GENERATION-INDUCED REPRESENTATION LEARNING: SELF-GUIDED DIFFUSION WITHOUT PRETEXT TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative training alone can drive semantic representation learning. We demonstrate this via a new framework that jointly trains an image encoder and a conditional diffusion model end-to-end from scratch, using no labels, no contrastive loss, no masking, and no data augmentation beyond horizontal flips. The encoder outputs a latent vector that both controls conditional generation and serves as a suitable feature representation for downstream semantic tasks. An additional diffusion model trained over the encoder's output space allows for sampling of latent representations, and, chained together with the conditional model, yields a system for unconditional image generation. Key to our system's learning dynamics are independently sampled noise timesteps for the conditional model's image and latent inputs. On CIFAR-10, CIFAR-100, and ImageNet-100, our system substantially improves unconditional generation over an architecturally matched DDPM baseline with identical compute. Our central claim is one of existence and minimality: an encoder *trained purely by the denoising diffusion loss*, under a deliberately minimal flip-only augmentation budget, reaches kNN classification accuracy comparable to dedicated self-supervised methods such as SODA, despite using no pretext task whatsoever. Moreover, equipping our method with the same data augmentation recipe as SimCLR, so that the two differ only in their training objective, our purely generative objective *outperforms* SimCLR's contrastive objective on three of four probing protocols across CIFAR-10 and CIFAR-100. Together, these results establish that conditional generation, when coupled with appropriate training mechanisms, is itself a viable representation learning objective.

1 INTRODUCTION

Conditional image generation models consistently outperform their unconditional counterparts, but current approaches source the conditioning signals driving this gap—*e.g.,* class labels (Ho & Salimans, 2022), text (Saharia et al., 2022; Ramesh et al., 2022), or features from pretrained self-supervised encoders (Li et al., 2024; Yu et al., 2025)—from outside the generative pipeline, presupposing a prior stage of annotation, captioning, or self-supervised pretraining on curated data. We explore an unanswered question: *is the conditional diffusion denoising objective sufficient, on its own, to drive an encoder to learn useful semantic representations that further benefit generation quality?* A positive answer would obviate the need for separate representation learning frameworks and unify representation learning with generation. Learning to generate via such a conditional architecture would, as a byproduct, also train the encoder to produce semantic feature embeddings.

Existing self-supervised learning (SSL) methods introduce explicit pretext tasks, such as contrastive discrimination (Chen et al., 2020; He et al., 2020), asymmetric prediction (Grill et al., 2020; Chen & He, 2021), self-distillation (Caron et al., 2021; Oquab et al., 2024), or masked reconstruction (He et al., 2022), precisely because raw generative signals were thought to be either too narrow or too unstable to produce general-purpose features. Two-stage pipelines such as RCG (Li et al., 2024) and REPA (Yu et al., 2025) embody this view: representation learning and generation are treated as separate problems, solved sequentially. We demonstrate such separation is unnecessary.

Our **G**eneration-**i**nduced **R**epresentation **L**earning (**GiRL**) framework jointly trains three components end-to-end: an encoder, a conditional image diffusion model that consumes the encoder's output, and a lightweight diffusion model that generates samples in the encoder's output space

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.

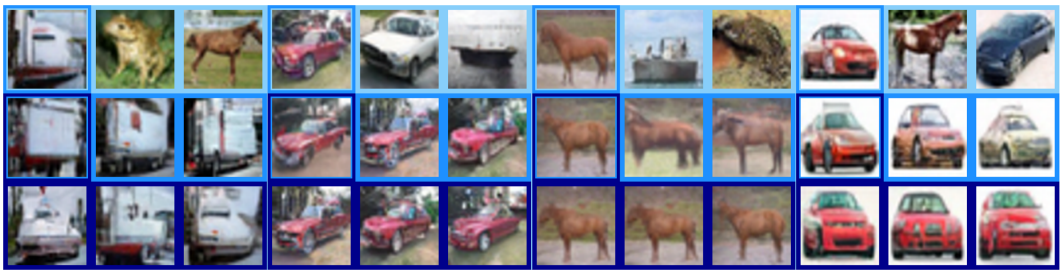

Figure 1: **Hierarchical semantic structure of the learned conditioning latent.** *Top row:* Unconditional samples from pure Gaussian noise. *Lower rows:* A selected image from the row above (marked by border color) is encoded, its latent is partially noised, and the result conditions our image diffusion model. Large latent noise *(middle)* changes scene or object layout; small latent noise *(bottom)* changes only local details. The comparison of interest is the diversity *within* each row: as latent noise decreases, the conditional distribution contracts and the remaining variation becomes progressively more local. Here, we fix the initial noise for image diffusion, so differences reflect the conditioning signal rather than sampling variance. The behavior of our generatively learned hierarchy parallels that obtained from pretrained contrastive features in Zhang et al. (2025).

(*i.e.,* generates latent feature representations). We train these components from scratch, using *only the denoising diffusion loss*—no contrastive loss, no masking, no self-distillation, and no data augmentation beyond horizontal flipping. Our lightweight feature-space diffusion model enables unconditional sampling at inference time: latent representations are first synthesized from pure noise, and then decoded into images by the conditional generator.

Figure 1 illustrates how moving in our learned latent space traverses the structure of conditionally-generated images. This noise-perturbation visualization shows the most significant bits of our latent features encode category-level semantics, and hence perform well for downstream classification tasks. Figure 2 diagrams our learning framework. A key novelty is full, independent noise schedules on both the image and the latent feature input to our conditional diffusion model. As a consequence, the encoder should produce latents useful for recovering the original image from any noise level, and thus should contain as much detail as possible. Yet, those same latents must themselves be noise-robust, forcing emergence of coarse-to-fine latent space geometry reflecting semantic hierarchy.

As evidenced through experiments in Section 4, our contributions include:

- Demonstration that *denoising alone suffices to drive representation learning*. Our encoder, trained purely by the image denoising loss, outperforms SimCLR (Chen et al., 2020) on three of four protocol–dataset combinations, including +7.2 kNN points on CIFAR-10. Generation is a natural objective; matching performance of contrastive methods is a call to reconsider the latter.
- Representation learning *free from reliance on data augmentation schemes*. Unlike contrastive methods, GiRL is performant with flip-only augmentation; SimCLR loses over 20 linear-probing points when color distortion and cropping are removed (Chen et al., 2020). Data augmentation is a crude mechanism for building in domain knowledge (semantic-invariant transformations); minimizing or eliminating it is a path toward a more robust and broadly applicable framework.
- Improved *generation quality*. GiRL improves unconditional generation (Figure 2, *right*) over an architectural- and compute-matched DDPM (Ho et al., 2020) baseline. Even if one only wanted a system for image synthesis, GiRL is a superior alternative to a standard DDPM.
- *Unification* of representation learning and generative modeling in a single, from scratch, end-to-end jointly-trained framework. Generation drives latent representation learning and latent conditioning improves generation.

The goal of unification and synergy between generative modeling and representation learning predates recent SSL methods, and includes earlier efforts at implementing such architectures with GANs (Donahue et al., 2016; Dumoulin et al., 2016). The subpar feature learning performance, in comparison to specialized SSL methods, of these GAN frameworks motivated focus on the latter. Our formulation of such unification in a diffusion-based framework closes this performance gap, and opens a new path for the future of representation learning systems.

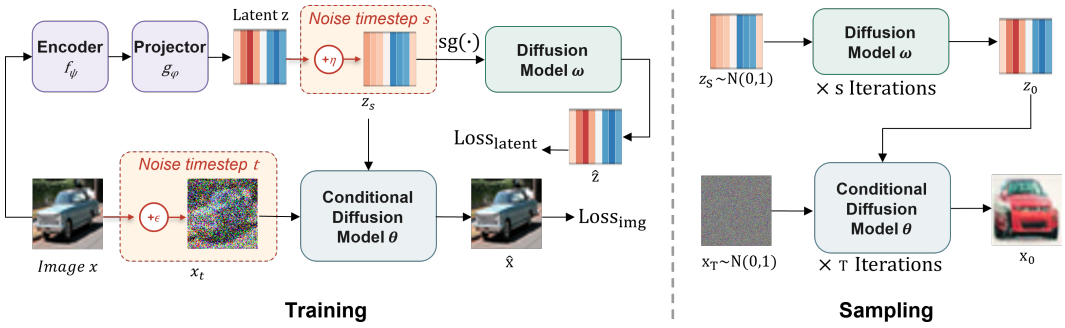

Figure 2: **GiRL framework.** *Training (left):* An encoder f_ψ and projector g_ϕ (with non-affine LayerNorm) map $\mathbf{x}$ to representation $\mathbf{z}$, which conditions image diffusion model θ via adaLN-Zero. An MLP-based diffusion model ω learns the distribution of $\mathbf{z}$. A stop-gradient on $\mathbf{z}$ before the latent branch ensures the encoder is updated solely by $\mathcal{L}_{\text{img}}$. *Sampling (right):* Generate $\mathbf{z}$ from Gaussian noise via the latent diffusion model; generate $\mathbf{x}$ via the image diffusion model conditioned on $\mathbf{z}$.

