report. Rather than tuning the step size by hand, we calibrate it on the target-only PGD model of Table 1: $\eta = 3$ matches the mean per-step displacement of the PGD sampler at its training step size. At test time the sampler uses the training step size and noise scale, and the step count T is selected as in Section 4.1.

Table 6: SGLD training runs on ImageNet 64×64 (ConvNeXt-Base) from uniform noise, differing only in the noise scale. Images seen is the selected checkpoint (training length 3.5M); the test schedule is selected as in Section 4.1 at the training step size and noise scale; metrics are reported at the selected checkpoint, for the collapsed run its best checkpoint before collapse.

σ	η	T	Seed	Images seen	Test T	FID ↓	Prec. ↑	Rec. ↑	Outcome
0.0025	3.0	150	1	3.1M	174	3.58	0.73	0.48	stable
0.005	3.0	150	1	1.0M	163	6.57	0.66	0.42	collapse (after 1.1M)

A.2 Classification, OOD Detection, and Calibration

A JEM-style model is one network that classifies, generates, and detects OOD inputs, and Grathwohl et al. (2020) showed that the generative term also improves the calibration of the classifier; we check whether relay training preserves these properties.

The relay-trained and informative-Gaussian models have nearly the same clean accuracy, below DAT (Yin et al., 2026) and further below the classifier they start from, and robust accuracy on par with DAT (Table 7). Generative training improves calibration: the adversarially trained classifier is strongly under-confident, the three generatively trained models much less so, with DAT the best calibrated; all four lie above the diagonal in Figure 4, so the remaining error is under-confidence.

On OOD detection (Table 8), the adversarially trained classifier is already a strong detector: it separates clean OOD inputs best on all three sources and keeps most of that separation under attack. Generative training does not improve on it uniformly: all three generatively trained models are lower on clean inputs, and under attack they are higher on some sources and lower on others, so no model is best throughout.

Across the three tasks, the two noise-initialized models are close on classification and calibration; on OOD detection the Gaussian model is ahead on CIFAR-10 and SVHN and the relay model on uniform noise. Relative to DAT, both lose some clean accuracy and calibration, keep its robust accuracy, and are lower on SVHN but otherwise comparable on OOD detection.

Table 7: Clean and robust accuracy and calibration on ImageNet 256×256. Robust accuracy is under AutoAttack with ℓ_2, $\epsilon = 3.0$; baseline accuracies are from Yin et al. (2026). Standard AT is trained under ℓ_∞ ($\epsilon = 4/255$), the other models under ℓ_2 (Table 3). ECE (%) is from Figure 4.

Model	Params	Clean acc. ↑	Robust acc. ↑	ECE ↓
Standard AT (Singh et al., 2023)	198M	78.3	33.4	33.8
DAT (Yin et al., 2026)	198M	75.8	56.4	7.1
Informative Gaussian	198M	70.2	56.2	15.6
Relay training (ours)	198M	70.6	56.4	13.3

Table 8: OOD detection on ImageNet 256×256 with the protocol of DAT (Yin et al., 2026): AUROC (%) of the maximum softmax probability, ImageNet validation images as the in-distribution set, 1,024 images per OOD source upsampled to 256×256. Adversarial columns attack only the OOD inputs (10 PGD steps, ℓ_2, $\epsilon = 3$, maximizing the KL divergence to a uniform prediction); as this does not minimize the score directly, adversarial AUROC can exceed clean AUROC.

	CIFAR-10		SVHN		Uniform noise	
Model	Clean	Adv.	Clean	Adv.	Clean	Adv.
Standard AT (Singh et al., 2023)	89.5	64.1	99.0	88.3	100.0	99.9
DAT (Yin et al., 2026)	80.4	63.8	97.0	89.3	99.3	94.3
Informative Gaussian	79.7	68.8	94.2	88.3	96.0	94.9
Relay training (ours)	78.7	66.8	91.6	83.5	99.6	99.4

