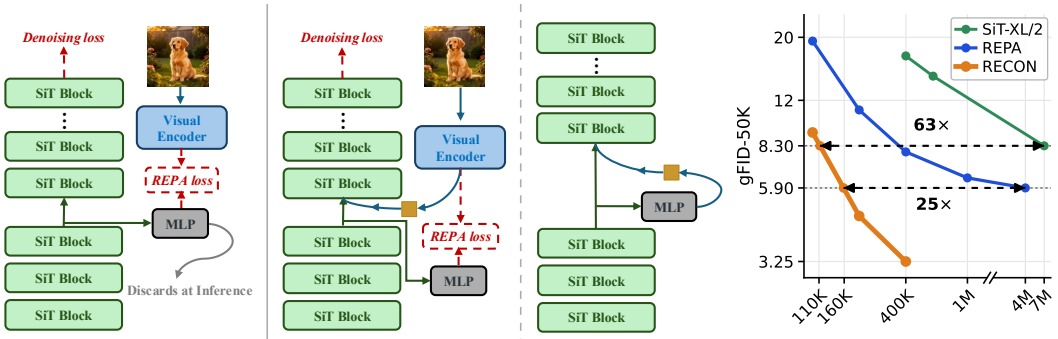
REPRESENTATION CONDITIONING FOR GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Representation-alignment methods effectively accelerate the diffusion transformer training. REPA, for example, uses a pretrained vision encoder to supervise an intermediate layer through an MLP projection head which is discarded after training. It aims to import semantics from the vision encoder to the diffusion backbone. While this objective is achieved at *early layers* by back-propagating the alignment loss, there is no guarantee that *later layers*, supervised by the velocity prediction loss alone, can well access such semantics. We propose a simple solution, representation conditioning (RECON), to improve semantics in later layers. During training, we feed the pretrained vision encoder tokens into later layers; during inference, because we no longer have the clean image, we reuse the MLP head (as a proxy to the vision encoder) and project its output tokens to the later layers. Our method adds no parameters and leads to much faster convergence: RECON is $63\times$ faster than SiT and $25\times$ faster than REPA.



(a) REPA (b) RECON training (c) RECON inference (d) Training steps vs. gFID-50K
Improved generation performance

Figure 1: **RECON conditions the later layers on the representation that REPA discards.** (a) REPA trains a projection head g_ϕ against a frozen vision encoder through the REPA loss, then discards it at inference. RECON inherits most of REPA, but (b) conditions later layers with the vision encoder tokens during training and (c) during inference retains g_ϕ as a proxy to the vision encoder and does the same conditioning. (d) gFID-50K against training step on ImageNet 256×256 without guidance; RECON reaches the final quality of SiT-XL/2 and of REPA much earlier.

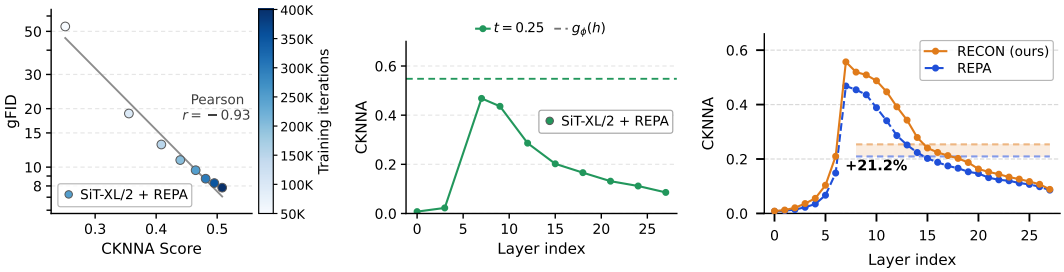
1 INTRODUCTION

An effective strategy for efficient diffusion transformer training is to align the diffusion features with an external vision encoder (Yu et al., 2025). Specifically, REPA (Yu et al., 2025) uses a lightweight projection head g_ϕ[1] to map the intermediate features of diffusion backbone to be similar to those of the frozen vision encoder (Fig. 1a). Several follow-up works build on this idea, *e.g.*, REG (Wu et al., 2025), DDT (Wang et al., 2026) and REPA-E (Leng et al., 2025).

By aligning the diffusion model with an external vision encoder, the assumption is that the diffusion backbone will *effectively* access and use the semantic features, so that generation will eventually improve. In this work, we examine the alignment between the vision encoder features and features of each layer of the diffusion transformer, and the feature from g_ϕ. We find that the feature from

[1]We use g_ϕ interchangeably with "projection head".

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.