2 Related Work

Diffusion models for image generation. Diffusion models (Ho et al., 2020) and their relatives (score matching (Song & Ermon, 2019), score-based SDEs (Song et al., 2021), flow matching (Lipman et al., 2023; Liu et al., 2023)) have advanced through architectures such as ADM (Dhariwal & Nichol, 2021), Latent Diffusion (Rombach et al., 2022), and DiT (Peebles & Xie, 2023), and through conditioning mechanisms from classifier(-free) guidance (Dhariwal & Nichol, 2021; Ho & Salimans, 2022) to text-to-image systems (Saharia et al., 2022; Ramesh et al., 2022). In all cases, the conditioning signal comes from outside the generative pipeline: labels, captions, or pretrained encoders. We ask whether such external signals are needed at all in the unconditional regime.

Representation-guided and self-conditioned diffusion. RCG (Li et al., 2024) conditions a generator on frozen MoCo v3 (Chen et al., 2021) features and fits a separate representation diffusion model for sampling; REPA (Yu et al., 2025) aligns DiT features with DINOv2 (Oquab et al., 2024); Zhang et al. (2025) nest diffusion models along a pretrained contrastive feature hierarchy. All treat an SSL-pretrained, frozen encoder as a prerequisite, so the encoder is never optimized to serve the generator and the recipe does not transfer to domains without suitable pretrained backbones. A second line of work derives the conditioning signal from the data itself, via clustering pseudo-labels (Liu et al., 2020) or masked modeling (Li et al., 2023; Chu et al., 2025), but retains an explicit reconstruction pretext task. Table 1 places GiRL among frameworks that combine an encoder, a generator, and a latent prior (Preechakul et al., 2022; Wang et al., 2023; Xiang et al., 2023; Li et al., 2024): it is the only one in which all three are trained jointly from scratch under the denoising loss alone, with no pretext task, auxiliary regularizer, or post hoc stage.

Noise on conditioning signals. Noising a conditioning input appears in prior work in restricted forms: noise conditioning augmentation (Ho et al., 2022) perturbs a fixed observed low-resolution image for robustness, classifier-free guidance (Ho & Salimans, 2022) is a two-point schedule (condition present or dropped), and SDEdit (Meng et al., 2022) perturbs the image at inference with no matching training objective. GiRL instead noises a *jointly learned* latent across a full schedule as part of the training objective, with latent timestep s decoupled from image timestep t, so that a single generator learns the continuum from unconditional ($s{=}S$) to instance-determined ($s{=}0$) synthesis, and inference can operate with any arbitrary level $\tilde{s}$ of latent noise. Appendix D quantifies the cost of omitting this schedule.

Self-supervised representation learning. SSL methods learn features through pretext tasks: contrastive discrimination (Chen et al., 2020; He et al., 2020), asymmetric prediction (Grill et al., 2020; Chen & He, 2021), self-distillation (Caron et al., 2021; Oquab et al., 2024), and masked reconstruction (He et al., 2022). GiRL shows that a conditional diffusion objective alone, with no pretext task and only horizontal flip augmentation, is competitive on standard probes. Our stop-gradient resembles that of BYOL (Grill et al., 2020) and SimSiam (Chen & He, 2021) but serves a different

Method	Encoder trained by	Pretext / auxiliary term	Latent prior	Joint?
MAGE (Li et al., 2023)	masked token modeling (VQ tokens)	masking	—	No[†]
Diff-AE (Preechakul et al., 2022)	diffusion loss through decoder	none	DDIM on latents, trained post-hoc (stage 2)	No
InfoDiffusion (Wang et al., 2023)	diffusion loss	mutual-information regularizer	yes	Partial
DDAE (Xiang et al., 2023)	not trained (features read from a pretrained unconditional denoiser)	none	—	N/A
RCG (Li et al., 2024)	not trained (frozen pretrained SSL encoder, MoCo v3)	contrastive (inherited from pretraining)	RDM on frozen features, trained separately	No
GiRL (ours)	**diffusion loss only**	**none**	**LiL, trained jointly**	**Yes**

Table 1: Design-point comparison of frameworks combining an encoder, a generator, and a latent prior. GiRL is the only one in which all components are trained jointly from scratch under the denoising loss alone. [†]Single training stage, but masking itself constitutes the reconstruction pretext task.

purpose: there it breaks symmetry between augmented views and is the mechanism for learning; here it decouples two generative objectives that would otherwise pull the encoder in opposite directions. Finally, extracting features from pretrained diffusion models post hoc (Stracke et al., 2025) decouples representation from generation and cannot use semantics to improve samples; GiRL trains both jointly, and since our encoder sees clean images, it yields clean features when used alone.

3 Method

Figure 2 gives an overview of GiRL. An encoder produces a representation $\mathbf{z}$ that conditions a pixel-space image diffusion model; a lightweight latent diffusion model is trained alongside to capture the distribution of $\mathbf{z}$, allowing generation of latent representations. The encoder receives gradients only from the image diffusion denoising loss. The remaining components stabilize learning dynamics.

3.1 Notation and Preliminaries

We denote an image by $\mathbf{x} \in \mathbb{R}^{H\times W\times 3}$ and its learned representation by $\mathbf{z} \in \mathbb{R}^d$. Two diffusion processes operate in parallel: an *image diffusion* process over $\mathbf{x}$ with timestep $t \in \{1,\dots,T\}$, and a *latent diffusion* process over $\mathbf{z}$ with timestep $s \in \{1,\dots,S\}$. Both follow the standard DDPM formulation, with their own variance schedules $\{\beta_t^x\}$ and $\{\beta_s^z\}$; we use $\bar\alpha_t^x$ and $\bar\alpha_s^z$ to denote the corresponding cumulative products.

3.2 Joint Training of Encoder and Image Diffusion

The encoder is a Vision Transformer (Dosovitskiy et al., 2021) that maps an image $\mathbf{x}$ to a single representation vector via its [CLS] token. To shape this representation into a form suitable for downstream conditioning, we pass the [CLS] token through a feature projector g_ϕ consisting of three Linear–LayerNorm–SiLU blocks followed by a non-affine LayerNorm (Ba et al., 2016):

$$\mathbf{z} = g_\phi(f_\psi(\mathbf{x})_{[\mathrm{CLS}]}), \qquad g_\phi(\cdot) = \mathrm{LN}_{\text{no-affine}} \circ \big(\mathrm{Linear} \circ \mathrm{LayerNorm} \circ \mathrm{SiLU}\big)^{\times 3}(\cdot), \tag{1}$$

where f_ψ denotes the ViT encoder. The role of the final non-affine LayerNorm is discussed in Section 3.4.

