REPFLOW: RECIPROCAL SUPERVISION IMPROVES GENERATION AND REPRESENTATION IN FLOW MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models learn visual structure through denoising, yet their internal states are entangled with both noise level and network depth, making it difficult to obtain a stable visual representation from the generator itself. We introduce RepFlow, which learns such a representation from the generator's evolving computation and uses it to guide generation. Specifically, a separate timestep-free encoder is trained, through a timestep-conditioned predictor, to recover generator states across depths and noise levels from masked clean images. By excluding the reference noise and masked-out content from the encoder's input, we encourage the encoder to distill visual information that is recoverable from the visible image context and predictive of generator states. The representation learned from the generator's evolving states is then fed back to guide and improve the generator, whose updated states provide supervision for further representation learning, forming a reciprocal learning process. This reciprocal process improves multi-step generation and representation quality, as measured by frozen linear probing on ImageNet with latent-space SiT and pixel-space JiT, without an externally pretrained representation teacher. Across the two unconditional settings, FID decreases by 19.2–40.8% relative to native training, while linear-probe accuracy improves by 6.80–10.04 percentage points over the best searched raw generator features. The learned representation also serves as a distributional metric for one-step JiT post-training, extending its role from instance-level alignment to distribution-level supervision.

1 INTRODUCTION

Generative modeling (Ho et al., 2020; Lipman et al., 2022; Peebles & Xie, 2023; Ma et al., 2024) and representation learning (Chen et al., 2020; He et al., 2022; Assran et al., 2023) impose different demands on an image model. A generator must retain instance-specific detail to model the data distribution, whereas a representation learner is rewarded for discarding nuisance variation while preserving information useful for transfer. Both, however, must organize the same underlying visual structure. This shared requirement creates an opportunity to learn representations that are useful both for understanding images and for improving the process that generates them.

Prior work establishes that information can flow in both directions. Features extracted from pretrained generators support recognition and transfer (Gabeur et al., 2026; Baranchuk et al., 2021; Tang et al., 2023; Stracke et al., 2025; Xiang et al., 2023), while pretrained visual representations can guide generative optimization and synthesis (Yu et al., 2025; Zheng et al., 2026). Self-representation alignment further shows that a generator can obtain useful guidance from its own internal states (Jiang et al., 2025). We investigate how such guidance can be formed through a separate image representation learned alongside the generator. The question is how to turn the generator's evolving computation into a representation that can, in turn, guide its subsequent learning (Figure 1).

The difficulty lies in the form of information available from a generator. A flow or diffusion model exposes a family of internal states indexed by noise level and network depth, each reflecting image content together with uncertainty and stage-specific denoising computation. A feature useful at one layer and timestep may behave differently elsewhere in the trajectory (Baranchuk et al., 2021; Tang et al., 2023; Stracke et al., 2025; Xiang et al., 2023). These states nevertheless provide related observations of the same underlying image. Viewed as learning targets, their value depends on what

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.

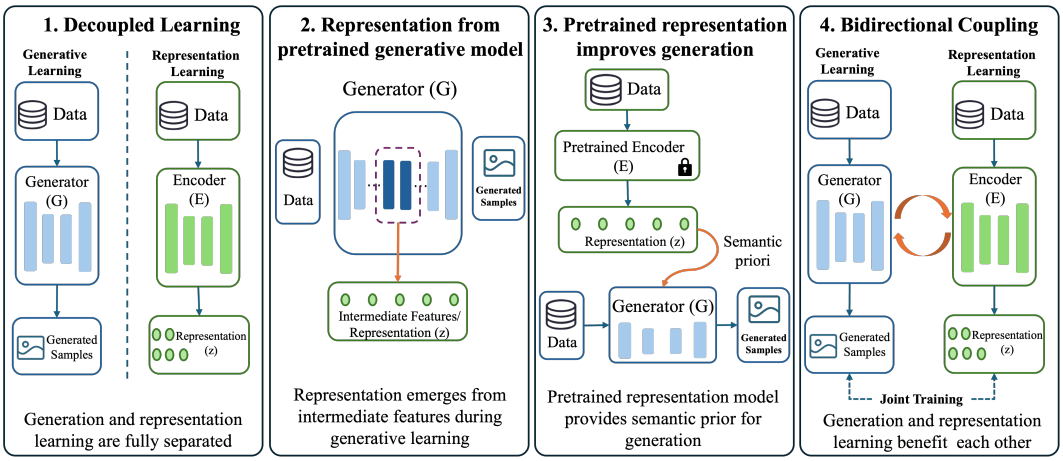


Figure 1: Connections between generation and representation learning. Two common strategies extract features from a pretrained generator or use a pretrained representation to guide generation. RepFlow learns a clean-image representation from the generator's evolving states and uses it to shape subsequent generative learning.

an image encoder must learn to predict them. This perspective makes it possible to use the diversity of generative states as supervision for a single clean-image representation.

We introduce **RepFlow**, based on the idea that a generator can be guided by a representation learned from the predictable structure of its own computation. A separate student predicts generator states from several depths at a sampled timestep, given only a masked clean image. Since the student observes neither the reference noise nor the masked image content, reducing prediction error rewards information recoverable from visible context. Across training, the same encoder must support prediction across depths and noise levels, while timestep conditioning is confined to the predictor. The student thus learns through a family of contextual prediction tasks and produces a representation that can be extracted without selecting a generator layer or timestep.

This learned representation becomes part of the generator's own training signal. The student processes the full clean image and supplies an alignment target for a generator state. The returned target has been shaped by a different learning problem: predicting generative computation from restricted image context. It therefore provides a clean-image reference whose input does not depend on the current noise realization or denoising stage. As this reference guides the generator, the resulting updates change the states from which the student subsequently learns. RepFlow couples the two models through this evolving supervision, while retaining their separate parameters and the generator's native objective. The representation used to guide generation is learned within the same process, without an externally pretrained representation teacher.

On ImageNet, RepFlow improves both multi-step generation and frozen linear probing with latent-space SiT (Ma et al., 2024) and pixel-space JiT (Li & He, 2026). Predicting generator states already yields a student that outperforms the best searched raw generator features; reciprocal training further improves both the student and the generator. In unconditional experiments, RepFlow reduces FID from 32.83 to 19.43 on SiT and from 35.58 to 28.76 on JiT. Frozen linear-probe accuracy improves by 10.04 and 6.80 percentage points, respectively, over the best searched raw features of each generator. Class-conditional results are competitive with REPA (Yu et al., 2025) and SRA (Jiang et al., 2025). Controlled ablations favor generator-state targets over masked-pixel reconstruction and multi-depth supervision over a single-depth target, supporting the role of generative states as a source of predictive supervision.

The same representation also serves as a distributional metric for one-step JiT post-training (Yang et al., 2026): under matched initialization and optimization budgets, replacing a pretrained MAE feature with the RepFlow student lowers FID from 6.34 to 3.14. Its use thus extends from instance-level supervision during reciprocal learning to distribution-level supervision afterward.

Our contributions are:

- We introduce RepFlow, a reciprocal learning framework in which the representation guiding a flow generator is learned from its own evolving computation through a separate predictive encoder, without an externally pretrained representation teacher.
- We show that stage-dependent generative states can supervise a timestep-free encoder whose features outperform the best searched raw generator features in linear probing.
- We demonstrate improvements in multi-step generation across latent-space SiT and pixel-space JiT, and show through matched one-step JiT post-training that the learned representation also provides effective distribution-level supervision.

2 PRELIMINARIES

2.1 FLOW-BASED GENERATIVE MODELING

Let $x_0 \sim p_{\text{data}}$ denote a clean sample in the generator's modeling space and $x_1 \sim p_{\text{ref}}$ an independent noise sample. In our experiments, x_0 consists of VAE latents for SiT and image pixels for JiT. We define an interpolation

$$x_t = \alpha_t x_0 + \sigma_t x_1, \qquad t \in [0, 1], \tag{1}$$

where $(\alpha_0, \sigma_0) = (1, 0)$ and $(\alpha_1, \sigma_1) = (0, 1)$, so that $t = 0$ corresponds to data and $t = 1$ to noise. The associated conditional velocity is

$$u_t(x_0, x_1) = \frac{\mathrm{d}x_t}{\mathrm{d}t} = \dot{\alpha}_t x_0 + \dot{\sigma}_t x_1. \tag{2}$$

SiT (Ma et al., 2024) directly parameterizes a velocity field $v_\theta(x_t, t)$ and minimizes

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{x_0, x_1, t}\left[\|v_\theta(x_t, t) - u_t(x_0, x_1)\|_2^2\right]. \tag{3}$$

JiT (Li & He, 2026) directly predicts clean data, $\hat{x}_{0,\theta} = G_\theta(x_t, t)$. For the linear path $\alpha_t = 1 - t$ and $\sigma_t = t$ used in our experiments, its native velocity-space loss can be written as a weighted reconstruction objective:

$$\mathcal{L}_{\text{JiT}} = \mathbb{E}_{x_0, x_1, t}\left[w(t)\,\|G_\theta(x_t, t) - x_0\|_2^2\right], \tag{4}$$

where $w(t) = \max(t, t_{\min})^{-2}$ and $t_{\min}$ is the denominator-clipping threshold specified in Appendix C. Expectations use each backbone's native timestep sampling distribution. We write G_θ for either generative backbone, $\mathcal{L}_{\text{gen}}$ for its native objective, and $h_\ell^G(x_t, t)$ for its internal state at layer ℓ.