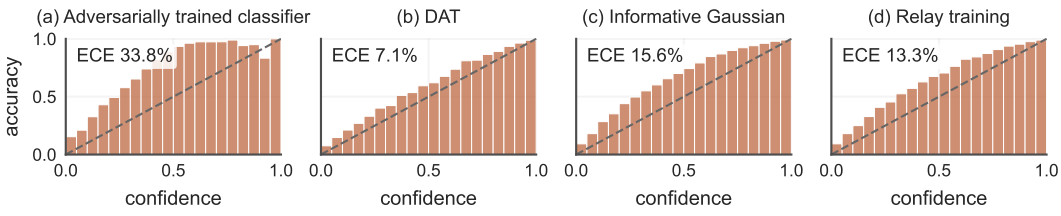


Figure 4: Reliability diagrams on the ImageNet validation set with 20 confidence bins.

A.3 Denoising by Short-Run Descent

We test whether the trained energies remove corruption from a real image by short-run descent alone. The three PGD-trained ImageNet 64×64 models of Table 1 receive the same corrupted images, one validation image from each of the 1,000 classes with 10% of the pixels replaced by black or white, and we run the PGD update of Section 4.1 from the corrupted image under the true label with $\eta = 2$ for all models and no fidelity term. Figure 5 reports mean PSNR and LPIPS against the clean image along the trajectory.

The outcome follows the initialization used in training. The two models trained from uniform noise denoise: PSNR rises by about 2 dB to a peak at $T = 10$ and LPIPS falls to its minimum at $T = 20$, after which the chains drift from the original, as expected without a fidelity term. The informative-Gaussian model gains much less: its PSNR falls below that of the input from $T = 5$ on, and its LPIPS levels off at 0.43 against 0.29 for the target-only and 0.27 for the relay model. The same division appears at 256×256 (Figure 6): the relay model removes the corruption, whereas the informative-Gaussian model overlays much of the image with a high-contrast texture.

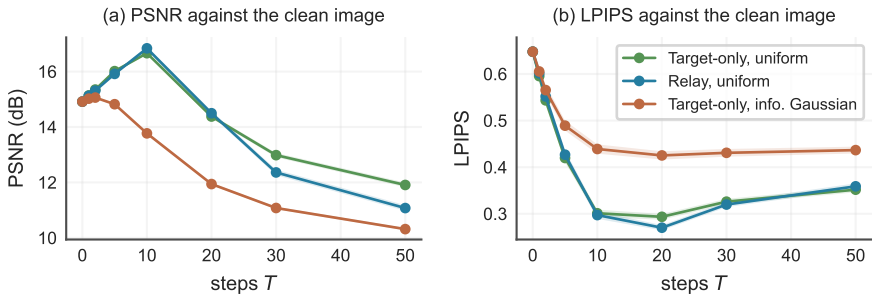


Figure 5: Denoising by short-run PGD descent on ImageNet 64×64 (one validation image per class, 10% salt-and-pepper corruption, $\eta = 2$): mean PSNR and LPIPS against the clean image along the trajectory, with 95% bootstrap bands over images, for the three PGD-trained models of Table 1.

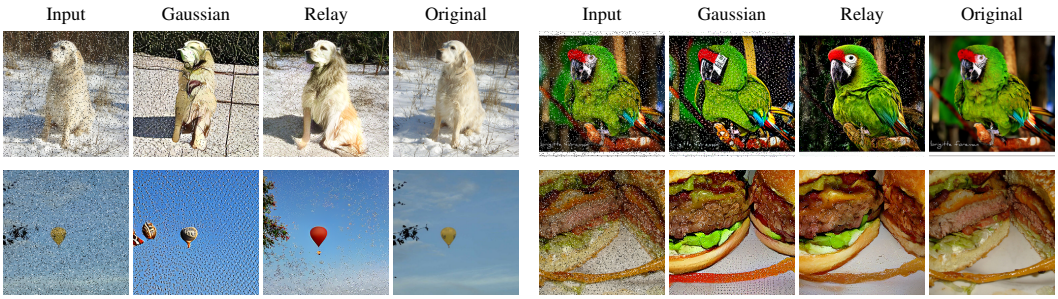


Figure 6: Denoising at 256×256 by the informative-Gaussian and relay models of Table 2: the input with 10% salt-and-pepper corruption, the result of $T = 50$ steps with $\eta = 2$, and the original image.