The image diffusion model is a Diffusion Transformer (Peebles & Xie, 2023) that predicts the noise added to a noisy image $\mathbf{x}_t$ conditioned on a noisy latent $\mathbf{z}_s$. The two diffusion processes use independent noise schedules and timesteps, sampled at each training step as $t \sim \mathcal{U}\{1,\dots,T\}$ and $s \sim \mathcal{U}\{1,\dots,S\}$:

$$\mathbf{x}_t = \sqrt{\bar\alpha_t^x}\,\mathbf{x}_0 + \sqrt{1-\bar\alpha_t^x}\,\boldsymbol{\epsilon}, \qquad \mathbf{z}_s = \sqrt{\bar\alpha_s^z}\,\mathbf{z}_0 + \sqrt{1-\bar\alpha_s^z}\,\boldsymbol{\eta}, \tag{2}$$

Algorithm 1 GiRL Training

Require: image $\mathbf{x}$, encoder f_ψ, projector g_ϕ, image diffusion $\boldsymbol{\epsilon}_\theta$, latent diffusion $\boldsymbol{\epsilon}_\omega$

$\mathbf{z} \leftarrow g_\phi(f_\psi(\mathbf{x})_{[\text{CLS}]})$ ▷ Encode
Sample $t \sim \mathcal{U}\{1, \dots, T\}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
Sample $s \sim \mathcal{U}\{1, \dots, S\}$, $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
$\mathbf{x}_t \leftarrow \sqrt{\bar{\alpha}_t^x}\,\mathbf{x} + \sqrt{1-\bar{\alpha}_t^x}\,\boldsymbol{\epsilon}$
$\mathbf{z}_s \leftarrow \sqrt{\bar{\alpha}_s^z}\,\mathbf{z} + \sqrt{1-\bar{\alpha}_s^z}\,\boldsymbol{\eta}$
$\mathcal{L}_{\text{img}} \leftarrow \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t,\, t,\, \mathbf{z}_s,\, s)\|_2^2$
$\mathcal{L}_{\text{lat}} \leftarrow \|\boldsymbol{\eta} - \boldsymbol{\epsilon}_\omega(\text{sg}(\mathbf{z})_s,\, s)\|_2^2$ ▷ Stop-gradient
$\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{img}} + \mathcal{L}_{\text{lat}}$
Update $\psi, \phi, \theta, \omega$ via $\nabla \mathcal{L}_{\text{total}}$

Algorithm 2 GiRL Sampling

Require: latent steps S, image steps T, optional early-stop $\tilde{s} \in [0, S]$ (default 0)

Sample $\mathbf{z}_S \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $s = S, S{-}1, \dots, \tilde{s}{+}1$ **do**
  $\mathbf{z}_{s-1} \leftarrow \textsc{LatentStep}(\mathbf{z}_s, s;\, \boldsymbol{\epsilon}_\omega)$
end for
$\mathbf{z} \leftarrow \mathbf{z}_{\tilde{s}}$ ▷ Conditioning signal
Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, T{-}1, \dots, 1$ **do**
  $\mathbf{x}_{t-1} \leftarrow \textsc{ImageStep}(\mathbf{x}_t, t, \mathbf{z}, \tilde{s};\, \boldsymbol{\epsilon}_\theta)$
end for
return $\mathbf{x}_0$

where $\boldsymbol{\epsilon}, \boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\mathbf{z}_0$ is the projector output defined in equation 1. We inject the conditioning signal into the DiT through adaLN-Zero (Perez et al., 2017), following the standard design. Concretely, we form a single conditioning vector $\mathbf{c}_x$ by concatenating the two timestep embeddings with the noisy latent $\mathbf{z}_s$, and project it back to the model dimension with a linear layer:

$$\mathbf{c}_x = \text{Linear}\big(\big[\,\text{TimeEmbed}(t) \,\|\, \text{TimeEmbed}(s) \,\|\, \mathbf{z}_s\,\big]\big)\,, \tag{3}$$

where $\|$ denotes concatenation. The vector $\mathbf{c}_x$ is then fed to every adaLN-Zero block of the DiT to modulate its activations. The image diffusion loss follows the simplified DDPM objective:

$$\mathcal{L}_{\text{img}} = \mathbb{E}_{\mathbf{x}_0,\, \boldsymbol{\epsilon},\, \boldsymbol{\eta},\, t,\, s}\Big[\, \big\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t,\, t,\, \mathbf{z}_s,\, s)\big\|_2^2 \,\Big]\,. \tag{4}$$

Crucially, gradients from $\mathcal{L}_{\text{img}}$ flow back through $\mathbf{z}_s$ into both the projector g_ϕ and the ViT encoder f_ψ. The image diffusion loss is the sole driver of representation learning in GiRL: the encoder is implicitly trained to produce features that maximally help denoising across all latent-noise levels s.

3.3 Latent Diffusion over Learned Representations

The image diffusion model relies on $\mathbf{z}_s$ as a conditioning input, but during unconditional sampling no real image is available to compute $\mathbf{z}_0$ from the encoder. To close this loop, we train a second diffusion model to learn the distribution of clean representations $\mathbf{z}_0$ produced by the encoder. Reusing the noising process in Eq. 2, the latent diffusion model $\boldsymbol{\epsilon}_\omega$ is a lightweight MLP-based denoiser that predicts the noise component $\boldsymbol{\eta}$ from $\mathbf{z}_s$ and timestep s. Its training objective is

$$\mathcal{L}_{\text{lat}} = \mathbb{E}_{\mathbf{z}_0,\, \boldsymbol{\eta},\, s}\Big[\, \big\|\boldsymbol{\eta} - \boldsymbol{\epsilon}_\omega(\mathbf{z}_s,\, s)\big\|_2^2 \,\Big]\,. \tag{5}$$

The total training loss is the simple sum of the two objectives:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{img}} + \mathcal{L}_{\text{lat}}. \tag{6}$$

We deliberately do not introduce a tunable weighting coefficient between the two terms, as this would add a hyperparameter that interacts with the stabilization mechanisms described next. We instead rely on the architectural choices in Section 3.4 to balance the two objectives.

3.4 Stabilizing Joint Training

Naively optimizing $\mathcal{L}_{\text{total}}$ end-to-end fails: the encoder rapidly collapses to a degenerate representation that minimizes both losses simultaneously without encoding any meaningful image content. We identify two coupled failure modes and address each with a targeted design choice.

Stop-gradient between the two diffusion models. Without intervention, gradients from $\mathcal{L}_{\text{lat}}$ flow through $\mathbf{z}$ into the encoder and create opposing pressures: $\mathcal{L}_{\text{img}}$ pushes for expressive representations that aid pixel-level denoising, while $\mathcal{L}_{\text{lat}}$ is minimized when $\mathbf{z}$ follows a low-entropy distribution that

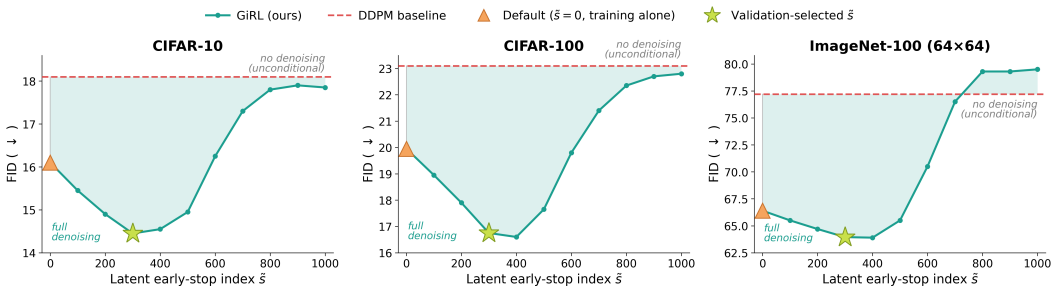

Figure 3: **FID across latent-diffusion stopping points** $\tilde{s}$ on CIFAR-10, CIFAR-100, and ImageNet-100. Each $\tilde{s}$ runs the latent diffusion for $S - \tilde{s}$ reverse steps and conditions the image diffusion on the partially denoised $\mathbf{z}_{\tilde{s}}$. At $\tilde{s} = 1000$ the latent is pure noise and GiRL reduces to unconditional generation, matching the DDPM baseline. FID improves as the latent becomes more informative, but the optimum lies at $\tilde{s} \approx 300$–400 rather than 0 on all three datasets (see Section 4.1). Headline FIDs use $\tilde{s} = 300$, selected by a per-dataset search over this sweep that converged to the same value on all three datasets (Appendix A.7); Table 2 also reports the search-free point $\tilde{s} = 0$.

is trivial to model. We eliminate this conflict by applying a stop-gradient on $\mathbf{z}$ before it enters the latent diffusion model:

$$\mathcal{L}_{\text{lat}} = \mathbb{E}_{\mathbf{z},\,\boldsymbol{\eta},\,s}\left[\,\left\|\boldsymbol{\eta} - \boldsymbol{\epsilon}_\omega(\mathrm{sg}(\mathbf{z})_s,\; s)\right\|_2^2\right], \tag{7}$$

where $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator. The encoder is now updated exclusively through $\mathcal{L}_{\text{img}}$, and the two objectives are decoupled at the encoder, removing the need for explicit loss balancing. This asymmetric coupling resembles the stop-gradient in BYOL (Grill et al., 2020) and SimSiam (Chen & He, 2021), but here decouples two generative objectives rather than two augmented views.