2.2 PREDICTIVE REPRESENTATION LEARNING

Masked predictive learning trains an encoder by predicting targets from partial image observations (Baevski et al., 2022; Assran et al., 2023; He et al., 2022). Let x denote the clean RGB image corresponding to x_0, and let x^{mask} be a masked view. The encoder U_ϕ produces

$$u^{\text{mask}}(x) = U_\phi\left(x^{\text{mask}}\right). \tag{5}$$

Given a target feature y and an optional conditioning variable c, a predictor P_ψ is trained through

$$\mathcal{L}_{\text{pred}} = \mathbb{E}\left[\ell\left(P_\psi\left(u^{\text{mask}}(x), c\right), \text{sg}(y)\right)\right], \tag{6}$$

where ℓ is a prediction loss and $\text{sg}(\cdot)$ stops gradients through the target in this objective. The target-producing model may still be updated by other objectives. In RepFlow, y is constructed from generator states and c is the generative timestep, supplied only to the predictor. This keeps the encoder's input independent of the target's generative stage.

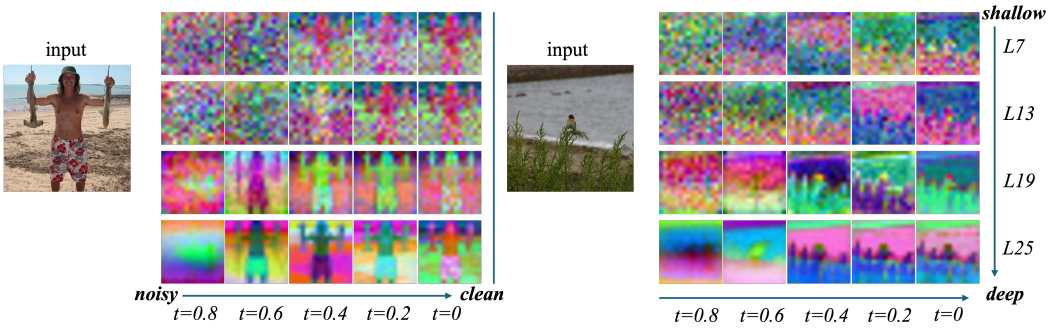


Figure 2: Generator states across network depth and noise level. The same image induces different feature maps at different stages of generative computation. RepFlow uses these states as prediction targets for a clean-image encoder.

3 METHOD

3.1 FROM GENERATIVE STATES TO PREDICTIVE TARGETS

RepFlow learns the representation that guides a generator from the predictable structure of that generator's own computation. Internal states at different depths and noise levels define related prediction tasks about the same image (Figure 2). We train a separate image encoder through these tasks, then use its full-image features to guide the generator. The resulting updates change the states that supervise subsequent representation learning.

The predictive task restricts what the encoder can observe. Let $\mathcal{M}$ denote the masked spatial positions, and let x^{mask} contain the visible image patches and their positions. For a fixed generator, let $Y_{t,i}$ collect normalized generator features from selected depths at a masked position i. Under a squared-error surrogate, the Bayes-optimal prediction satisfies

$$F_i^\star(x^{\mathrm{mask}}, t) = \mathbb{E}\big[Y_{t,i} \mid x^{\mathrm{mask}}, t\big], \qquad i \in \mathcal{M}. \tag{7}$$

Visible context is therefore useful insofar as it explains target variation beyond the timestep alone. Only the variation unresolved by this conditioning contributes to irreducible squared prediction error. Supplying t to the predictor allows prediction to vary with the generative stage while the encoder depends only on image observations. Appendix B gives the corresponding risk decomposition. This squared-loss analysis explains the prediction problem; the implemented objective uses Smooth L1, as specified below.

3.2 LEARNING THE SUPERVISORY REPRESENTATION

The generator G_θ and encoder U_ϕ have separate parameters. We use x for the clean RGB image and x_0 for its representation in the generator's modeling space: a VAE latent for SiT and pixels for JiT. The generator receives the corresponding noised input x_t, while the encoder receives a masked view of x. In both instantiations, generator features and full-image encoder features share an N-position spatial grid.

Let $\mathcal{D} = \{\ell_1, \ldots, \ell_K\}$ be the selected generator depths. At each spatial position i, the predictive target is

$$Y_{t,i} = \mathrm{Concat}_{\ell \in \mathcal{D}} \left[\mathrm{LN}\left(h^G_{\ell,i}(x_t, t)\right)\right], \tag{8}$$

where LN denotes channel-wise layer normalization at each token. Normalization places features from different depths on a common scale before concatenation. Class conditioning, when present, is implicit in the generator states; the encoder receives no class label, reference noise, or timestep.

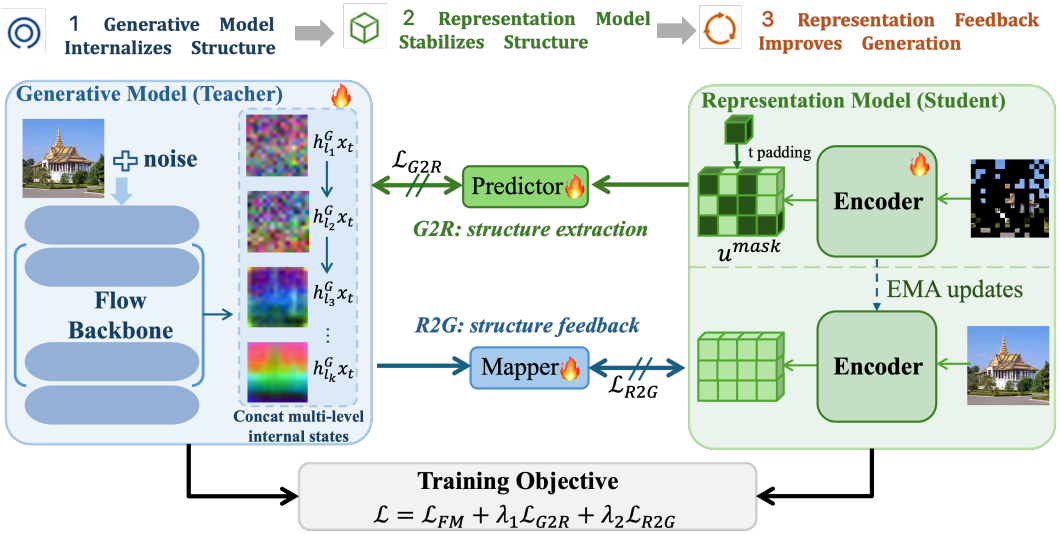


Figure 3: RepFlow. Generator states supervise masked prediction through G2R, while full-image features from an EMA encoder guide an early generator state through R2G. Stopped targets separate the gradient paths, and the models interact through the targets used in subsequent updates. The native generator objective is $\mathcal{L}_{\mathrm{FM}}$ for SiT and $\mathcal{L}_{\mathrm{JiT}}$ for JiT. Flames indicate trainable modules; double slashes indicate stopped gradients.

The encoder produces a masked-view code $u_\phi^{\mathrm{mask}}(x)$. A timestep-conditioned predictor reconstructs a full spatial grid of target predictions, $\hat{Y}_t = P_\psi(u_\phi^{\mathrm{mask}}(x), t)$. The encoder features supplied to this predictor are specified for each setting in Appendix C. We supervise only masked positions:

$$\mathcal{L}_{\mathrm{G2R}} = \mathbb{E}\left[\frac{1}{|\mathcal{M}|}\sum_{i\in\mathcal{M}} \ell_{\mathrm{SL1}}\left(\hat{Y}_{t,i}, \mathrm{sg}(Y_{t,i})\right)\right], \tag{9}$$

where ℓ_{SL1} is Smooth L1 with $\beta = 1$, averaged over feature channels. Expectations include the sampled data, reference noise, timestep, and mask. The resulting loss updates only the encoder and predictor.

Across training, the encoder supports prediction at several depths and sampled noise levels from image context alone. The predictor handles timestep-dependent decoding, while the encoder learns the information useful across these prediction tasks. The generator continues to evolve under the training objective below, so the student's targets evolve with it.