(a) Strong correlation of gFID and CKNNA

(b) g_ϕ output has more semantics than any layer

(c) RECON improves CKNNA across layers

Figure 2: **Alignment with the vision encoder, measured by CKNNA against DINOv2-B.** (a) gFID against CKNNA over training checkpoints (b) CKNNA of backbone features at each layer. Dashed line is CKNNA between the projector output $g_\phi(h)$ and the vision encoder, which is stronger than any layer. (c) CKNNA of every layer for REPA and RECON at $t = 0.25$.

g_ϕ has the highest level of alignment with the vision encoder, as shown in Fig.2b. This finding is not surprising, because g_ϕ is physically closest to the vision encoder and is directly optimized to minimize alignment loss.

Surprisingly, despite its strong alignment with the vision encoder, existing methods typically discard the projection head g_ϕ after training, as shown in Fig.1a. We argue that the practice is a waste of model resources including learned weights and semantics: the full potential of representation alignment is not achieved.

We introduce a very simple fix, named REpresentation CONditioning (RECON). Our idea is that instead of removing g_ϕ after training, we keep it and use it as a proxy of the vision encoder during inference, so that strong semantics from g_ϕ can be explicitly used to benefit generation. Specifically, during inference, RECON pools the g_ϕ output features into *semantic tokens*, concatenates them with the patch tokens, and lets the diffusion backbone attend to both types of tokens. Correspondingly, we have a small modification for training recipe: during training, the semantic tokens directly come from the vision encoder and serve as condition for the backbone.

Despite simple, this fix effectively improves the semantics of the backbone features, as shown in Fig.2c. Meanwhile, this method also improves generation quality: we report FID = 3.25 at 80 epochs using SDVAE, which is competitive with the state-of-the-art methods. This is consistent with previous observation that better semantic alignment leads to better generation quality Fig.2a. Furthermore, our method is advantageous in that it adds no parameters[2] and needs fewer sampling steps to reach the same quality Tab. 5. We summarize the main points of this paper below.

- We question the common practice of discarding g_ϕ after training. Retaining it allows the semantics learned through representation alignment to guide generation.
- We introduce RECON that reuses g_ϕ at inference to make the semantics more accessible. Fig. 1c.
- RECON converges $63\times$ faster than SiT and $25\times$ faster than REPA (Fig. 1d, Fig. 3) and is parameter efficient and sampling-step efficient.

2 Related Work

2.1 Representation alignment for diffusion models

Representation alignment. REPA (Yu et al., 2025) is a representative work. DDT (Wang et al., 2026) builds the separation into the architecture, dividing the transformer into a semantic encoder and a velocity decoder and applying the alignment loss to the encoder output. VA-VAE (Yao et al., 2025) moves the alignment out of the generator into the latent space. and REPA-E (Leng et al., 2025) makes joint optimization of VAE and diffusion backbone possible. what is common in all these works is discarding the projection head after training, which, as we argue, is not optimal. Another line of works rely on the supervisory signal from the model itself rather than external vision encoder. SRA (Jiang et al., 2026) distills the representation of a later layer at low noise into an earlier layer at

[2]It generally holds for B, L, and XL models. For certain S models, we add a linear layer. See Sec. 5.1.

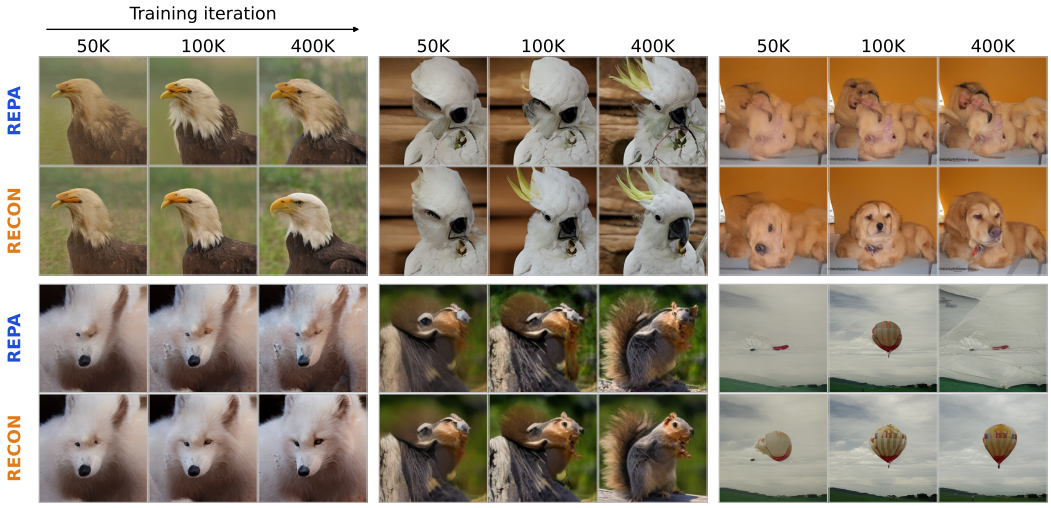


Figure 3: **Representation conditioning improves visual scaling.** RECON produces higher-quality images at different steps compared with vanilla REPA. Results for both methods are sampled using the same seed, noise and class label. We use no classifier-free guidance.

high noise. Self-Flow (Chefer et al., 2026) instead creates information asymmetry across tokens by assigning them different noise levels, so the model has to infer corrupted tokens from cleaner ones. We provide a more detailed related-work discussion in Section A.