Non-affine LayerNorm at the projector output. A second failure mode appears even with stop-gradient: $\mathcal{L}_{\text{lat}}$ stalls at a high value when the encoder's output distribution drifts in scale or shape during training, leaving the latent diffusion model with a non-stationary target it cannot fit. We address this by terminating the projector g_ϕ with a non-affine LayerNorm,

$$\mathrm{LN}_{\text{no-affine}}(\mathbf{u}) = \frac{\mathbf{u} - \mu(\mathbf{u})}{\sigma(\mathbf{u})}, \tag{8}$$

which guarantees per-sample zero mean and unit variance regardless of the encoder's evolving weights. The absence of learnable affine parameters is essential: a standard LayerNorm would let the encoder rescale and shift its outputs, defeating the constraint.

3.5 Sampling

At inference, image generation proceeds in two stages: we first obtain a conditioning latent from the lightweight latent diffusion model, and then synthesize an image conditioned on that latent via the image diffusion model. Both stages run their respective reverse processes starting from Gaussian noise, with no input image required.

Concretely, we draw $\mathbf{z}_S \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and run the latent diffusion model in reverse, but the user may stop the process at any intermediate step $\tilde{s} \in [0, S]$ rather than running all S steps. The resulting latent $\mathbf{z}_{\tilde{s}}$ is then passed to the image diffusion model, which is run for its full T reverse steps, conditioned on $\mathbf{z}_{\tilde{s}}$ together with the same $\tilde{s}$ that was used to produce it. This procedure is well-defined for every $\tilde{s} \in [0, S]$, because the image diffusion model is trained on $\mathbf{z}_s$ for all values of s in the latent-diffusion schedule (Section 3.2); a noisy conditioning latent is therefore an in-distribution input to the image generator at inference, not an out-of-distribution substitute. This distinguishes GiRL from two-stage pipelines in which the generator is trained on clean latents alone and has no reason to behave well when given a noisy latent at inference.

4 EXPERIMENTS

We evaluate GiRL on unconditional image generation, focusing on three questions: (i) Does jointly learning representations and diffusion outperform standard unconditional baselines? (ii) Which design choices are responsible for the gain? (iii) How does the framework behave along the compute–quality axis exposed by partial latent denoising?

Datasets. We conduct all experiments on CIFAR-10, CIFAR-100 (Krizhevsky, 2009), and ImageNet-100 (Deng et al., 2009), three standard benchmarks for image generation. We use the official training split and apply random horizontal flipping as the **only** data augmentation.

Architectures. Encoder f_ψ is a ViT-S/4 (Dosovitskiy et al., 2021) with patch size 4, and the image diffusion model $\boldsymbol{\epsilon}_\theta$ is a DiT-S/4 (Peebles & Xie, 2023) with the same patch size. Projector g_ϕ (Eq. 1) outputs $\mathbf{z} \in \mathbb{R}^{384}$, and latent diffusion model $\boldsymbol{\epsilon}_\omega$ is MLP-based with 12 hidden layers of width 384.

Evaluation. For *generation quality* we report FID (Heusel et al., 2017) over 50K samples against the full training set (Ho et al., 2020; Dhariwal & Nichol, 2021). For *representation quality* we follow the standard SSL protocol and report kNN, linear-probe, and fine-tuning accuracy on the encoder.

Generation baselines. We compare against DDPM (Ho et al., 2020) as a representative unconditional diffusion baseline, trained under the same architecture (DiT-S/4) and compute budget as GiRL. We focus on DDPM as the canonical baseline because its training recipe and noise schedule are most directly comparable to ours.

Representation baselines. A central finding is that image representations *emerge* from GiRL's joint diffusion training without any representation-specific machinery. Modern self-supervised methods rely on hand-designed components such as view-generating augmentations, masking schemes, momentum encoders, and negative-sample mining; GiRL uses none of these. To quantify the quality of its emergent features, we compare GiRL's encoder against representative SSL methods trained on identical data: SimCLR (Chen et al., 2020) and MoCo v3 (Chen et al., 2021) (contrastive), BYOL (Grill et al., 2020) (asymmetric prediction), MAE (He et al., 2022) (masked modeling), and SODA (Hudson et al., 2023) (contrastive diffusion). All baselines share the ViT-S/4 architecture and are trained on the same data for the same number of epochs. We report kNN classification, linear probing, and full fine-tuning accuracy.

4.1 MAIN RESULTS

Figure 3 reports FID across CIFAR-10, CIFAR-100, and ImageNet-100 under different inference configurations, and Table 2 reports two operating points side by side. At the reported operating point ($\tilde{s} = 300$, common to all three datasets), GiRL achieves an FID of 14.46 on CIFAR-10, 16.73 on CIFAR-100, and 63.88 on ImageNet-100, improving over the DDPM baseline at the same compute budget by 3.62, 6.39, and 13.29 FID points respectively. We describe the selection protocol transparently: $\tilde{s}$ was searched per dataset against FID over the sweep in Figure 3, and this search converged to the identical value $\tilde{s} = 300$ on all three datasets, so a single, dataset-agnostic operating point reproduces every headline number. To remove any residual dependence on this search, Table 2 also reports the search-free operating point $\tilde{s} = 0$, the natural default corresponding to full denoising of the latent, which uses neither validation nor test data: even there, GiRL improves over DDPM on all three datasets. The improvement is thus consistent across benchmarks and operating points, indicating that representations learned end-to-end from images carry semantic information that meaningfully aids generation regardless of dataset scale or complexity.

Figure 4 shows samples generated when conditioning on $\mathbf{z}_{\tilde{s}}$ obtained by encoding a fixed reference image (leftmost column) and forward-noising to step $\tilde{s}$. At $\tilde{s} = 0$, the generated images closely reproduce the reference's category and instance-level details. As $\tilde{s}$ increases, generations gradually shed instance-level details while retaining category and global color, and finally collapse into unrelated samples at $\tilde{s} \approx 1000$. This provides direct visual evidence that the encoder transmits semantic information through $\mathbf{z}$.

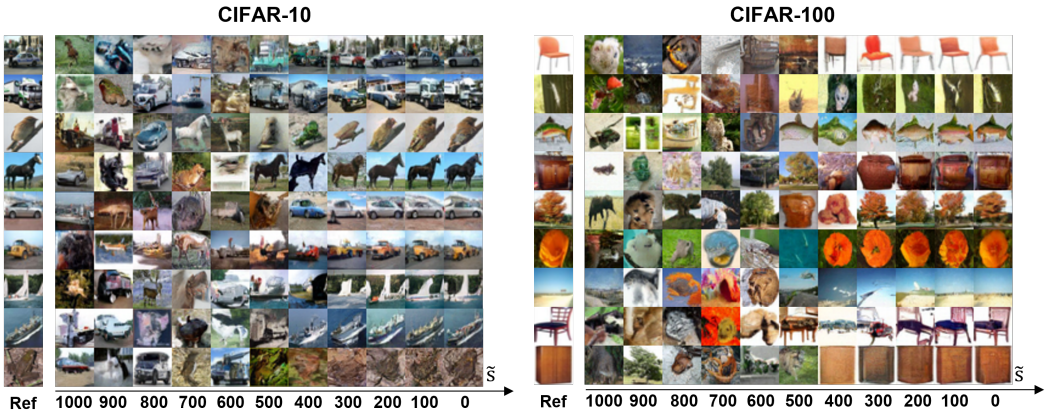

Figure 4: **Generation faithfulness scales with latent information content.** Qualitative samples on CIFAR-10 (left) and CIFAR-100 (right). Each row corresponds to one reference image (leftmost column), whose latent is then corrupted with progressively decreasing noise levels from left to right: the leftmost columns condition on near-pure Gaussian noise (effectively unconditional generation), while the rightmost columns condition on a clean latent. As the conditioning latent becomes cleaner, generated samples transition smoothly from semantically unrelated outputs to faithful reconstructions of the reference, demonstrating that GiRL actively exploits the latent and that the encoder captures generation-relevant semantic information to improve generation quality. The initial noise of the image diffusion model is fixed across the compared noise levels within each row, so the differences shown are attributable to the conditioning signal rather than to sampling variance.