3.3 Reciprocal Training

The learned representation supplies an auxiliary target for the generator (Figure 3). An EMA encoder $U_{\bar{\phi}}$ processes the full clean image. For each position i, we define its feature and the projected generator feature as

$$V_i(x) = \left[U_{\bar{\phi}}(x)\right]_i, \qquad Z_i(x_t, t) = g_\omega\left(h^G_{\ell_s,i}(x_t, t)\right), \tag{10}$$

where ℓ_s is an early generator layer and g_ω maps its features to the encoder's feature dimension. Placing the alignment at an early layer allows the resulting updates to affect the states processed by subsequent generator blocks.

We use token-wise cosine alignment (Yu et al., 2025), with the supervisory target learned through the predictive task above. Writing $\nu(a) = a/\|a\|_2$ for feature normalization,

$$\mathcal{L}_{\mathrm{R2G}} = \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\left(1 - \langle \nu(Z_i), \mathrm{sg}(\nu(V_i))\rangle\right)\right]. \tag{11}$$

The target is evaluated on the full image and is independent of the current noise realization and timestep. Its parameters are nevertheless shaped by predicting generator states from masked context.

The complete training objective is

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_{\text{gen}} + \lambda_1 \mathcal{L}_{\text{G2R}} + \lambda_2 \mathcal{L}_{\text{R2G}}, \tag{12}$$

where $\mathcal{L}_{\text{gen}}$ is the native SiT or JiT objective. The stopped targets give

$$\nabla_\theta \mathcal{L}_{\text{joint}} = \nabla_\theta \mathcal{L}_{\text{gen}} + \lambda_2 \nabla_\theta \mathcal{L}_{\text{R2G}}, \qquad \nabla_\phi \mathcal{L}_{\text{joint}} = \lambda_1 \nabla_\phi \mathcal{L}_{\text{G2R}}. \tag{13}$$

The predictor and projection head are optimized through G2R and R2G, respectively. After each online update, we update $\bar{\phi} \leftarrow m\bar{\phi} + (1-m)\phi$. We use $\lambda_1 = 1$, $\lambda_2 = 0.1$, and $m = 0.9999$; architectural and optimization settings are given in Appendix C.

Reciprocity arises through the supervision available at the next update. Updating θ changes the states that the student must predict; updating ϕ changes the EMA features that guide subsequent generator updates. The encoder thereby turns the generator's computation into a clean-image training target, while its own prediction task changes as the generator learns. Appendix B formalizes these update dependencies.

For representation evaluation, we average the final-layer tokens from $U_{\bar{\phi}}(x)$. This requires a single full-image encoder pass, with no generator, predictor, or layer–timestep selection.

3.4 Representation-guided One-step Post-training

Starting from a jointly trained JiT checkpoint, we freeze the EMA encoder and reuse its feature space for one-step distribution matching. A terminal-noise input produces $\hat{x} = G_\theta(x_1, y, 1)$, where y denotes the class condition. Following FD-loss (Yang et al., 2026), we define $r(x) = \text{Pool}(U_{\bar{\phi}}(x))$ and compute the real feature mean and covariance (μ_r, Σ_r) once from $\mathcal{D}_{\text{train}}$. Generated statistics (μ_g, Σ_g) are tracked through an EMA of first and second moments. The objective is

$$\begin{aligned} \mathcal{L}_{\text{FD}} &= \|\mu_r - \mu_g\|_2^2 + d_{\text{cov}}(\Sigma_r, \Sigma_g), \\ d_{\text{cov}}(\Sigma_r, \Sigma_g) &= \text{Tr}\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r^{1/2}\Sigma_g\Sigma_r^{1/2}\right)^{1/2}\right). \end{aligned} \tag{14}$$

Real statistics and historical generated moments remain fixed during backpropagation; gradients through the current generated-image features update the generator while the encoder parameters remain frozen. This reuses the representation learned by reciprocal training as a distributional metric, extending its role beyond instance-level alignment.

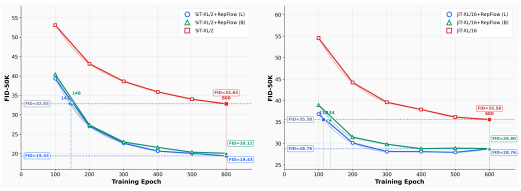
4 Experiments

Our experiments test whether a representation learned from a generator can improve the generator that supplies its training targets. We first evaluate sample quality and frozen representations under reciprocal training, then use controlled ablations to examine how feedback and target choice affect both. Class-conditional generation and one-step post-training test the utility of the learned supervision in the presence of class labels and under a different training objective.

4.1 Experimental Setup

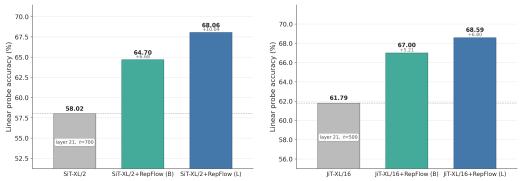
We instantiate RepFlow with latent-space SiT (Ma et al., 2024) and pixel-space JiT (Li & He, 2026) on ImageNet (Russakovsky et al., 2015). Each native/RepFlow comparison matches the generative backbone, data, epoch budget, sampler, and evaluator. We report FID over 50,000 generated samples (FID-50K) (Heusel et al., 2017) and ImageNet top-1 accuracy from frozen linear probing. All of our generation evaluations use CFG $= 1$.

RepFlow-B and RepFlow-L denote the use of a separate ViT-B or ViT-L student (Dosovitskiy et al., 2020), respectively; the generator backbone is unchanged. For the raw-generator representation baseline, we search layers and timesteps and select block 21 at implementation timestep $\tau = 0.7$ for SiT and $\tau = 0.5$ for JiT, where $\tau = 1 - t$. RepFlow uses the final-layer, full-view EMA student feature in a single clean-image forward pass.



(a) SiT (b) JiT

Figure 4: Unconditional ImageNet FID-50K across training epochs. Annotations mark when RepFlow reaches the native 600-epoch FID.



(a) SiT (b) JiT

Figure 5: Frozen ImageNet linear probing. Raw generator features use the best searched layer and timestep; RepFlow uses the final-layer, full-view student feature.

Class-conditional comparisons include REPA (Yu et al., 2025), which uses a pretrained DINOv2 representation target, and SRA (Jiang et al., 2025), which aligns states within the generator. Appendix C details the architectures, optimization, evaluation protocols, and computational cost. Training progress below is compared in epochs; the student adds computational cost per update.

4.2 Unconditional Generation and Representation

RepFlow improves generation on both backbones (Figure 4). After 600 epochs, RepFlow-L reduces SiT FID from 32.83 to 19.43 (40.8%) and JiT FID from 35.58 to 28.76 (19.2%). RepFlow-B reaches 20.12 and 28.80, respectively, so most of the generative gain is already present with the smaller student. The benefit also appears before the final checkpoint: RepFlow-L reaches the native 600-epoch FID after approximately 143 epochs on SiT and 114 epochs on JiT.

The same training runs yield stronger frozen representations (Figure 5). On SiT, linear-probe accuracy increases from 58.02% for the best searched raw generator feature to 64.70% with RepFlow-B and 68.06% with RepFlow-L. On JiT, the corresponding accuracies are 61.79%, 67.00%, and 68.59%. RepFlow-L therefore improves on the best searched raw features by 10.04 and 6.80 percentage points while requiring no layer–timestep selection at inference. Increasing student capacity improves linear probing on both backbones, although JiT FID changes little between the two student sizes. Recognition accuracy and effectiveness as generative supervision thus need not improve to the same extent.

4.3 Analyzing Reciprocal Supervision

Table 1 evaluates all student variants with ViT-L. For linear probing, Native uses its best searched raw generator feature, whereas the remaining variants use the frozen student. The single-level targets are block 19 for SiT and block 20 for JiT; the multi-level target set is $\mathcal{D} = \{5, 12, 19, 26\}$.

Table 1: Unconditional ImageNet ablations at 600 epochs. LP denotes linear-probe accuracy (%).

		SiT		JiT	
Component	Variant	FID ↓	LP ↑	FID ↓	LP ↑
Reciprocal path	Native / raw state	32.83	58.02	35.58	61.79
	G2R only	32.83	65.35	35.58	66.19
	G2R+R2G	**19.43**	**68.06**	**28.76**	**68.59**
G2R target	Pixel reconstruction	25.23	57.23	31.56	59.88
	Generator states	**19.43**	**68.06**	**28.76**	**68.59**
Target depth	Single level	20.71	65.26	30.44	67.17
	Multi-level	**19.43**	**68.06**	**28.76**	**68.59**

Table 1 first isolates the effect of returning the learned representation to the generator. Disabling R2G ($\lambda_2 = 0$) leaves the generator's native update unchanged because G2R uses stopped generator targets. Under matched initialization and stochasticity, Native and G2R-only therefore share the same generator trajectory. Even in this setting, masked prediction raises linear-probe accuracy from 58.02% to 65.35% on SiT and from 61.79% to 66.19% on JiT. Learning from the generator's evolving states thus produces a stronger representation than reading out its best searched raw feature.