Relation to REG. REG (Wu et al., 2025) adds a representation token and diffuses it jointly with the patch tokens, so the token is generated from noise alongside the image while the alignment projector is still discarded. REG performs best when aligning with the encoder *class* token. This is a limitation because not every vision encoder has the class token. In comparison, RECON does not have this requirement and thus can work well with various vision encoders.

3 Preliminaries

Flow matching. Let x_0 be a clean image in RGB space. As in latent diffusion (Rombach et al., 2022), we work with its latent $z_0 = E_{\text{VAE}}(x_0)$, where E_{VAE} is a pretrained VAE encoder. We write z_0 as N patches of dimension d_z, so it lies in $\mathcal{Z} = \mathbb{R}^{N \times d_z}$. Flow matching (Lipman et al., 2023) trains a network to transport noise $z_1 \sim \mathcal{N}(0, I)$ of the same shape to such latents. Following SiT (Ma et al., 2024), the two are connected by the linear interpolant of rectified flow (Liu et al., 2023),

$$z_t = (1-t)\, z_0 + t\, z_1, \qquad t \in [0,1], \tag{1}$$

where $t = 0$ is the clean latent and $t = 1$ is pure noise. The velocity along this path is $\mathrm{d}z_t/\mathrm{d}t = z_1 - z_0$, and a network v_θ learns to predict it from z_t and t alone:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t, x_0, z_1}\Big[\, \| v_\theta(z_t, t) - (z_1 - z_0) \|^2 \,\Big]. \tag{2}$$

SiT also derives from the same v_θ an SDE with the same marginals, which we use in our experiments (Section 6.1). Diffusion models fit the same template with a different interpolant and prediction target, so we use flow-matching notation throughout.

Intermediate representations in diffusion transformers. We parameterize v_θ with a diffusion transformer, as in SiT (Ma et al., 2024), which takes z_t as a sequence of N *patch tokens* of hidden dimension C and processes it with a stack of L transformer blocks. Writing B_j for block j and $h^{(j)} \in \mathcal{H} = \mathbb{R}^{N \times C}$ for its output,

$$h^{(j)} = B_j\big(h^{(j-1)}, t\big), \qquad j = 1, \dots, L, \tag{3}$$

where $h^{(0)}$ holds the patch tokens of z_t and we omit the class label for brevity.

Following REPA (Yu et al., 2025), one can view the diffusion model $v_\theta(z_t, t)$ as a composition of two functions $\mathcal{D}_\theta \circ \mathcal{E}_\theta$: an encoder $\mathcal{E}_\theta : \mathcal{Z} \to \mathcal{H}$ with $\mathcal{E}_\theta(z_t, t) = h^{(\ell)} = h$, made of the first ℓ blocks,

and a decoder $\mathcal{D}_\theta : \mathcal{H} \to \mathcal{Z}$ with $\mathcal{D}_\theta(h, t) = \hat{v}$, made of the remaining $L - \ell$ blocks. We call these blocks the *early* and the *later layers*, respectively. The two are parts of a jointly trained transformer; the split marks where REPA supervises the backbone and where RECON inserts its semantic tokens (Fig. 1b,c).

Representation alignment. Trained with the flow-matching loss (Eq. 2) alone, the diffusion transformer learns semantic features slowly (Yu et al., 2025). REPA (Yu et al., 2025) speeds this up by supervising h with a frozen pretrained vision encoder f_{ext}, such as DINOv2 (Oquab et al., 2024). Let $y_0 = f_{\text{ext}}(x_0) \in \mathbb{R}^{N \times D}$. A small trainable MLP($g_\phi$), maps each token of h into this feature space, shown in Fig. 1(a). REPA then maximizes the similarity between $g_\phi(h)$ and y_0, patch by patch:

$$\mathcal{L}_{\text{align}}(\theta, \phi) = -\,\mathbb{E}_{t,x_0,z_1}\Big[\,\frac{1}{N}\sum_{n=1}^{N}\mathrm{sim}\big(y_0^{[n]},\ g_\phi(h^{[n]})\big)\,\Big], \tag{4}$$

where n indexes patches, and sim is the cosine similarity. This loss is added to Eq. 2 with a weight λ. We call ℓ the *alignment depth* ($\ell = 8$ for SiT-XL/2 in REPA). After training, g_ϕ is discarded and only the diffusion transformer is used for sampling.