A.4 Qualitative Comparison on ImageNet 64 × 64

Figure 7 shows uncurated samples of three classes from the four models of Table 1, each at its selected schedule. Within-class variety in setting, pose, and lighting increases from the PGD to the SGLD target-only model and again for the informative-Gaussian and relay-trained models, in the order of their recall; the last two are visually close, as are their recall values.

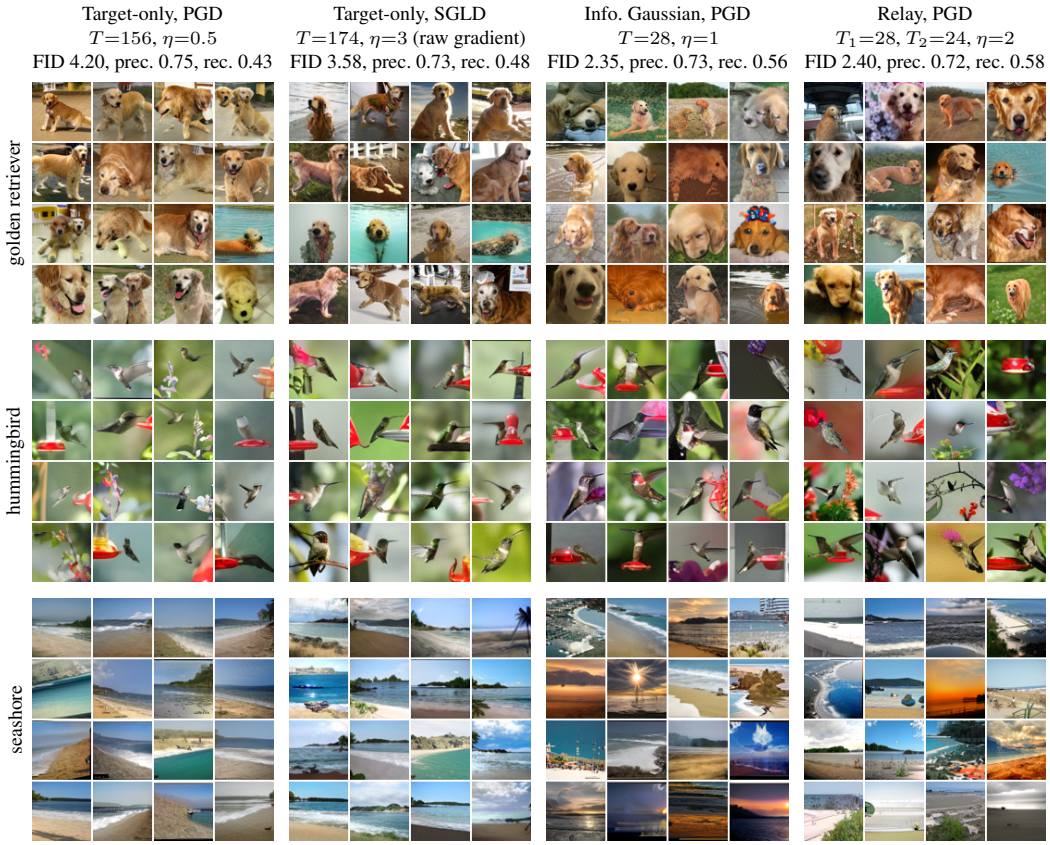


Figure 7: Uncurated samples of three ImageNet classes at 64 × 64 (rows) from the four models of Table 1 (columns), each at its selected schedule (Table 4) with its FID, precision, and recall from Table 1: the first 16 samples of the class from each model's evaluation set.

A.5 IMAGENET 256×256 SAMPLES

Figure 8 shows uncurated class-conditional samples of the relay-trained 256×256 model of Table 2, and Figure 9 the same classes from the informative-Gaussian model. The Gaussian model's samples vary less within a class in pose, framing, and background, in particular in the golden retriever and cheeseburger rows, consistent with its lower recall.