Enabling feedback improves both models: FID decreases by 13.40 on SiT and 6.82 on JiT, while student accuracy increases by a further 2.71 and 2.40 percentage points. R2G supplies no direct gradient to the student, so its influence on representation learning passes through the generator states used as G2R targets. The additional student gains are therefore consistent with feedback improving the supervision available for subsequent representation learning.

Target choice also affects both outcomes. We replace generator-state prediction with MAE-style masked-pixel reconstruction while retaining the student, masking ratio, R2G branch, data, and training budget. Pixel reconstruction itself improves native FID, reaching 25.23 on SiT and 31.56 on JiT. Predicting generator states further reduces these values to 19.43 and 28.76, while raising student accuracy from 57.23% to 68.06% and from 59.88% to 68.59%, respectively. This control shows that the target used to train the student affects both the representation it acquires and the quality of the generative supervision it supplies.

Supervision across depth provides a further gain. Relative to the strongest evaluated single-depth target, multi-level supervision lowers FID by 1.28 on SiT and 1.68 on JiT and improves linear-probe accuracy by 2.80 and 1.42 percentage points. These results favor learning from several stages of generator computation in the tested settings, while retaining a single student representation at inference.

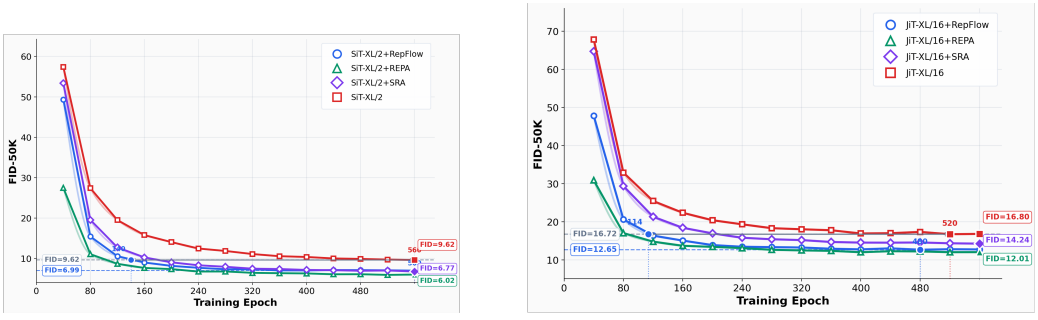
4.4 Class-Conditional Generation

RepFlow also improves generation when class labels already provide semantic conditioning (Table 2 and Figure 6). Table 2 reports FID, spatial FID (sFID), IS, precision (Pre.), and recall (Rec.). RepFlow and SRA use no externally pretrained representation teacher, whereas REPA uses pretrained DINOv2. At 600 epochs, RepFlow reduces SiT FID from 9.62 to 6.99 (27.3%) and JiT FID from 16.80 to 12.65 (24.7%). The learned image-level supervision therefore remains useful in the presence of class conditioning.

Table 2: Class-conditional ImageNet at 600 epochs. Bold/underline denote the best/second-best results per backbone; ties share formatting.

Method	FID↓	sFID↓	IS↑	Pre.↑	Rec.↑
SiT	9.62	5.31	123.0	0.67	0.68
SiT + SRA	<u>6.77</u>	**4.91**	142.0	<u>0.69</u>	0.68
SiT + RepFlow	6.99	5.17	<u>147.9</u>	0.68	**0.70**
SiT + REPA	**6.02**	<u>5.05</u>	**159.3**	**0.70**	<u>0.69</u>
JiT	16.80	<u>6.93</u>	93.7	0.61	<u>0.67</u>
JiT + SRA	14.24	7.04	101.8	<u>0.63</u>	<u>0.67</u>
JiT + RepFlow	<u>12.65</u>	**6.52**	<u>116.4</u>	<u>0.63</u>	**0.68**
JiT + REPA	**12.01**	7.93	**122.0**	**0.64**	<u>0.67</u>

Against SRA, RepFlow has slightly higher SiT FID (6.99 versus 6.77), with higher IS and recall; on JiT, it achieves lower FID and sFID and higher IS and recall. REPA retains lower FID on both backbones, with gaps of 0.97 on SiT and 0.64 on JiT, while RepFlow achieves higher recall. These comparisons place the representation learned within generative training close to pretrained-encoder guidance in FID, with different tradeoffs across the other metrics.



(a) SiT (b) JiT

Figure 6: Class-conditional ImageNet FID-50K across training epochs for native training, RepFlow, SRA, and REPA. RepFlow learns its representation target concurrently with the generator.

4.5 Representation-Guided One-Step Post-Training

We next test whether a representation learned through instance-level reciprocal supervision can also guide distribution matching after joint training. Starting from the same 400-epoch RepFlow-JiT checkpoint, we freeze either the RepFlow ViT-L/16 student or an ImageNet-pretrained MAE ViT-L/16 encoder, matching the student architecture, and update only the generator for 30 epochs using FD-loss (Yang et al., 2026), as described in Section 3. The data, sampling protocol, optimizer, schedule, and CFG $= 1$ are shared; only the feature space changes.

Table 3: Class-conditional one-step ImageNet generation. The final two rows share the generator checkpoint and post-training protocol. Ext. Enc. lists externally pretrained representation encoders. FDr^6 (Yang et al., 2026) averages normalized Fréchet distances in Inception (Szegedy et al., 2016), ConvNeXt (Liu et al., 2022), DINOv2 (Oquab et al., 2023), MAE (He et al., 2022), SigLIP (Zhai et al., 2023), and CLIP (Radford et al., 2021) spaces. † denotes FD-SIM training in SigLIP, Inception, and MAE spaces. RepFlow-student FD uses its own learned encoder.

Method	Space	Epoch	NFE	Ext. Enc.	FID ↓	FDr^6 ↓	IS ↑	Prec. ↑	Rec. ↑
One-step generative models									
iCT-XL/2 (Song & Dhariwal, 2024)	latent	–	1	No	34.24	–	–	–	–
Shortcut-XL/2 (Frans et al., 2025)	latent	250	1	No	10.60	–	–	–	–
MeanFlow (Geng et al., 2025)	latent	240	1	No	3.43	–	–	–	–
iMF-XL (Geng et al., 2026)	latent	800	1	No	1.82	8.39	278.9	0.78	0.63
Drift-L (latent) (Deng et al., 2026)	latent	1280	1	latent-MAE	1.53	10.92	257.2	0.79	0.63
Drift-L (pixel) (Deng et al., 2026)	pixel	640	1	pixel-MAE	1.43	10.51	305.8	0.81	0.60
pMF-L (Lu et al., 2026)	pixel	320	1	ConvNeXt-v2+VGG	2.72	9.09	261.7	0.81	0.56
pMF-H (Lu et al., 2026)	pixel	360	1	ConvNeXt-v2+VGG	2.29	6.87	267.2	0.80	0.59
Distilled or post-trained one-step models									
JiT-L + FD-loss† (Yang et al., 2026)	pixel	600+100	1	(SigLIP+Incep.+MAE)	0.77	3.24	317.3	0.77	0.66
iMF-XL + FD-loss† (Yang et al., 2026)	latent	800+100	1	(SigLIP+Incep.+MAE)	0.76	2.45	301.3	0.77	0.67
pMF-L + FD-loss† (Yang et al., 2026)	pixel	320+100	1	(SigLIP+Incep.+MAE)	0.78	2.09	309.2	0.76	0.67
RepFlow-JiT									
RepFlow-JiT-XL, naive one-step	pixel	400	1	No	268.44	217.75	2.42	–	–
RepFlow-JiT-XL + MAE FD	pixel	400+30	1	MAE	6.34	9.33	286.1	0.77	0.56
RepFlow-JiT-XL + RepFlow-student FD	pixel	400+30	1	No	3.14	8.06	312.8	0.82	0.56

Direct one-step sampling from the starting checkpoint gives FID 268.44 (Table 3). After post-training, replacing MAE with the RepFlow student lowers FID from 6.34 to 3.14, a 50.5% reduction, and FDr^6 from 9.33 to 8.06. IS and precision also increase, while recall remains 0.56. The student thus retains supervisory value after it is frozen and its role changes from instance-level feedback to a distribution-level objective.

The remaining systems in Table 3 provide context across different architectures, training budgets, and objectives. The daggered FD-loss systems train in feature spaces that also appear in the reported FID/FDr^6 evaluations. RepFlow-student FD uses a separate learned feature space and none of the six evaluation encoders in its objective. Figure 7 shows selected samples from the one-step RepFlow-JiT-XL model post-trained with RepFlow-student FD.



Figure 7: Selected one-step samples from RepFlow-JiT-XL with RepFlow-student FD on ImageNet at 256×256 resolution.

5 Conclusion

RepFlow learns a generator's representation guidance from its own evolving computation. Predicting states across depth and noise level from masked clean images trains a separate, timestep-free encoder; feedback from this encoder then changes the states that supervise subsequent representation learning. The supervisory signal is therefore an outcome of the same training process it helps guide.