4 Motivation

We are motivated by the following three considerations.

First, stronger alignment with external vision encoder *positively correlates* with better generation. For different REPA checkpoints, we plot their CKNNA and generation FID (gFID) in Fig.2(a). CKNNA compares the nearest-neighbor structure of two representation; a higher CKNNA score means higher alignment. The figure demonstrates that alignment and generation improve together.

Second, features of g_ϕ have the *highest similarity* with features of the external vision encoder. This is plotted in Fig.2(b), where CKNNA of g_ϕ is clearly higher than all the transformer layers. This is understandable because g_ϕ is the mapping layer and stays closest to the vision encoder.

Third, existing practices *discard* g_ϕ after training. As a result, the later layers only see h, which is less aligned with the vision encoder than the output of g_ϕ (Fig. 2b).

Based on the above, we argue that it is not reasonable to simply drop the projector g_ϕ, the most semantic-rich module in the diffusion backbone; we keep g_ϕ and give $\mathcal{D}_\theta$ direct access to the best-aligned features, at no extra cost.

5 Proposed Representation Conditioning (RECON)

Overview. The key is to retain $g_\phi(h)$. At inference, we fuse the output of $g_\phi(h)$ with the diffusion patch tokens, which serves as in-context conditioning for the later diffusion layers. Before fusion, the $g_\phi(h)$ output is pooled into a few *semantic tokens*. During training, we directly use the vision encoder features and pool them into semantic tokens, called *teacher forcing*, because the projection layer is being trained and thus not stable. Using perception model features for training and projected features for inference creates a distribution gap. We mitigate this through a *time-dependent scheduling for semantic token injection*. Overall framework is depicted in Fig.1(b) and (c). We detail these components below.

5.1 Training pipeline

Mapping vision encoder tokens to semantic tokens. Let $y_0 = f_{\text{ext}}(x_0) \in \mathbb{R}^{N \times D}$ be the vision encoder features of the clean image. We pool them into $k \ll N$ semantic tokens and map these tokens to the hidden dimension of the model:

$$s = m\big(\mathrm{Pool}(y_0)\big) \in \mathbb{R}^{k \times C}, \tag{5}$$

where C is the working dimension of transformer blocks. Pool is parameter-free and will produce a squared number of semantic tokens, *e.g.*, $k = 1^2, 2^2$. We adopt $k = 1$ by default. m is a function that matches the dimensions. m is the identity mapping when $D = C$; it is zero-padding if $D < C$; otherwise it is a linear mapping. By default we use zero-padding, because the vision encoder dimension D is usually lower than C when the diffusion transformer is large enough, i.e., of

size Base, Large, and Extra Large. Zero-padding adds no parameters. If we use a Small transformer, $D > C$, then we use a linear mapping m, which adds very small number of parameters, *e.g.*, less than 1M parameters. We study the choice of k and m in Table 6.

In-context conditioning. The semantic tokens are concatenated with patch tokens h. They are then inserted in the later blocks $\mathcal{D}_\theta$:

$$\tilde{h} = \left[\, h \,;\, s \,\right] \in \mathbb{R}^{(N+k)\times C}, \qquad \hat{v} = \mathcal{D}_\theta(\tilde{h}, t). \tag{6}$$

In our method, every block of $\mathcal{D}_\theta$ processes patch and semantic tokens together. The patch tokens attend to the semantic tokens at any depth. The semantic tokens are dropped before the final block.

Teacher forcing. During training, we build semantic tokens from the vision encoder features y_0 (Eq. 5), because we have access to the clean image. This is the same teacher forcing method as language models: the model is conditioned on the ground truth tokens rather than on its own predictions. We could also use g_ϕ output to obtain semantic tokens. But we choose teacher forcing because g_ϕ is being trained. Its output starts far from y_0 and keeps changing as training proceeds. Therefore, conditioning on the g_ϕ output would ask the later layers to rely on a signal that is neither accurate nor unstable, whereas y_0 is stable from the first training step. We compare teacher forcing with other training strategies in Table 7.

Time-dependent scheduling for semantic token injection. Teacher forcing at every timestep may introduce a train–inference gap. At inference the semantic tokens come from $g_\phi(h)$, which matches y_0 relatively well at low noise levels and poorly at high noise levels. Training with y_0 at high t (noisy steps) would therefore teach $\mathcal{D}_\theta$ to rely on information that $g_\phi(h)$ cannot provide at generation time. We mitigate this by injecting semantic tokens at timesteps only below a threshold $t_{\max}$,

$$s = \begin{cases} m\big(\mathrm{Pool}(y_0)\big), & t < t_{\max} \\ \varnothing, & t \geq t_{\max} \end{cases}, \tag{7}$$

where $\varnothing$ denotes that no semantic tokens are appended and $\mathcal{D}_\theta$ runs on h alone.