The previous experiments evaluated GiRL as a generative model. We now ask whether the encoder, trained only by the image diffusion loss, also produces useful representations. Table 3 reports kNN classification accuracy on frozen encoder features. GiRL achieves **44.88%** kNN-200 accuracy on ImageNet-100, on par with diffusion-based SSL methods such as SODA (47.26%) under matched compute. Dedicated contrastive and distillation-based methods (MoCo v3 and BYOL) achieve higher accuracy, but the gap is moderate given that GiRL introduces no representation learning objective and uses only horizontal flipping while SSL baselines rely on carefully tuned augmentation pipelines. As a control, Table 3 also reports the same probes on a randomly initialized, completely untrained ViT-S/4: it trails the trained GiRL encoder by roughly 45 and 38 kNN-20 points on CIFAR-10 and CIFAR-100 respectively, with similarly large gaps in linear probing, confirming that the measured representation quality is driven by the training procedure itself rather than by the architecture or initialization.

We interpret this result as evidence that joint diffusion training induces a form of *implicit* representation learning, and we attribute the effect to two architectural constraints that together prevent the encoder from taking a trivial shortcut. First, the projector maps the [CLS] token to a representation $\mathbf{z} \in \mathbb{R}^{384}$ followed by a non-affine LayerNorm, constraining every $\mathbf{z}$ to a fixed-radius sphere. Compared to the input dimensionality, the encoder must summarize, this normalized 384-dimensional channel is far too narrow to transmit pixel-level content, forcing the encoder to allocate capacity to features that are informative about $\mathbf{x}$ rather than ones that mirror it. Second, during training s is sampled uniformly from $\{1, \ldots, S\}$ (Algorithm 1), so the conditioning signal $\mathbf{z}_s$ is heavily noised in most steps; only low-frequency, semantically coherent features survive this corruption to remain useful for denoising. Under these constraints, copying pixels through $\mathbf{z}$ is infeasible, and the only way the encoder can lower $\mathcal{L}_{\text{img}}$ is

	DDPM	GiRL	
Dataset	baseline	$\tilde{s} = 0$	$\tilde{s} = 300$
CIFAR-10	18.08	16.12	**14.46**
CIFAR-100	23.12	19.94	**16.73**
ImageNet-100	77.17	66.39	**63.88**

Table 2: Unconditional FID (↓). $\tilde{s} = 0$ is the search-free default; $\tilde{s} = 300$ is the operating point recovered by a per-dataset search over the Figure 3 sweep on all three datasets. GiRL improves over the compute- and architecture-matched DDPM baseline at both.

Method	Aug.	Framework	CIFAR-10 (%)			CIFAR-100 (%)			ImageNet-100 (%)		
			kNN-20	Lin.	FT	kNN-20	Lin.	FT	kNN-200	Lin.	FT
Random init (untrained)	—	—	32.24	40.67	—	10.54	17.16	—	—	—	—
Contrastive / distillation-based											
SimCLR (Chen et al., 2020)	Strong	Contrastive	76.56	84.54	93.09	51.90	57.32	72.58	59.56	68.28	78.32
MoCo v3 (Chen et al., 2021)	Strong	Contrastive	91.87	92.05	93.20	67.32	67.93	80.01	74.70	75.38	79.92
BYOL (Grill et al., 2020)	Strong	Asym. pred.	86.98	89.00	94.80	53.94	60.47	75.99	58.88	61.96	80.16
Generation-based (with data augmentation)											
SODA (Hudson et al., 2023)	Strong	Contrastive+Diffusion	80.33	82.73	89.20	39.43	52.57	68.66	47.26	51.84	68.46
MAE (He et al., 2022)	Medium	Masked modeling	90.46	91.89	95.01	56.24	64.86	85.44	51.14	68.16	82.36
GiRL (ours)	Strong‡	Diffusion-only	83.79	86.04	—	52.17	56.31	—	—	—	—
Generation-based (trivial augmentation only)											
GiRL (ours)	**Flip-only**	**Diffusion-only**	**77.05**	**77.19**	**87.70**	**48.91**	**50.61**	**64.52**	**44.88**	**48.22**	**66.50**

Table 3: Representation quality across three evaluation protocols (kNN / Linear probing / Finetune) on CIFAR-10, CIFAR-100, and ImageNet-100 (64×64). All methods use ViT-S/4. GiRL learns representations purely as a byproduct of generative training. ‡Trained with SimCLR's exact augmentation stack for a matched-objective comparison (Appendix D), our learned encoder matches SimCLR in feature quality *(comparison in red)*. Unlike other approaches, GiRL does not depend on semantic priors communicated via strong data augmentation schemes; diffusion denoising alone induces robust representation learning *(bottom row in **bold**)*.

to extract abstract semantic structure, which is precisely the kind of representation dedicated SSL methods are explicitly trained to capture.

4.2 Ablation Studies

We ablate the stop-gradient and the non-affine LayerNorm on CIFAR-10 at the same compute budget as the main experiment; the two-stage pipeline ablation is in Appendix D.

Stop-gradient vs. weighted losses. A natural alternative is to combine the two objectives with a weighting coefficient, $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{img}} + \lambda \mathcal{L}_{\text{lat}}$, and tune λ. We sweep $\lambda \in \{1, 10^{-3}, 10^{-6}\}$ and find that no setting matches stop-gradient (Table 7a). At $\lambda = 1$, $\mathcal{L}_{\text{lat}}$ dominates and the encoder collapses to near-constant outputs that provide no useful conditioning, recovering the no-encoder baseline. As λ decreases, $\mathcal{L}_{\text{lat}}$ exerts vanishing influence on the encoder and the configuration approaches but never matches stop-gradient. Stop-gradient avoids this hyperparameter search by structurally decoupling the encoder from $\mathcal{L}_{\text{lat}}$.

Non-affine LayerNorm vs. other feature constraints. We compare non-affine LayerNorm against several natural alternatives for constraining the representation distribution: no constraint at all, L2 normalization, BatchNorm, and standard LayerNorm with learnable affine parameters. Across these variants, non-affine LayerNorm consistently yields the lowest FID (Table 7b). In all ablated configurations we observe that the latent diffusion loss $\mathcal{L}_{\text{lat}}$ fails to converge to a low value, consistent with our claim in Section 3.4 that an unconstrained or dynamically-rescaled representation distribution is too irregular for the lightweight latent diffusion model to fit.

5 Conclusion

GiRL jointly trains an encoder, an image diffusion model, and a latent diffusion model over the encoder's output. Decoupled noise schedules, a stop-gradient between the two diffusion objectives, and a non-affine LayerNorm at the projector output, suffice to make joint training stable. On CIFAR-10, CIFAR-100, and ImageNet-100, this system improves unconditional generation over a matched DDPM baseline and produces semantic representations approaching the quality of dedicated SSL methods under a flip-only augmentation budget; when trained under the same data augmentation stack, it outperforms SimCLR. Our results demonstrate the viability of a new paradigm in which representation learning is encompassed within generative modeling, enhancing both.

AI USE STATEMENT

In this work, we did not use generative AI tools for any task in the required-disclosure category: we did not use them to generate synthetic datasets, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, clean or reformat datasets, or interpret results. The remaining required-disclosure tasks (developing theoretical models or conceptual frameworks, formulating mathematical claims, providing ingredients for or writing proofs, translation, and qualitative or thematic data analysis) do not apply to this work. Additionally, we used generative AI tools to edit the manuscript for readability, i.e., to polish grammar, wording, and sentence flow of text written by the authors. No other recommended disclosure tasks (e.g., drafting sections, generating code or figures, literature search, or reference formatting) were performed with generative AI. We have reviewed all AI-assisted edits: every suggested change was checked by the authors against the original text and the reported results to ensure no technical claim, number, or citation was altered. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All main results fix every random seed (data ordering, initialization, and sampling) to 0 so that they are exactly reproducible; robustness to seed choice is verified in Appendix D. Appendix A specifies the architecture of every component (encoder, image diffusion model, and latent diffusion model), the diffusion schedules and conditioning mechanism, the optimization recipe, and the full evaluation protocol for FID and for kNN, linear-probe, and fine-tuning accuracy, with all hyperparameters consolidated in Table 4. The protocol used to select the operating point $\tilde{s}$ is described in Section 4.1 and Appendix A.7, and the search-free operating point $\tilde{s} = 0$ is reported alongside it in Table 2. Appendix B lists compute resources, and Appendix F lists the open-source implementations used for all baselines.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL https://arxiv.org/abs/1607.06450.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020.

Xinlei Chen and Kaiming He. Exploring simple Siamese representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Xiangxiang Chu, Renda Li, and Yong Wang. USP: Unified self-supervised pretraining for image generation and understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18475–18486, 2025.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 8780–8794, 2021.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Jean-Bastien Grill, Florian Strub, Florian Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

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 (NeurIPS)*, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 6840–6851, 2020.

Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022.

Drew A. Hudson, Daniel Zoran, Mateusz Malinowski, Andrew K. Lampinen, Andrew Jaegle, James L. McClelland, Loic Matthey, Felix Hill, and Alexander Lerchner. Soda: Bottleneck diffusion models for representation learning, 2023. URL https://arxiv.org/abs/2311.17901.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Tianhong Li, Huiwen Chang, Shlok Kumar Mishra, Han Zhang, Dina Katabi, and Dilip Krishnan. Mage: Masked generative encoder to unify representation learning and image synthesis, 2023. URL https://arxiv.org/abs/2211.09117.

Tianhong Li, Dina Katabi, and Kaiming He. Return of unconditional generation: A self-supervised representation generation method, 2024. URL https://arxiv.org/abs/2312.03701.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Steven Liu, Tongzhou Wang, David Bau, Jun-Yan Zhu, and Antonio Torralba. Diverse image generation via self-conditioned gans, 2020. URL https://arxiv.org/abs/2006.10728.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

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

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer, 2017. URL https://arxiv.org/abs/1709.07871.

Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10619–10629, 2022.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022. URL https://arxiv.org/abs/2204.06125.

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

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022. URL https://arxiv.org/abs/2205.11487.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2019. URL https://arxiv.org/abs/1907.05600.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

Nick Stracke, Stefan Andreas Baumann, Kolja Bauer, Frank Fundel, and Björn Ommer. Cleandift: Diffusion features without noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 117–127, June 2025.

Yingheng Wang, Yair Schiff, Aaron Gokaslan, Weishen Pan, Fei Wang, Christopher De Sa, and Volodymyr Kuleshov. Infodiffusion: Representation learning using information maximizing diffusion models. In *International Conference on Machine Learning*, pp. 36336–36354. PMLR, 2023.

Weilai Xiang, Hongyu Yang, Di Huang, and Yunhong Wang. Denoising diffusion autoencoders are unified self-supervised learners. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15756–15766, 2023.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think, 2025. URL https://arxiv.org/abs/2410.06940.

Xiao Zhang, Ruoxi Jiang, Rebecca Willett, and Michael Maire. Nested diffusion models using hierarchical latent priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2502–2512, June 2025.

A Implementation Details

This section collects the implementation details required to reproduce GiRL, organized to mirror the structure of Section 3. We first describe the architecture of each component (encoder, image diffusion model, latent diffusion model), then the diffusion schedules and conditioning mechanism, and finally the optimization and evaluation protocols. Unless stated otherwise, all numbers refer to the CIFAR-10 setup; CIFAR-100 and ImageNet-100 use identical hyperparameters with only the dataset substituted.

A.1 Encoder architecture

The encoder is a Vision Transformer (`ViT-S/4`) following the standard configuration of patch size 4, embed dimension 384, depth 12, and 6 attention heads. We use the `timm` implementation as the backbone and use the `[CLS]` token as the global representation. A three-layer projection head then maps the 384-dimensional token embedding to the conditioning dimension $D = \text{hidden_size}_{\text{DiT}}$ (here 384, matching DiT-S):

```
Linear(384, 384) -> LayerNorm -> SiLU
-> Linear(384, 384) -> LayerNorm -> SiLU
-> Linear(384, 384) -> LayerNorm(elementwise_affine=False)
```

The final non-affine LayerNorm is the second of the two stabilization choices discussed in Section 3.4. It anchors the representation distribution to a unit-variance shell, which guarantees that the latent diffusion model always sees the same input statistics regardless of how the encoder evolves.

A.2 Image diffusion model

The image diffusion model is a Diffusion Transformer (`DiT-S/4`) with hidden size 384, depth 12, and 6 heads, parameterized to predict the noise ϵ rather than x_0 or v. The patch size of 4 produces $8 \times 8 = 64$ tokens for 32×32 CIFAR images; for ImageNet-100 at 64×64, the same patch size produces $16 \times 16 = 256$ tokens.

All three are first embedded to D-dimensional vectors (sinusoidal embedding plus a two-layer MLP for t and s; $\mathbf{z}_s$ is already D-dimensional and used as is), then concatenated and projected through a per-block embedder.

```
Linear(3D, D) -> SiLU -> LayerNorm -> Linear(D, D)
```

to produce a single D-dimensional vector that drives adaLN-Zero modulation (Peebles & Xie, 2023). Conditioning on s in addition to t allows the image diffusion model to disambiguate the noise level of the conditioning signal at inference time, which is what enables the inference-time knob discussed in Section 3.

A.3 Latent diffusion model

The latent diffusion model ϵ_ω is a fully-connected residual network operating directly in the D-dimensional representation space, following the SimpleMLP design of RCG (Li et al., 2024). Each residual block has the structure

```
LayerNorm -> SiLU -> Linear(D, D_mid)
+ time_embed_projection
LayerNorm -> SiLU -> Dropout -> zero_init Linear(D_mid, D)
```

with a residual connection from input to output. The final linear layer of every block is zero-initialized so that the network behaves as the identity at initialization, matching the RCG convention. The output head is `LayerNorm` $\to$ `SiLU` $\to$ zero-initialized `Linear` predicting ϵ at the same dimension as the input.

We use the `LiL-S` configuration with model channels 1536, bottleneck channels 1536, 12 residual blocks, time-embedding dimension D, and dropout 0. The total parameter count is approximately

36M, comparable to a small ViT but operating on a single D-dimensional vector rather than a sequence of patch tokens.

A.4 Diffusion schedules

Both the image diffusion process and the latent diffusion process use the standard DDPM linear β schedule of Ho et al. (2020) with $T = S = 1000$ steps, $\beta_{\text{start}} = 10^{-4}$, $\beta_{\text{end}} = 2 \times 10^{-2}$. The variance is fixed (`ModelVarType.FIXED_SMALL`); we do not learn variances. Sampling at evaluation time uses DDPM (Ho et al., 2020) with 1000 steps for the image diffusion and 1000 steps for the latent diffusion unless stated otherwise.

A.5 Optimization

All experiments use AdamW (Loshchilov & Hutter, 2019) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, no ϵ-modification, and a base learning rate of 1×10^{-4}. The DiT and the latent diffusion model use weight decay 0; the encoder doesn't use weight decay. The learning rate is warmed up linearly from 0 to 1×10^{-4} over 5 000 steps and held constant thereafter. Gradients are clipped at a global norm of 1.0 for every component independently. For fair comparison, we fix the total training budget at 204.8M samples ($256 \times 800{,}000$ in our reference configuration), and scale the iteration count proportionally when the global batch size differs.

We maintain an exponential moving average (EMA) of all model weights with decay 0.9999 and use the EMA weights for all evaluation. The EMA buffer is stored in `float32` regardless of the training precision to avoid catastrophic cancellation in the high-decay update.

Training is performed in mixed precision with `torch.amp.autocast` (`dtype=torch.bfloat16`) and a `GradScaler`; `bfloat16` is preferred over `float16` because of its larger dynamic range, which avoids the underflow issues we observed at low noise levels with `float16`. We additionally use `torch.compile` to accelerate training on CIFAR-10 and CIFAR-100, but disable it for ImageNet-100, where we observed that compilation introduced training instability that did not appear in the eager-mode runs. The only data augmentation used during training is a random horizontal flip; we do not use random crop, color jitter, or any other augmentation. All inputs are normalized to $[-1, 1]$.

A.6 Training

We train with AdamW (Loshchilov & Hutter, 2019) at a base learning rate of $1{\times}10^{-4}$ and a 5K-step linear warmup. Both diffusion processes use $T = S = 1000$ steps and a linear noise schedule (Ho et al., 2020). We maintain an exponential moving average (EMA) of all model weights with decay rate 0.9999 and use the EMA weights for evaluation. All models, including ours and all baselines, are trained for exactly 200K iterations with a batch size of 1024.

A.7 Evaluation protocol

FID. We compute FID (Heusel et al., 2017) against the full training set using 50,000 generated samples and the standard Inception-v3 features. Sampling uses the EMA weights of both the image diffusion model and the latent diffusion model, with DDPM-1000 for both processes.