On ImageNet, this process improves multi-step generation and frozen linear probing with both latent-space SiT and pixel-space JiT, without an externally pretrained representation teacher. Ablations show that feedback benefits both models and favor generator-state prediction across multiple depths over the tested pixel-reconstruction and single-depth alternatives. Once frozen, the learned student also supports one-step JiT distribution matching, extending the value of the representation beyond the joint training objective that produced it.

6 AI USE STATEMENT.

Generative AI tools were used during manuscript preparation to draft and edit text for readability, refine the presentation of existing material, and assist with formatting tables, citations, and document layout. They were not used for generating or cleaning data, developing conceptual or theoretical models, translating research content, analyzing data, or interpreting results. The authors reviewed and verified all AI-assisted work and take full responsibility for the final text, claims, results, and artifacts in this paper.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15619–15629, 2023.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. In *International conference on machine learning*, pp. 1298–1312. PMLR, 2022.

Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. *arXiv preprint arXiv:2112.03126*, 2021.

Nicholas M Boffi, Michael S Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *arXiv preprint arXiv:2406.07507*, 2024.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Hila Chefer, Patrick Esser, Dominik Lorenz, Dustin Podell, Vikash Raja, Vinh Tong, Antonio Torralba, and Robin Rombach. Self-supervised flow matching for scalable multi-modal synthesis. *arXiv*, 2026.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021.

Xinlei Chen, Zhuang Liu, Saining Xie, and Kaiming He. Deconstructing denoising diffusion models for self-supervised learning. In *International Conference on Learning Representations*, volume 2025, pp. 55458–55472, 2025.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770*, 2026.

Chen Dengsheng, Jie Hu, Xiaoming Wei, and Enhua Wu. Denoising with a joint-embedding predictive architecture. In *International Conference on Learning Representations*, volume 2025, pp. 22479–22517, 2025.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, volume 2025, pp. 34668–34684, 2025.

Valentin Gabeur, Shangbang Long, Songyou Peng, Paul Voigtlaender, Shuyang Sun, Yanan Bao, Karen Truong, Zhicheng Wang, Wenlei Zhou, Jonathan T Barron, et al. Image generators are generalist vision learners. *arXiv preprint arXiv:2604.20329*, 2026.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *NeurIPS*, 2025.

Zhengyang Geng, Yiyang Lu, Zongze Wu, Eli Shechtman, J Zico Kolter, and Kaiming He. Improved mean flows: On the challenges of fastforward generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 30467–30476, 2026.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan 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 *NeurIPS*, 2020.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 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. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Dengyang Jiang, Mengmeng Wang, Liuzhuozheng Li, Lei Zhang, Haoyu Wang, Wei Wei, Guang Dai, Yanning Zhang, and Jingdong Wang. No other representation component is needed: Diffusion transformers can provide representation guidance by themselves. *arXiv preprint arXiv:2505.02831*, 2025.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36115–36125, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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.

Yiyang Lu, Susie Lu, Qiao Sun, Hanhong Zhao, Zhicheng Jiang, Xianbang Wang, Tianhong Li, Zhengyang Geng, and Kaiming He. One-step latent-free image generation with pixel mean flows. *arXiv preprint arXiv:2601.22158*, 2026.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

Soumik Mukhopadhyay, Matthew Gwilliam, Yosuke Yamaguchi, Vatsal Agarwal, Namitha Padmanabhan, Archana Swaminathan, Tianyi Zhou, Jun Ohya, and Abhinav Shrivastava. Do text-free diffusion models learn discriminative visual representations? In *European Conference on Computer Vision*, pp. 253–272. Springer, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

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.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Yuxi Ren, Xin Xia, Yanzuo Lu, Jiacheng Zhang, Jie Wu, Pan Xie, Xing Wang, and Xuefeng Xiao. Hyper-sd: Trajectory segmented consistency model for efficient image synthesis. *Advances in neural information processing systems*, 37:117340–117362, 2024.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast high-resolution image synthesis with latent adversarial diffusion distillation. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–11, 2024a.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2024b.

Jaeyo Shin, Jiwook Kim, and Hyunjung Shim. Representation alignment for just image transformers is not easier than you think. *arXiv preprint arXiv:2603.14366*, 2026.

Jaskirat Singh, Xingjian Leng, Zongze Wu, Liang Zheng, Richard Zhang, Eli Shechtman, and Saining Xie. What matters for representation alignment: Global information or spatial structure? *arXiv preprint arXiv:2512.10794*, 2025.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *International Conference on Learning Representations*, volume 2024, pp. 15078–15097, 2024.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023.

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*, pp. 117–127, 2025.

Qiao Sun, Zhicheng Jiang, Hanhong Zhao, and Kaiming He. Is noise conditioning necessary for denoising generative models?, 2025. URL https://arxiv.org/abs/2502.13129.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. *Advances in neural information processing systems*, 36: 1363–1389, 2023.

Runqian Wang and Kaiming He. Diffuse and disperse: Image generation with representation regularization. *arXiv preprint arXiv:2506.09027*, 2025.

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. 15802–15812, 2023.

Yanwu Xu, Yang Zhao, Zhisheng Xiao, and Tingbo Hou. Ufogen: You forward once large scale text-to-image generation via diffusion gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8196–8206, 2024.

Jiawei Yang, Zhengyang Geng, Xuan Ju, Yonglong Tian, and Yue Wang. Representation fréchet loss for visual generation. *arXiv preprint arXiv:2604.28190*, 2026.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15703–15712, 2025.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6613–6623, 2024b.

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. In *ICLR*, 2025.

Weili Zeng and Yichao Yan. Flow matching in the low-noise regime: Pathologies and a contrastive remedy. *arXiv preprint arXiv:2509.20952*, 2025.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 11975–11986, 2023.

Zijian Zhang, Zhou Zhao, and Zhijie Lin. Unsupervised representation learning from pre-trained diffusion probabilistic models. *Advances in neural information processing systems*, 35:22117–22130, 2022.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *ICLR*, 2026.

A Related Work

A.1 Predictive Representation Learning and Generative Features

Self-supervised representation learning includes contrastive objectives such as SimCLR and MoCo (Chen et al., 2020; He et al., 2020), non-contrastive methods such as BYOL, SimSiam, and DINO (Grill et al., 2020; Chen & He, 2021; Caron et al., 2021), and masked prediction (He et al., 2022). I-JEPA (Assran et al., 2023), for example, predicts latent targets rather than pixels. RepFlow adopts this predictive perspective but takes its targets from the generator's internal computation rather than from another clean-view encoder.

Pretrained diffusion and flow models also provide useful features (Zhang et al., 2022; Chen et al., 2025; Zeng & Yan, 2025) for correspondence, segmentation, and classification (Baranchuk et al., 2021; Tang et al., 2023; Mukhopadhyay et al., 2024), although feature quality depends strongly on layer and timestep. DIFT selects denoising states for a downstream task (Tang et al., 2023), whereas CleanDIFT distills noisy generator features into a clean-image encoder (Stracke et al., 2025). RepFlow differs by training its student jointly from multiple generator depths and returning the resulting feature to the generator.

A.2 Representation-guided Generative Modeling

Representation-level objectives can improve generation. REPA aligns generator states with a frozen visual encoder (Yu et al., 2025; Singh et al., 2025), while Representation Autoencoders replace conventional VAE encoders with pretrained representation encoders (Yao et al., 2025; Zheng et al., 2026). RepFlow considers a different source of supervision: the guiding encoder is learned concurrently from the generator's own states. This removes external representation pretraining from the method, but not the computational cost of training the additional student.

Several methods obtain representation guidance from the generator itself. SRA aligns an early, noisy state with a later EMA state at a cleaner timestep (Jiang et al., 2025); Dispersive Loss regularizes internal feature geometry (Wang & He, 2025); Self-Flow creates token-wise noise

asymmetry within a flow model (Chefer et al., 2026); and D-JEPA adds a predictive objective to generative training (Dengsheng et al., 2025). These methods avoid a frozen external encoder, but keep representation learning inside the generative backbone. Their behavior can also depend on the generative parameterization: alignment recipes developed for latent diffusion do not transfer unchanged to pixel-space JiT (Shin et al., 2026).

Architecturally, RepFlow keeps the generator and representation encoder separate. G2R trains the student from multi-depth generator states, and R2G returns the student feature to the generator. By contrast, D-JEPA and Self-Flow place predictive or asymmetric-noise objectives inside a single generative backbone, while CleanDIFT adapts a pretrained generator for feature extraction. We study the same explicit student–generator interface in both latent-space SiT and pixel-space JiT.