5.2 Inference

In inference, we use the output of g_ϕ in place of y_0 to obtain semantic tokens:

$$s = m\big(\mathrm{Pool}(g_\phi(h))\big). \tag{8}$$

We use g_ϕ because 1) g_ϕ is trained to predict the vision-encoder features from h and thus is a good proxy of the vision encoder and 2) there is no clean image available during inference. In the same way as training, the semantic tokens are then concatenated to the patch tokens and serve as in-context conditioning for later layers. There is no time-dependent scheduling for inference, the benefit of which is shown in Table 8.

6 Experiments

We evaluate RECON introduced in Section 5. Three questions organize the evaluation:

1. Does RECON improve generation quality and accelerate convergence, for both class-conditional and text-to-image generation? (Sections 6.2)
2. How much does RECON improve the semantics of intermediate diffusion features? What is its benefit apart from generation quality? (Section 6.3)
3. Does the improvement hold across model scales, latent spaces, and target encoders? (Section 6.4)

6.1 Setup

Implementation details. We follow the setup of SiT (Ma et al., 2024) and REPA (Yu et al., 2025) unless stated otherwise. We use ImageNet (Deng et al., 2009) at 256×256 resolution and follow ADM (Dhariwal & Nichol, 2021) for data preprocessing. Each image is encoded into a $32 \times 32 \times 4$ latent by the SD-VAE (Rombach et al., 2022). We use the B/2, L/2 and XL/2 models of SiT, which have a patch size of 2, and a batch size of 256. RECON uses alignment depth 8, DINOv2-B

Table 1: **gFID Comparison without CFG** on ImageNet 256×256. SD-VAE latents SD-VAE latents by default. †: VA-VAE latents.

Method	Iter.	gFID↓
No representation alignment		
DiT-XL/2	7M	9.60
SiT-XL/2	7M	8.30
MDT (Gao et al., 2023)	6.5M	6.23
MaskDiT (Zheng et al., 2024)	8M	5.69
Alignment w/o external encoder		
SRA	4M	7.27
LayerSync	4M	6.32
Self-Flow	4M	5.70
Alignment w/ external encoder		
REPA	400K	7.90
REPA	4M	5.90
REG	400K	3.40
REG	1M	2.70
LightningDiT† (Yao et al., 2025)	400K	4.29
REPA-E	400K	4.07
RECON	100K	9.25
RECON	200K	4.64
RECON	400K	3.25
RECON	1M	2.54
RECON	2.4M	2.21

Table 2: **System-level comparison** on ImageNet 256×256 with classifier-free guidance, *: additional CFG scheduling; Baselines are quoted from respective papers.

Model	Epochs	FID↓	IS↑	Pre.↑	Rec.↑
Pixel diffusion					
ADM-G + ADM-U (Dhariwal & Nichol, 2021)	400	3.94	186.7	0.82	0.52
VDM++ (Kingma & Gao, 2023)	560	2.40	225.3	–	–
Simple diffusion (Hoogeboom et al., 2023)	800	2.77	211.8	–	–
CDM (Ho et al., 2022)	2160	4.88	158.7	–	–
Latent diffusion, U-Net					
LDM-4 (Rombach et al., 2022)	200	3.60	247.7	0.87	0.48
Latent diffusion, Transformer + U-Net hybrid					
U-ViT-H/2 (Bao et al., 2023)	240	2.29	263.9	0.82	0.57
DiffiT* (Hatamizadeh et al., 2024)	–	1.73	276.5	0.80	0.62
MDTv2-XL/2*	1080	1.58	314.7	0.79	0.65
Latent diffusion, Transformer					
MaskDiT	1600	2.28	276.6	0.80	0.61
SD-DiT (Zhu et al., 2024)	480	3.23	–	–	–
DiT-XL/2	1400	2.27	278.2	0.83	0.57
SiT-XL/2	1400	2.06	270.3	0.82	0.59
+ REPA	800	1.80	284.0	0.81	0.61
+ REPA*	800	1.42	305.7	0.80	0.65
+ REG	480	1.40	296.9	0.77	0.66
+ RECON	80	2.26	241.22	0.82	0.57
+ RECON	480	1.73	273.67	0.81	0.61



Figure 4: **Qualitative results on ImageNet** 256×256 from SiT-XL/2 + RECON, using classifier-free guidance with $w = 4$

(Oquab et al., 2024) as vision encoder target, $k = 1$, $t_{\max} = 0.5$, and zero-padding for dimension matching (Table 6). For text-to-image generation, we train MMDiT (Esser et al., 2024) on MS-COCO (Lin et al., 2014) and NanoGen (Leng et al., 2026)[3] on BLIP3-o (Chen et al., 2025) (Appendix B.2). More details are in Appendix B.