Selection of the early-stop point $\tilde{s}$. All headline FIDs are reported at $\tilde{s} = 300$. This operating point was obtained by searching, independently for each dataset, over the $\tilde{s}$ sweep of Figure 3 against FID; the search converged to the identical value $\tilde{s} = 300$ on CIFAR-10, CIFAR-100, and ImageNet-100, so a single dataset-agnostic operating point reproduces every reported number. We note explicitly that this search is metric-based. For this reason, the main text additionally reports the search-free operating point $\tilde{s} = 0$ (Table 2), which corresponds to full denoising of the latent, uses neither validation nor test data, and still improves over the DDPM baseline on all three datasets.

Random seeds. All main results fix every random seed (data ordering, initialization, and sampling) to 0 for exact reproducibility. To verify robustness to seed choice, we reran GiRL on CIFAR-10 with three additional, randomly chosen seeds, keeping all other hyperparameters identical; across the four runs the standard deviation of kNN and linear-probe accuracy is below 0.3%.

Representation evaluation. We report three standard probes on the encoder's representation. (i) *kNN*: $k = 20/200$ for CIFAR/ImageNet-100, cosine similarity, temperature 0.07 (DINO convention (Caron et al., 2021)), evaluated on the full CIFAR test set with the full CIFAR train set as the memory bank. (ii) *Linear probing*: a single linear classifier trained on the CLS token for 90 epochs with SGD, learning rate swept over $\{0.01, 0.03, 0.1, 0.3\}$, batch size 1024, no weight decay, with the encoder frozen. (iii) *Full fine-tuning*: AdamW for 100 epochs with learning rate 5×10^{-4}, weight decay 0.05, layer-wise learning rate decay 0.65, batch size 1024, following the MAE fine-tuning recipe (He et al., 2022). All probing baselines (SimCLR, MoCo v3, BYOL, MAE, SODA) use the same ViT-S/4 backbone, the same training data, the same number of epochs, and the same probing recipes; only the pretraining objective differs.

A.8 Hyperparameter summary

Table 4 consolidates all hyperparameters used in the main results. Hyperparameters not listed are the defaults of the referenced implementations.

Table 4: Hyperparameter summary for GiRL on CIFAR-10. CIFAR-100 and ImageNet-100 use identical settings; ImageNet-100 differs only in image size (64×64) and dataset.

Component	**Value**
Encoder	ViT-S/4 (patch 4, dim 384, depth 12, heads 6)
Projection head	$3 \times$ Linear(384) with LayerNorm + SiLU; final non-affine LN
Image diffusion model	DiT-S/4 (dim 384, depth 12, heads 6)
Latent diffusion model	LiL-S (dim 1536, bottleneck 1536, depth 12)
Diffusion steps $T = S$	1000
β schedule	linear, $10^{-4} \to 2 \times 10^{-2}$
Prediction target	ϵ (both processes)
Variance type	fixed small
Sampler at evaluation	DDPM-1000
Optimizer	AdamW $(\beta_1, \beta_2) = (0.9, 0.999)$
Base learning rate	1×10^{-4}
Warmup	5,000 steps, linear
Weight decay	DiT: 0 encoder: 0 LiL: 0
Gradient clipping	global norm 1.0
EMA decay	0.9999
Global batch size	1024
Total training steps	200k
Augmentation	horizontal flip only

B Compute Resources

All experiments were conducted on a heterogeneous GPU cluster comprising a mix of accelerator types (e.g., NVIDIA A100, H100, and L40S). Each individual run was assigned to a single homogeneous node, and we did not mix GPU types within a run. Each CIFAR-10 or CIFAR-100 experiment was trained on 4 GPUs with at least 32 GB of memory each, completing in approximately 16 hours. Storage requirements per CIFAR run, including model checkpoints, EMA weights, and intermediate sample artifacts, totaled roughly 100 GB. ImageNet-100 experiments required approximately 2 days of training time on the same multi-GPU setup, with per-GPU memory increased to 80 GB to accommodate the larger image resolution, and storage requirements of approximately 100 GB per run. We did not measure exact wall-clock time across heterogeneous hardware, and the figures above should be interpreted as representative rather than exact.

C Additional Discussion

Principles for stable joint encoder and generator training. Figure 2 diagrams our framework. While a coarse view of its architecture is a straightforward coupling of encoder and diffusion models, our key technical innovation consists of mechanisms to stabilize the training dynamics of this coupled system. Naïve attempts to jointly learn the encoder, the image diffusion model, and the latent diffusion model can yield degenerate results; we employ three design principles, whose implementation Section 3 details, in order to shape robust learning dynamics:

- Both image and latent inputs to the conditional diffusion model are noised, via decoupled noise schedules spanning the full noise range. When constructing a training input, we sample two independent noise timesteps for the image and latent. This forces the encoder to produce latents that are informative *on average* across all noise levels of the image, and to structure the latents so that noising the latents themselves erases less important information (*e.g.,* pixel-level details) about the image.
- Latents are trained to help denoise the image, not to be easy to denoise themselves. The lightweight diffusion model for synthesizing latent features does not backpropagate gradients through the encoder. The latent-conditioned image diffusion model is the encoder's sole source of gradients. Otherwise, the encoder is incentivized to collapse its output into a low-entropy distribution. Modeling of the latent representation space by diffusion is a downstream requirement, not a coequal objective for training the encoder.
- The dynamic range of the latents produced by the encoder must be appropriate for a diffusion noising schedule. Diffusion models presume their target is approximately zero-mean and unit-variance. The conditioning representation must therefore be constrained to a well-conditioned shell that the lightweight diffusion model can reliably fit, while leaving the encoder free to learn whatever internal geometry best serves image synthesis.

Relation to RCG. The closest prior method, RCG (Li et al., 2024), also conditions a diffusion model on a learned representation and trains a representation diffusion model for unconditional sampling, but its encoder is a separately pretrained, frozen MoCo v3 backbone. RCG's encoder is therefore optimized for SSL discriminability rather than for the generator's needs, and the framework cannot speak to whether generation alone could have produced the encoder, because SSL pretraining is a prerequisite. By training the encoder jointly with the generator under a single objective, GiRL both removes this dependency and turns the underlying question into an empirical experiment.

Why is the FID optimum not at $\tilde{s} = 0$? The sampler first partially synthesizes a latent (down to the early-stop point $\tilde{s}$) and then conditionally synthesizes an image given that latent. If our learned latent prior were perfect, matching exactly the distribution of latents the encoder produces for real images, we would expect $\tilde{s} = 0$ (no early stopping) to yield the best FID. In practice the learned prior is imperfect: its samples carry residual approximation error and lie slightly off the manifold the encoder actually produces. It can then be better to let the conditional image model compensate for those imperfections by feeding it a slightly noisy latent ($\tilde{s} \approx 300$–400) rather than insisting that it generate an image exactly corresponding to a noiseless but imperfect latent. Because the generator is trained on $\mathbf{z}_s$ at every noise level (Section 3.5), such noisy latents are in-distribution inputs, and the two-stage ablation in Appendix D confirms that this robustness is a training-time property of the full latent noise schedule.

As an additional sanity check, the $\tilde{s} = 1000$ configuration, which discards the representation entirely by feeding pure Gaussian noise as the conditioning signal, recovers FID nearly identical to the DDPM baseline (17.83 vs 18.08 on CIFAR-10). This confirms that the gains observed at the optimal $\tilde{s}$ are genuinely attributable to the learned representations rather than to incidental architectural or training advantages, since at $\tilde{s} = 1000$ GiRL differs from DDPM only in whether the representation pathway carries useful information.

D Additional Experiments

Isolating the Objective: Matched-Augmentation Comparison. Table 3 compares methods under their own standard recipes, but dedicated SSL methods are not augmentation-agnostic: SimCLR

	CIFAR-10		CIFAR-100	
Method	kNN-20	Linear probe	kNN-20	Linear probe
SODA†	80.33	82.73	39.43	52.57
SimCLR	76.56	84.54	51.90	**57.32**
GiRL (ours)	**83.79**	**86.04**	**52.17**	56.31

Table 5: Matched-augmentation comparison. SimCLR and GiRL are trained with SimCLR's exact augmentation stack under the same ViT-S/4 backbone and compute budget; the only remaining difference is the objective (contrastive vs. pure denoising). Best in **bold** among the matched-augmentation methods. †SODA numbers are under its own strong-augmentation setting (as in Table 3), not re-run under the matched stack; shown for reference only.

loses over 20 linear-probe points when color distortion and cropping are removed (Chen et al., 2020), with similar sensitivity reported for BYOL and DINO. Gaps between flip-only GiRL and strongly augmented SSL baselines therefore conflate two variables: the *loss family* and the *invariances injected by augmentation*. To isolate the first, we ran the controlled comparison directly: we retrained GiRL under SimCLR's exact augmentation stack, so that both methods receive identical invariances, share the same backbone and compute budget, and differ only in the training objective (contrastive vs. pure denoising). GiRL still trains its encoder and both diffusion models entirely from scratch, driven only by the generative denoising objective.