A.3 Flow Models and One-Step Generation

One-step generation replaces numerical integration with a single noise-to-data evaluation. Existing approaches use progressive or consistency distillation, distribution matching, and adversarial or score-based objectives (Salimans & Ho, 2022; Song et al., 2023; Song & Dhariwal, 2024; Yin et al., 2024b;a; Yang et al., 2026; Sauer et al., 2024b;a; Ren et al., 2024; Xu et al., 2024). Others learn finite-time maps directly: Shortcut Models condition on step size, flow-map methods learn finite transitions, and MeanFlow predicts average velocity (Frans et al., 2025; Boffi et al., 2024; Geng et al., 2025). Improved MeanFlow, Pixel MeanFlow, and Drifting Models scale these ideas to larger latent- and pixel-space models (Geng et al., 2026; Lu et al., 2026; Deng et al., 2026). RepFlow leaves the joint-training objective multi-step and uses the learned student only as the metric for a short JiT post-training stage.

B Objective and Optimization Analysis

We analyze two aspects of RepFlow: the information rewarded by G2R and the way stop-gradient couples G2R and R2G across optimization steps. The results characterize the training mechanism; they do not imply semantic utility or improved generation, which remain empirical questions.

Let X be a clean image, $\widetilde{X} = \mathcal{M}(X;\zeta)$ its masked view, and T the generative timestep. The multi-level G2R target is

$$Y_T = \mathrm{Concat}\left[Y_T^{(1)}, \ldots, Y_T^{(K)}\right], \qquad Y_T^{(k)} = h_{l_k}^G(X_T, T), \tag{15}$$

where the randomness in X_T includes the sampled reference noise. The student code is $U = U_\phi(\widetilde{X})$ and the predictor has the form $f(U,T)$. All expectations below include data, mask, timestep, and reference noise randomness.

B.1 G2R as Explained Teacher Variation

For a fixed representation U, define the optimal squared prediction risk

$$\mathcal{R}_2(U) := \inf_f \mathbb{E}\left[\|Y_T - f(U,T)\|_2^2\right]. \tag{16}$$

The following decomposition identifies exactly what reducing this risk requires from the representation.

Proposition 1 (Conditional-risk decomposition)**.** *For every square-integrable predictor $f(U,T)$,*

$$\begin{aligned}\mathbb{E}\|Y_T - f(U,T)\|_2^2 &= \mathbb{E}\ \mathrm{Tr}(\mathrm{Var}[Y_T \mid U,T]) \\ &\quad + \mathbb{E}\,\|f(U,T) - \mathbb{E}[Y_T \mid U,T]\|_2^2 .\end{aligned} \tag{17}$$

Consequently,

$$f_U^\star(U,T) = \mathbb{E}[Y_T \mid U,T], \qquad \mathcal{R}_2(U) = \mathbb{E}\ \mathrm{Tr}(\mathrm{Var}[Y_T \mid U,T])\,. \tag{18}$$

If $\mathcal{R}_2(T)$ denotes the risk of a predictor that observes the timestep but not the image representation, then

$$\mathcal{R}_2(T) - \mathcal{R}_2(U) = \mathbb{E}\,\|\mathbb{E}[Y_T \mid U,T] - \mathbb{E}[Y_T \mid T]\|_2^2 \geq 0. \tag{19}$$

Proof. Set $m(U,T) = \mathbb{E}[Y_T \mid U,T]$ and expand $Y_T - f = (Y_T - m) + (m - f)$. The cross term has zero expectation after conditioning on (U,T), which gives the first identity and its minimizer. Applying the same orthogonal decomposition to $m(U,T) - \mathbb{E}[Y_T \mid T]$ yields Equation 19. □

Equation 19 has a direct interpretation: G2R rewards the student for explaining teacher variation beyond timestep alone. Because U is a function of $\widetilde{X}$, the same orthogonal decomposition yields

$$\mathcal{R}_2(U) = \mathcal{R}_2(\widetilde{X}) + \mathbb{E}\left\|\mathbb{E}[Y_T \mid \widetilde{X}, T] - \mathbb{E}[Y_T \mid U, T]\right\|_2^2 \geq \mathcal{R}_2(\widetilde{X}), \tag{20}$$

where $\mathcal{R}_2(\widetilde{X})$ is the Bayes risk when the predictor sees the entire masked view. Equality holds if and only if

$$\mathbb{E}[Y_T \mid U,T] = \mathbb{E}[Y_T \mid \widetilde{X}, T] \quad \text{almost surely.} \tag{21}$$

Hence any student code that attains the masked-view Bayes risk is a mean-sufficient statistic for the cross-timestep prediction profile $\{\mathbb{E}[Y_T \mid \widetilde{X}, T = t]\}_t$, it need not retain aspects of the masked image that never change this profile. Conversely, target components that depend on reference noise or unobserved image content contribute to the irreducible term $\mathcal{R}_2(\widetilde{X})$ and cannot be recovered by any student representation of the masked view.

Experiments use Smooth L1 rather than squared loss. Its population minimizer is a conditional Huber location, so the variance identities apply only to the squared surrogate in Section 3. Under either loss, the predictor acts only on (U,T). The objective alone does not guarantee semantic features, masking and a shared timestep-free student supply the inductive bias. We use linear probing to evaluate whether the resulting representation separates ImageNet classes.

No independence assumption between target layers is needed: both squared and Smooth L1 losses decompose over target coordinates. The predictor may specialize its outputs across layers and timesteps, but every target is decoded from the same code U. This construction encourages a shared explanatory code without asserting that every selected layer carries complementary information. At inference, RepFlow exposes $\bar{U}_\phi(X)$ without a layer or timestep index, the removal of stage selection is an architectural property, not by itself a claim of semantic superiority.

B.2 Gradient Routing and Reciprocal Dynamics

Stop-gradient makes the instantaneous parameter routing exact:

$$\begin{aligned} \nabla_\theta \mathcal{L}_{\text{joint}} &= \nabla_\theta \mathcal{L}_{\text{gen}} + \lambda_2 \nabla_\theta \mathcal{L}_{\text{R2G}}, & \nabla_\phi \mathcal{L}_{\text{joint}} &= \lambda_1 \nabla_\phi \mathcal{L}_{\text{G2R}}, \\ \nabla_\psi \mathcal{L}_{\text{joint}} &= \lambda_1 \nabla_\psi \mathcal{L}_{\text{G2R}}, & \nabla_\omega \mathcal{L}_{\text{joint}} &= \lambda_2 \nabla_\omega \mathcal{L}_{\text{R2G}}. \end{aligned} \tag{22}$$

G2R has no direct generator gradient, and the EMA full-view branch has no direct student gradient. Reciprocity arises across optimization steps. Suppressing optimizer state, the updates have the schematic form

$$(\phi_{k+1}, \psi_{k+1}) \leftarrow (\phi_k, \psi_k) - \eta_s \nabla_{\phi,\psi} \mathcal{L}_{\text{G2R}}(\phi_k, \psi_k; \theta_k), \tag{23}$$

$$(\theta_{k+1}, \omega_{k+1}) \leftarrow (\theta_k, \omega_k) - \eta_g \nabla_{\theta,\omega} \left[\mathcal{L}_{\text{gen}}(\theta_k) + \lambda_2 \mathcal{L}_{\text{R2G}}(\theta_k, \omega_k; \bar{\phi}_k)\right], \tag{24}$$

$$\bar{\phi}_{k+1} \leftarrow m\bar{\phi}_k + (1-m)\phi_{k+1}. \tag{25}$$

Semicolons mark stopped arguments: θ_k determines the current G2R targets, and $\bar{\phi}_k$ determines the current R2G targets, but neither receives a gradient through that occurrence. The loop nevertheless closes across iterations. R2G changes θ_{k+1} and therefore the generator states used by later G2R updates; G2R changes ϕ_{k+1} and therefore the EMA target used by later generator updates. R2G can thus improve the eventual student without sending it a direct gradient, consistent with Table 1.

Any stationary fixed point of this idealized update, with the EMA tracking the online student, satisfies

$$\nabla_{\phi,\psi} \mathcal{L}_{\text{G2R}} = 0, \qquad \nabla_{\theta,\omega} \left(\mathcal{L}_{\text{gen}} + \lambda_2 \mathcal{L}_{\text{R2G}}\right) = 0, \qquad \bar{\phi} = \phi. \tag{26}$$

At such a fixed point, the student summarizes predictable variation in the current generator, while the generator is stationary under both its native objective and alignment to the current student.

The risk decomposition identifies the prediction profile preserved by a Bayes-optimal G2R code, and the gradient equations show how two stopped paths still create feedback over time.

C Implementation and Evaluation Details

Unless noted otherwise, all experiments use ImageNet-1K at 256×256 resolution, a global batch size of 1024, bfloat16 mixed precision, and eight NVIDIA H200 GPUs. Transformer layers are indexed from zero, with an index referring to the output of that block. The main text parameterizes the path from data (t=0) to noise (t=1); the implementation uses $\tau = 1 - t$, and all timesteps below are reported in τ. Following Sun et al. (2025), we omit explicit noise-level (timestep) conditioning from the denoising network in all flow-matching experiments.