Datasets and evaluation protocols. We report gFID, inception score, precision and recall over 50K samples, drawn with an SDE sampler at 250 steps for ImageNet. For text-to-image generation, we report FID and CLIP score on MS-COCO (83K images), and FID and IS on BLIP3o-256 (36M images). Samples are drawn with an ODE sampler at 50 steps.

6.2 Impact on generation quality and training speed

We first evaluate class-conditional generation (C2I), and then text-to-image generation (T2I).

C2I: comparison with existing methods. Table 1 summarizes the comparisons using SD-VAE and without CFG. At 400K training steps, our method is stronger than state-of-the-art methods such as REG (3.40) and REPA-E (4.07). Longer training to 1M and 2.4M steps reduces gFID to 2.54 and 2.21, respectively. Fig. 4 shows samples from the final model. Fig. 1(d) shows the training speedup.

C2I: system-level comparisons. Table 2 summarizes the comparisons with CFG. At 80 epochs, RECON is better than MaskDiT (2.28) and DiT (2.27), trained for 1600 and 1400 epochs. At 480 epochs, RECON is better than SiT (2.06) and REPA (1.80), trained for 1400 and 800 epochs. However, our preliminary observation is that CFG helps our method less than REPA and REG. Our

[3]NanoGen's text-to-image setup, which uses a DDT backbone.

Table 3: **Text-to-image on MS-COCO.** MMDiT, ODE, 50 NFE, 150K iter.; w: guidance scale. RECON is superior to REPA.

	w=1		w=2	
Method	FID↓	CLIP↑	FID↓	CLIP↑
+ REPA	9.85	0.2273	4.73	0.2443
+ RECON (ours)	8.26	0.2313	4.55	0.2462

Table 4: **Text-to-image on BLIP3-o** (Chen et al., 2025)**.** NanoGen (Leng et al., 2026); ODE, 50 NFE, 100K iter.

	w=1		w=8	
Method	FID↓	IS↑	FID↓	IS↑
+ REPA	5.41	22.40	5.30	27.61
+ RECON (ours)	3.90	24.31	5.04	27.91

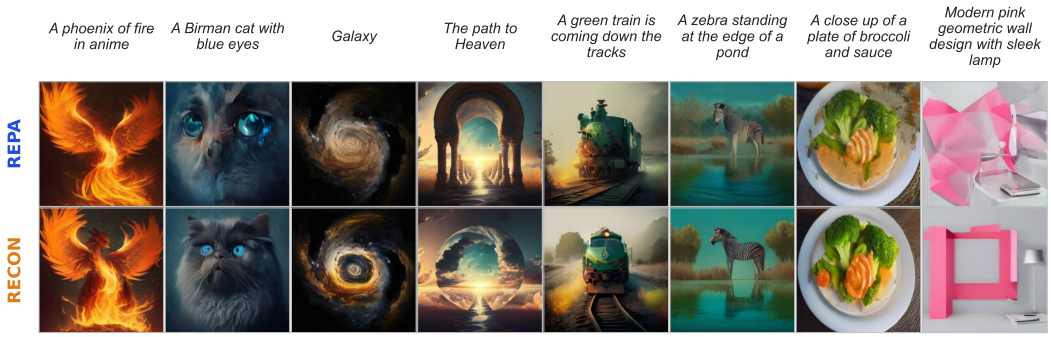


Figure 5: **Qualitative comparison on text-to-image generation (BLIP3-o).** Curated qualitative samples from REPA and RECON trained for 100K iterations. Prompts are shown on top.

hunch is that, with the semantic tokens, the backbone is rich in semantics even without the class label. So the unconditional prediction is already close to the conditional one, and pushing away from it helps less. RAE (Zheng et al., 2026), which trains diffusion on semantically rich DINOv2 features, reports a similar observation regarding CFG. More details in Sec. 7

T2I: comparison with REPA. We compare our method with REPA in Table 3 and Table 4, where we report consistently lower FID, and higher CLIP score and IS. On two different datasets, the gFID of RECON is 8.26 and 3.90, while gFID of REPA is 9.85 and 5.41, respectively. The BLIP-3o results are more useful because the training set is much larger than MS-COCO. Qualitative results are shown in Fig. 5. These results indicate that RECON is effective for T2I training.