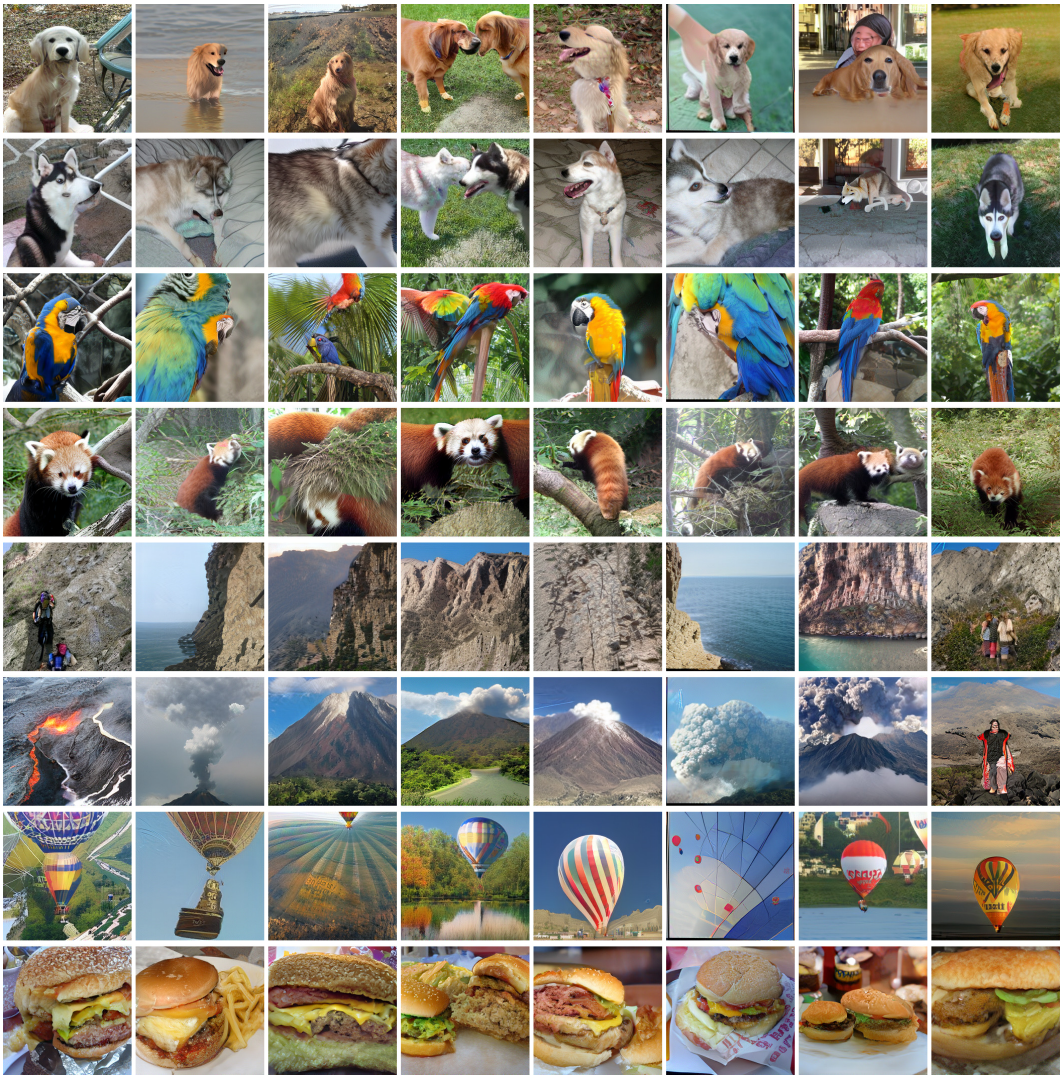


Figure 8: Uncurated class-conditional samples on ImageNet 256×256 from the relay-trained ConvNeXt-Large model of Table 2, generated with $(T_1, T_2) = (36, 25)$ and step size $\eta = 8$. Each row is one class (golden retriever, Siberian husky, macaw, red panda, cliff, volcano, balloon, cheeseburger) and shows seeds 0–7 from left to right; all samples are included without filtering or replacement.