C.1 Data and Generative Backbones

Training uses the ImageNet-1K split (1,281,167 images, 1,000 classes) (Russakovsky et al., 2015). We apply the ADM center crop at 256×256, random horizontal flipping, and rescaling to $[-1, 1]$.

SiT. SiT-XL/2 (Ma et al., 2024) operates on 32×32×4 latents from the frozen `sd-vae-ft-ema` VAE with scale 0.18215. Its 28 blocks use width 1152, 16 heads, MLP ratio 4, and patch size 2. We use the official linear interpolant and velocity objective with $\tau \sim \mathrm{Unif}[0, 1]$. The student receives the corresponding RGB image; both branches contain $16 \times 16 = 256$ spatial tokens.

JiT. JiT-XL/16 (Li & He, 2026) acts directly on RGB patches. Its 28 blocks use width 1152, 16 heads, a 128-dimensional bottleneck patch embedding, 2D RoPE, QK normalization, and SwiGLU MLPs. In implementation coordinates,

$$z_\tau = \tau x_0 + (1 - \tau)x_1, \qquad \tau = \sigma(\varepsilon), \quad \varepsilon \sim \mathcal{N}(-0.8, 0.8^2). \tag{27}$$

The network predicts x_0 and minimizes unweighted velocity MSE with $v = (x_0 - z_\tau)/(1 - \tau)$, clamping the denominator at 0.05. Equivalently, the x_0 regression loss is weighted by $(1 - \tau)^{-2}$. The attached RepFlow modules use the same optimizer schedule summarized in Table 4. Conditional training uses label dropout 0.1; unconditional training uses the null label. All reported samples are generated without classifier-free guidance.

Table 4: Optimization and sampling settings. The EMA decay applies to the generator and, in RepFlow, to the full-view student target.

	SiT-XL/2	JiT-XL/16
AdamW β	$(0.9, 0.999)$	$(0.9, 0.95)$
Learning rate	1×10^{-4}	2×10^{-4}
Schedule	constant	cosine, 5 warmup epochs
Minimum learning rate	–	1×10^{-6}
Weight decay	0	0
EMA decay	0.9999	0.9999
Multi-step sampler	dopri5, 250 NFE	Heun, 50 NFE

C.2 RepFlow Instantiation

Student views. The student ViT receives no timestep, class, or adaLN conditioning. ViT-B/16 has 12 blocks, width 768, and 12 heads (85.6M parameters); ViT-L/16 has 24 blocks, width 1024, and 16 heads (303.0M). SiT students use fixed sine–cosine positional embeddings, whereas JiT students use 1D RoPE. G2R masks 75% of the 256 RGB tokens and encodes only the 64 visible tokens, without mask tokens. R2G and evaluation process the complete image. Across all configurations, U_ϕ is strictly an RGB pixel-space encoder: it never consumes VAE latents or any other generator-space inputs, including when the generator is SiT.

Generation-to-representation path. G2R concatenates generator states from

$$\mathcal{D} = \{5, 12, 19, 26\}, \tag{28}$$

forming a 4608-dimensional target at each spatial position. The targets are channel-wise layer-normalized and detached. The predictor restores the 16×16 grid, fills masked positions with sinusoidal timestep embeddings, and applies ten width-384 blocks with 12 heads. Smooth L1 loss ($\beta = 1$) is computed only at masked positions.

In every configuration, the predictor reads the final student block and uses width 384 with 12 heads.

Representation-to-generation path. R2G aligns the generator state after block 7 with the full-view student representation. A two-layer head projects the generator state to the student dimension; for JiT, its hidden width is 1024. Both sides are ℓ_2-normalized per token, and the loss is cosine distance. The stopped full-view target comes from an EMA student with decay 0.9999.

The implemented joint objective is therefore

$$\mathcal{L} = \mathcal{L}_{\text{gen}} + \lambda_1 \mathcal{L}_{\text{G2R}}^{\text{SmoothL1}} + \lambda_2 \mathcal{L}_{\text{R2G}}^{\cos}, \qquad \lambda_1 = 1, \quad \lambda_2 = 0.1. \tag{29}$$

Section 3 uses the same Smooth L1 G2R loss and cosine-distance R2G loss as the implemented objective above. Each joint update evaluates the generator once, the online student on a masked view, the predictor once, and the EMA student on the complete image.

C.3 Baselines and Evaluation Protocols

Generation controls. For unconditional experiments, the generator control retains G2R but sets $\lambda_2 = 0$. Because the G2R target is detached, the generator update is identical to native training, while the run still produces a student for representation evaluation. The class-conditional native baseline omits the student. In both comparisons, R2G is the only additional generator gradient.

REPA (Yu et al., 2025) and SRA (Jiang et al., 2025) match the native backbone, data, batch size, EMA, and sampler. REPA projects block 7 through a three-layer MLP and aligns it with frozen DINOv2-ViT-L/14 patch tokens (Oquab et al., 2023) using a cosine-loss coefficient of 0.5. SRA aligns block 7 with an EMA block-19 state at $\tau' = (\tau + \delta) \wedge 1$, $\delta \sim \text{Unif}[0, 0.2]$, using the same coefficient.

Generation metrics. FID-50K, spatial FID, Inception Score, precision, and recall are computed from 50,000 EMA samples with the ADM evaluator and the fixed ImageNet reference statistics. SiT uses the adaptive ODE sampler and JiT uses Heun as specified in Table 4. One-step JiT evaluation uses a single terminal-noise forward pass (NFE $= 1$). All evaluations use CFG $= 1$.

Frozen linear probing. We freeze the EMA student and average its 256 final-layer tokens. A linear classifier is then trained for 30 epochs with Adam, learning rate 10^{-3}, cosine decay, batch size 1024, and cross-entropy. Training uses the same center crop and horizontal flip as representation learning; validation uses only the center crop.

The raw generator baseline searches over layer and timestep on the unconditional G2R-only checkpoint. The selected features are layer 21 at $\tau = 0.7$ for SiT and layer 21 at $\tau = 0.5$ for JiT. This selection is used only for the raw-feature baseline; RepFlow always evaluates a single full-view student feature.

Ablation controls. The pixel-target control retains the ViT-L student, 75% masking ratio, R2G branch, data, and optimization budget, but replaces the detached multi-level generator target in G2R with masked RGB-patch reconstruction. The target-depth control retains the full reciprocal objective and predicts only the strongest evaluated single generator layer: block 19 for SiT and block 20 for JiT. The multi-level variant uses blocks $\{5, 12, 19, 26\}$.

C.4 One-Step Post-Training

The class-conditional one-step experiment begins from the 400-epoch RepFlow-JiT checkpoint, with the ViT-L student frozen. Real statistics are computed once from layer-normalized, mean-pooled ImageNet features. Generated first and second moments are tracked with EMA decay 0.999 over an effective population of 50,000 samples. Only the generator is updated by the mean and squared-Bures terms from Section 3; evaluation uses CFG $= 1$.

The matched control keeps the initialization, generated samples, optimizer, schedule, and duration fixed, replacing only the frozen RepFlow ViT-L/16 feature with an ImageNet-pretrained MAE ViT-L/16 feature. Real statistics use the same protocol.

Post-training uses AdamW with learning rate 10^{-5}, cosine decay to zero, 6,250 warmup steps, and global batch size 1024. The reported model is trained for 30 additional epochs (37,500 steps). FDr^6

averages normalized Fréchet distances measured in Inception, ConvNeXt, DINOv2, MAE, SigLIP, and CLIP feature spaces. FD-SIM directly optimizes SigLIP, Inception, and MAE distributions, overlapping FID and FDr^6 evaluation. RepFlow-student FD uses a learned feature space distinct from all six evaluation encoders.

C.5 Training Cost and Statistical Scope

RepFlow trades additional training cost for a jointly learned representation. At 600 epochs, SiT takes about 100, 128, and 141 hours for Native, RepFlow-L, and conditional RepFlow-L, respectively; JiT takes 66 and 99–105 hours for Native and RepFlow-L. Throughput excludes sampling, and all runs use seed 0.

Table 5: Training cost and wall-clock matching at global batch size 1024 on eight H200 GPUs. **Top:** measured system cost; SiT memory was not logged. **Bottom:** the RepFlow checkpoint with the same measured wall-clock as a 600-epoch Native run, computed as $600 \times s_{\mathrm{RF}}/s_{\mathrm{Native}}$.