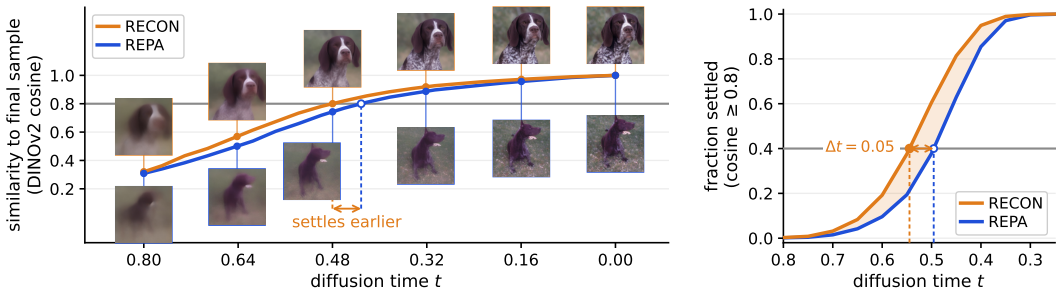
6.3 Effect on representation dynamics

RECON improves semantics in each diffusion layer. Fig. 2c and Fig. 7 use CKNNA and linear probing accuracy to measure semantics in each layer[4], comparing REPA and RECON. We clearly observe that under both measures, RECON brings more semantics to each layer. Specifically, in the later layers that receive semantic tokens, RECON raises mean CKNNA by 21.2% and mean linear probing accuracy by 4.9%.

Semantic structure emerges earlier in sampling. The richer intermediate features also change how a sample develops over generation. We visualize the *semantic trajectory* of a generated sample: at each step we measure the cosine similarity between the running estimate $\hat{x}_0(t)$ and the final image, in the space of DINOv2. Fig. 6 presents how much of the image subject and layout is already decided at time t. Fig. 6(a) shows one sample, while (b) shows the statistics of $10K$ samples. We clearly observe that RECON allows such layout to appear earlier than REPA. A similar behavior is observed under perturbation: given a heavily corrupted image, RECON recovers the identity and overall composition better than REPA (Fig. 9).

RECON works with fewer sampling steps. Since the structure of samples emerges earlier, RECON needs fewer sampling steps for generation. In Table 5, we reduce NFE to 100 and 50 and keep everything else fixed. gFID of REPA degrades from 7.9 to 8.4 and 10.82, respectively. In comparison, RECON keeps gFID of 3.2 at 100 steps, and goes to 3.3 at 50 steps. This means 2.5× to 5× fewer network evaluations at inference, with almost no cost in generation.

[4]For a fair comparison, we exclude the semantic tokens when evaluating both metrics.



(a) Semantic emergence over generation (b) When does structure settle?

Figure 6: **RECON settles the structure of a sample earlier in generation.** (a) measures how close the running estimate $\hat{x}_0(t)$ is, in semantic space, to the image that the same run finally produces. It is measured for a single sample, where RECON is above REPA at every step (both methods start from a same noise). In (b), we generate 10K images. At each diffusion timestep, we measure the percentage of samples whose running estimate $\hat{x}_0(t)$ similarity with the final image is greater than 0.8. RECON is above REPA.

6.4 Generalization and Scalability

Unless stated otherwise every run in this section uses SD-VAE latents, DINOv2-B as the target encoder and alignment depth 8, and is evaluated without classifier-free guidance. Every baseline is vanilla REPA trained under the same protocol. The full metrics are in Appendix E.

Impact of model size. Fig. 8a compares SiT-B/2, SiT-L/2 and SiT-XL/2 at 20 epochs. RECON improves REPA significantly at every model scale. Interestingly, SiT-L/2 with RECON reaches gFID = 11.24, which is better than 18.88 for SiT-XL/2 with REPA.

Effectiveness of RECON for different VAEs. We compare RECON and REPA on three VAEs: SD-VAE (Rombach et al., 2022), VA-VAE (Yao et al., 2025) and In-VAE (Leng et al., 2025). Results are in Fig. 8b. RECON yields consistent improvements over REPA for each tokenizer: gFID improves from 7.84 → 3.25 on SD-VAE, from 7.41 → 4.78 on In-VAE, and from 4.95 → 4.02 on VA-VAE. The improvement on VA-VAE is smaller because it is trained with an additional REPA loss, so a part of the semantics RECON tries to inject into the diffusion backbone is already available. For further discussions see Appendix G.

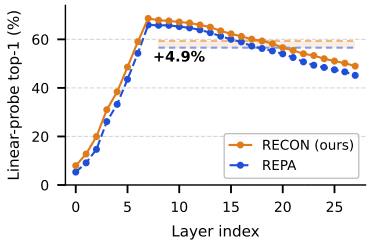
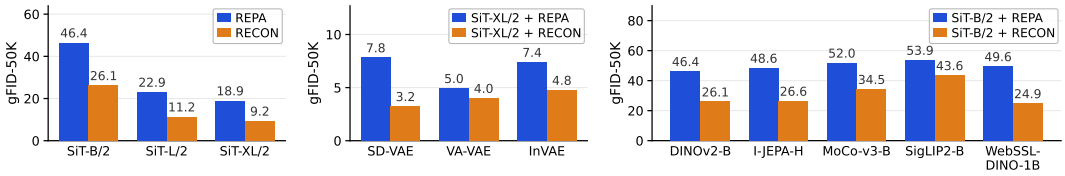


Figure 7: **Linear-probing vs layers.** ImageNet-1K validation.[5]

Table 5: **Sampling steps.** gFID-50K; SiT-XL/2, 400K iter., no CFG.