Under this controlled setting, the picture inverts (Table 5): GiRL outperforms SimCLR on three of the four protocol–dataset combinations, **+7.2** kNN and **+1.5** linear-probe points on CIFAR-10, and **+0.3** kNN points on CIFAR-100, while trailing by only 1.0 point on CIFAR-100 linear probing. GiRL moreover matches or exceeds SODA's strong-augmentation numbers on CIFAR-10 kNN-20 and linear probe and on CIFAR-100 kNN-20. The denoising objective is thus not the bottleneck: the gaps in Table 3 reflect our deliberately minimal, flip-only augmentation budget, which we chose to keep the existence claim clean. While GiRL with full augmentation does not fully close the gap to MoCo v3, MAE, or BYOL, contrastive and masked SSL methods have undergone many iterations of engineering refinement; our results are a first demonstration that purely generative representation learning is a competitive alternative worthy of further exploration.

Robustness to seed choice. All main results fix every random seed (data ordering, initialization, and sampling) to 0 so that they are exactly reproducible. To verify that they are also representative, we reran GiRL on CIFAR-10 with three newly and randomly chosen seeds, keeping all other hyperparameters identical. Across these four runs, the standard deviation of kNN and linear-probe accuracy is below 0.3%, indicating that the reported gains are not an artifact of a favorably chosen seed.

Single-stage joint training vs. a two-stage pipeline. GiRL trains the image generator across a full noise schedule on the conditioning latent, so that a single generator learns the whole family of conditional distributions induced by the latent. Because this contribution is a training-time one, its value should be measurable by removing it, so we construct exactly the corresponding two-stage baseline. First, an image generator $G_x(\mathbf{x}_t, \mathbf{z}_0)$ is trained conditioned only on clean latents $\mathbf{z}_0$, i.e., the generator never sees a noisy latent during its own training. Second, a separate latent-diffusion prior $G_z(\mathbf{z}_s)$ is trained on top of this fixed generator. At sampling time, $\mathbf{z}_0$ is drawn from the prior and $\mathbf{x}_0$ is then sampled from the generator conditioned on this (never-seen-noisy) $\mathbf{z}_0$, using the same sampling procedure as GiRL.

The two-stage construction underperforms GiRL on both datasets (Table 6), and the gap holds even before any $\tilde{s}$ selection: at the search-free operating point $\tilde{s} = 0$, GiRL already improves by 3.78 FID on CIFAR-10 and 4.64 FID on CIFAR-100, widening to 5.44 and 7.85 at the reported operating point. We read this as the empirical cost of omitting the full latent noise schedule, and we attribute the mechanism to a distribution shift that single-stage training removes. At inference the conditioning latent is not an encoder output but a *sample from a learned prior*, and such a sample is never exact: it carries residual approximation error and lies slightly off the manifold the encoder actually produces. A generator trained exclusively on clean encoder latents has never seen the kind of input it is asked to consume at test time, and any error in the prior is passed to it as out-of-distribution

Table 6: Single-stage joint training vs. a two-stage pipeline (FID, ↓). Two-stage baseline: a generator $G_x(\mathbf{x}_t, \mathbf{z}_0)$ is trained conditioned only on clean latents, and a separate latent-diffusion prior is trained afterward on top of the fixed generator. The advantage of GiRL is present already at the search-free operating point $\tilde{s} = 0$ and grows at the reported operating point.

Dataset / FID	Two-stage baseline	GiRL @ $\tilde{s} = 0$	GiRL @ $\tilde{s} = 300$	Δ ($\tilde{s} = 0$ / $\tilde{s} = 300$)
CIFAR-10	19.90	16.12	**14.46**	3.78 / 5.44
CIFAR-100	24.58	19.94	**16.73**	4.64 / 7.85

conditioning. Training across the full schedule makes the generator robust to precisely this: imperfect or partially corrupted latents fall inside its training distribution, and it can still produce a coherent image from them. Consistent with this reading, the advantage is present already at $\tilde{s} = 0$, where the two models are asked to do the most similar thing, and grows as more noise is deliberately introduced.

Ablation Studies.

(a) Stop-gradient vs. weighted losses

Configuration	**FID (↓)**	**kNN (↑)**
$\lambda = 1$	19.24	29.74%
$\lambda = 10^{-3}$	15.54	68.94%
$\lambda = 10^{-6}$	16.68	73.42%
No-encoder baseline	18.08	—
Stop Gradient (Ours)	**14.46**	**77.05%**

(b) Feature constraint at projector output

Terminator	**FID (↓)**	**kNN (↑)**
None	22.40	71.59%
L2 normalization	18.45	42.02%
BatchNorm	57.96	48.48%
LayerNorm w/ affine	17.82	73.66%
LayerNorm w/o affine (ours)	**14.46**	**77.05%**

Table 7: Ablation studies on the two stabilization mechanisms in GiRL, evaluated on CIFAR-10. We report both generation quality (FID, lower is better) and representation quality (kNN-20 accuracy, higher is better). (a) Stop-gradient is essential to prevent encoder collapse: no choice of λ in a weighted-loss formulation matches our stop-gradient design on either metric. (b) Non-affine LayerNorm is the most effective constraint on the representation distribution.

E Limitations and Future Directions

Our experiments are limited to three benchmarks: CIFAR-10, CIFAR-100, and ImageNet-100, all at $\leq 64 \times 64$ resolution. We did not evaluate GiRL on ImageNet-1k or higher-resolution datasets, both because of the compute requirements discussed above and because our research question, whether a generation objective alone can drive representation learning, is most cleanly tested at scales where SSL baselines are well-characterized. Whether the findings reported here scale to larger datasets and resolutions, and whether the optimal $\tilde{s}$ at inference shifts with dataset complexity, are open empirical questions that we view as natural next steps.

F Credit to the Open-Source Community

This work would not have been possible without a substantial body of open-source code released by the broader research community. We list below the resources whose implementations we directly built upon or adapted, and we thank the original authors for making their work openly available.

Diffusion model architecture and training. Our image diffusion model is a Diffusion Transformer (Peebles & Xie, 2023), and we adapted the official DiT codebase[1] for both training and sampling. The underlying DDPM training recipe and noising schedule follow Ho et al. (2020), whose reference implementation[2] guided several of our design decisions, including EMA handling and noise-schedule construction.

[1] https://github.com/facebookresearch/DiT

[2] https://github.com/hojonathanho/diffusion

Encoder architecture. The ViT-S/4 encoder used throughout this work follows the Vision Transformer design of Dosovitskiy et al. (2021), with implementation details borrowed from the `timm` library.

Self-supervised baselines. For all representation-learning comparisons in Section 4, we trained baselines on identical data and compute using the following open-source implementations:

- **SimCLR** (Chen et al., 2020), adapted from the official codebase[3].
- **MoCo v3** (Chen et al., 2021), adapted from the official codebase[4].
- **BYOL** (Grill et al., 2020), using the unofficial PyTorch reimplementation[5], which we verified to reproduce the published results within standard variance.
- **MAE** (He et al., 2022), adapted from the official codebase[6].
- **SODA** (Hudson et al., 2023), the diffusion-based SSL baseline most directly comparable to ours, reimplemented from the paper since no official code release was available at the time of our experiments.

Evaluation metrics. FID (Heusel et al., 2017) was computed using the `pytorch-fid`, a faithful PyTorch port of the original TensorFlow reference implementation. kNN classification, linear probing, and full fine-tuning evaluations follow the standard SSL evaluation protocol established by He et al. (2020) and Chen et al. (2020), using the implementations released alongside the corresponding works above.

Datasets. Our experiments use CIFAR-10 and CIFAR-100 (Krizhevsky, 2009) and a 100-class subset of ImageNet (Deng et al., 2009) commonly referred to as ImageNet-100. CIFAR data is accessed through the standard `torchvision` interface; the ImageNet-100 split follows the class list distributed with prior representation-learning work.

[3] https://github.com/google-research/simclr
[4] https://github.com/facebookresearch/moco-v3
[5] https://github.com/lucidrains/byol-pytorch
[6] https://github.com/facebookresearch/mae