	SiT		JiT		
Method	Params	Steps/s	Params	Steps/s	Memory
Native	673M	2.09	675M	3.15	61.4 GB
REPA	682M	1.82	∼680M	2.59	64.5 GB
SRA	678M	1.78	∼680M	2.45	62.8 GB
RepFlow-B	781M	1.82	785M	2.70	69.3 GB
RepFlow-L	999M	1.63	1002M	2.12	80.1 GB
RepFlow-L (cond.)	999M	1.48	1002M	1.98	83.9 GB

Setting	Native rate	RepFlow rate	Native budget	Equal-time RepFlow
Unconditional SiT	2.09 steps/s	1.63 steps/s	600 epochs	468 epochs
Unconditional JiT	3.15 steps/s	2.12 steps/s	600 epochs	404 epochs
Class-conditional SiT	2.09 steps/s	1.48 steps/s	600 epochs	425 epochs
Class-conditional JiT	3.15 steps/s	1.98 steps/s	600 epochs	377 epochs

At all four matched checkpoints, the RepFlow curves in Figures 4 and 6 remain below the 600-epoch Native endpoint. The generative gain therefore persists when RepFlow is shortened to match Native wall-clock compute, rather than only when the two methods receive equal data exposure.

D Uncurated One-Step Samples

The class grids below show random one-step ImageNet samples (NFE $= 1$). All generated samples are displayed as produced, without manual selection or post hoc filtering.



Figure 8: Random, uncurated one-step ImageNet samples, classes 15–341.



Figure 9: Random, uncurated one-step ImageNet samples, classes 353–551.

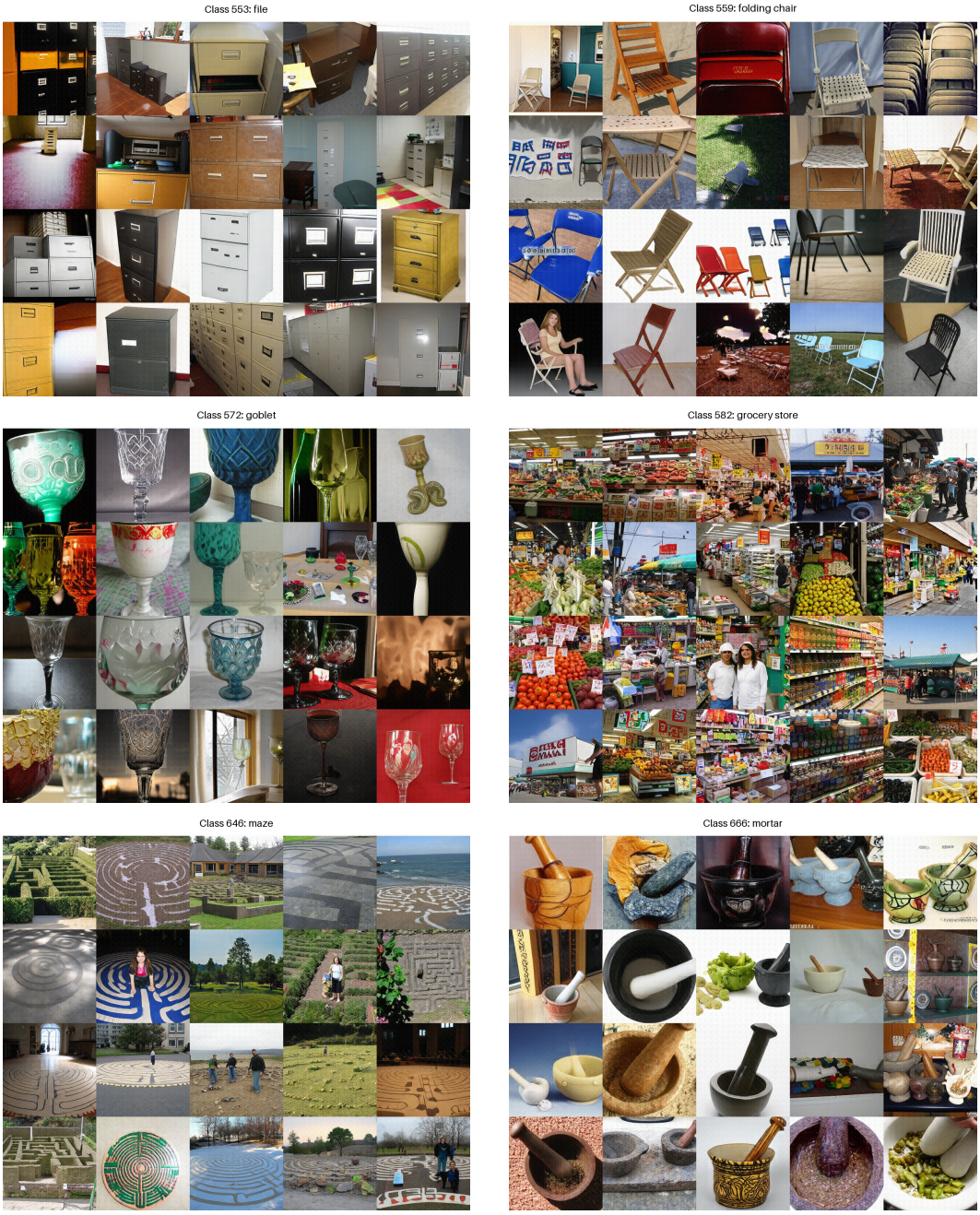


Figure 10: Random, uncurated one-step ImageNet samples, classes 553–666.



Figure 11: Random, uncurated one-step ImageNet samples, classes 676–804.

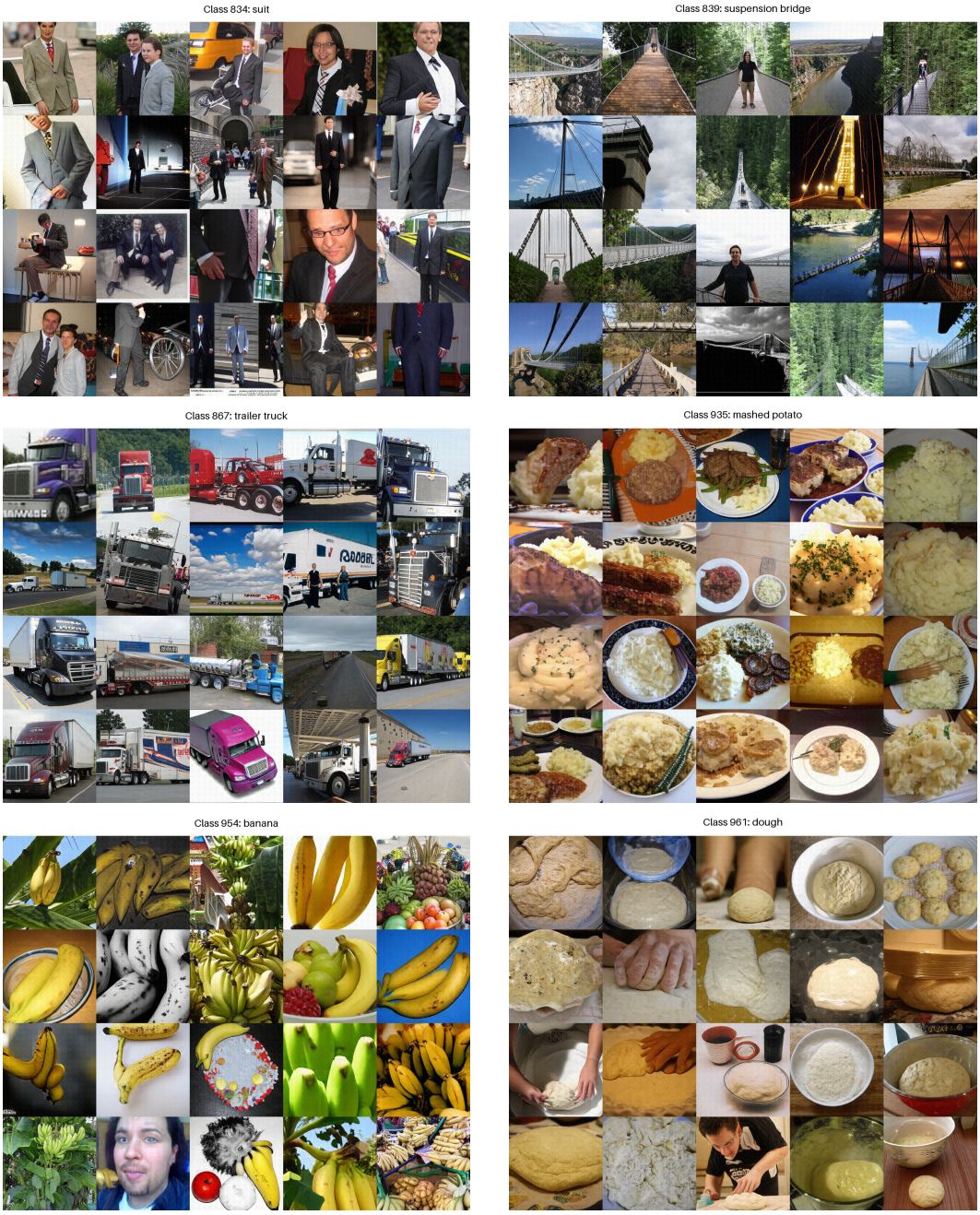


Figure 12: Random, uncurated one-step ImageNet samples, classes 834–961.