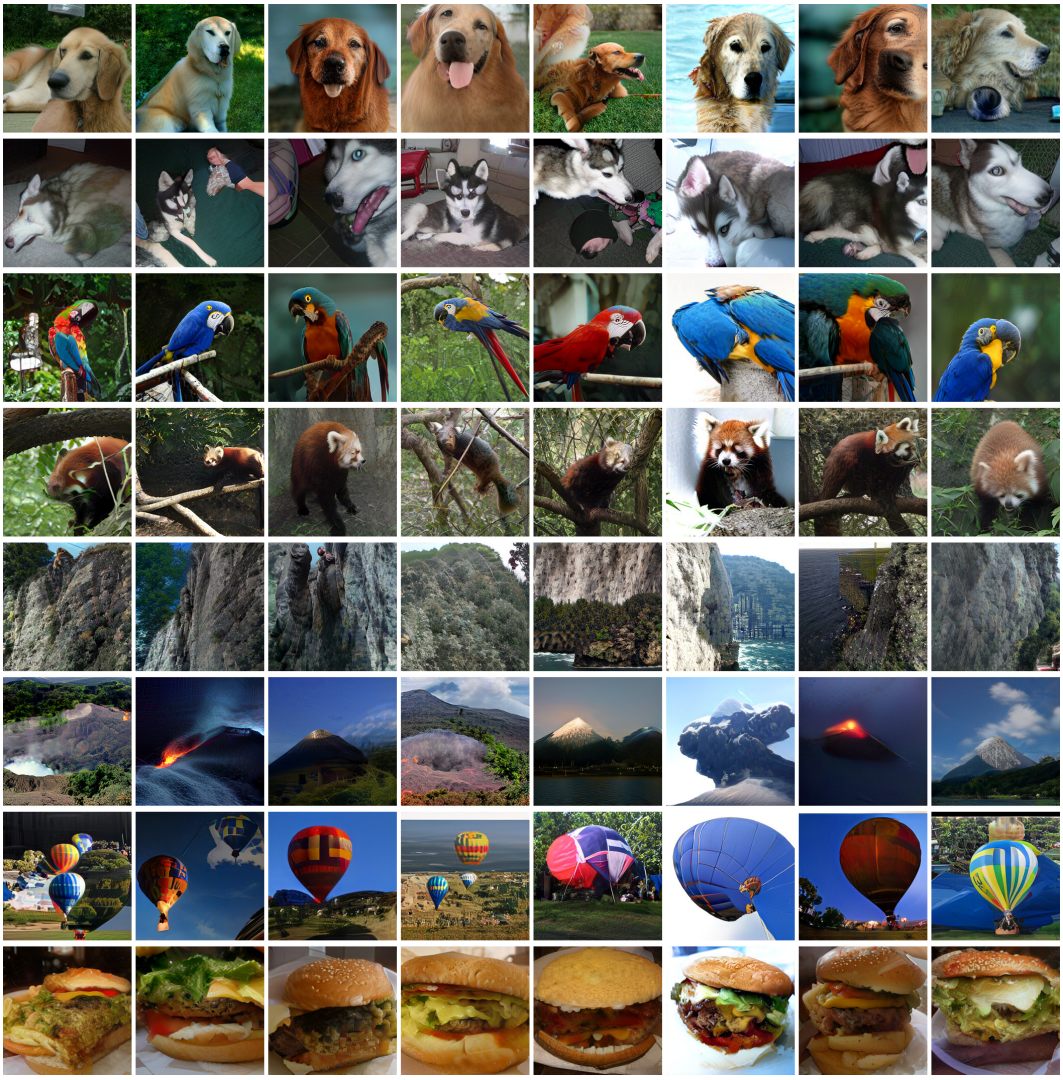


Figure 9: Uncurated class-conditional samples on ImageNet 256 × 256 from the informative-Gaussian ConvNeXt-Large model of Table 2, generated with $T = 20$ and step size $\eta = 8$; the same classes as Figure 8, seeds 0–7 per class, without filtering or replacement.