NFE	50	100	250
REPA	10.82	8.4	7.9
RECON	3.3	3.2	3.2



(a) Variation with in model scales (b) Variation in the tokenizers (c) Variation in the target encoders

Figure 8: **RECON *vs*. REPA across model scale, tokenizer and target encoder.** gFID-50K of REPA and RECON. (a) SiT-B/2, SiT-L/2 and SiT-XL/2 at 20 epochs. (b) Three tokenizers at 80 epochs. (c) Five target vision encoders, SiT-B/2 at 20 epochs.

Effectiveness of RECON for different target vision encoders. We use five encoders: I-JEPA-H (Assran et al., 2023), DINOv2-B, MoCo-v3-B (Chen et al., 2021), SigLIP2-B (Tschannen et al., 2025), and WebSSL-DINO-1B (Fan et al., 2025) and summarize results in Fig.8c. Across the encoders, RECON also gives consistent gFID improvement over REPA.

[5]Linear probing is measured at $t = 0.25$.

Table 6: **Component-wise analysis** on ImageNet 256×256. All models are SiT-XL/2 with DINOv2-B as the target, trained for 400K iterations and measured without classifier-free guidance. Each block varies a single component, shaded, with all others held at the default.

Depth	k	$t_{\max}$	DM mech.	FID↓	IS↑	Pre.↑	Rec.↑
Vanilla SiT-XL/2 (Ma et al., 2024)				17.20	76.52	0.65	0.63
SiT-XL/2 + REPA (Yu et al., 2025)				7.83	125.43	0.70	0.65
6	1	0.5	zero-pad	3.26	205.86	0.81	0.56
8	1	0.5	zero-pad	3.25	204.18	0.81	0.56
10	1	0.5	zero-pad	3.38	202.34	0.81	0.55
8	1	0.5	zero-pad	3.25	204.18	0.81	0.56
8	4	0.5	zero-pad	4.70	214.01	0.84	0.50
8	1	0.3	zero-pad	5.33	146.74	0.74	0.63
8	1	0.5	zero-pad	3.25	204.18	0.81	0.56
8	1	1.0	zero-pad	4.70	236.82	0.84	0.44
8	1	0.5	zero-pad	3.25	204.18	0.81	0.55
8	1	0.5	linear	3.68	198.35	0.80	0.56

Table 7: **Teacher forcing vs. other training strategies.** 'online'/'EMA': tokens from g_ϕ or its moving average. gFID and IS after 100K iterations.

Tokens built from	gFID↓	IS↑
$f_{\text{ext}}(x_0)$	9.2	118.8
$g_\phi(h)$, online	17.1	76.3
$g_\phi(h)$, EMA	16.8	77.5

Table 8: **Injection at inference.** Semantic tokens only for $t < t_{\max}$, as in training, or at every sampling step.

Injected at	gFID↓	IS↑
$t < t_{\max}$ only	6.3	149.9
All timesteps	3.2	202.2

Ablations and variations on design components. Table 6 varies the components RECON introduces, one block at a time: the alignment depth, the number of semantic tokens, the injection threshold $t_{\max}$, and the dimension-matching mechanism. For $t_{\max}$, we choose 0.5. Larger $t_{\max}$ makes diffusion exposed to unseen semantics during inference; smaller $t_{\max}$ makes it harder for diffusion blocks to understand how patch-tokens should interact with semantics. We choose k to be 1. If we increase k, errors in g_ϕ prediction are amplified and can harm the generation quality. For the **dimension-matching mechanism**, we choose zero-padding to increase the dimensions of semantic tokens to be the same as patch tokens. This method is easier and appears to be more effective than linear mapping. **Depth of alignment** analysis suggests that for XL models, Layer 8 is a good choice for alignment. It is also worth mentioning that RECON performance is relatively stable within a reasonable range of this factor.

7 Discussions

Why do later layers need more semantics? Later layers are only supervised by the flow-matching loss and only depend on h. In comparison, the gradient of REPA loss is back-propagated to early layers, so early layers are optimized towards higher semantics. By injecting semantics to late layers, the late layers, through training, can decide whether to use them or not, which is at least not detrimental and turns out to be beneficial.

Rethinking CFG for semantically rich diffusion models. As discussed in Section 6.2, our preliminary observation is that CFG improves RECON less than REPA and REG. Our hunch is that the semantic tokens make the unconditional prediction close to the conditional one, so pushing away from it helps less. RAE (Zheng et al., 2026) reports a similar observation. A future direction can be exploring guidance methods that suit semantically rich backbones.

Limitations. RECON reuses a projection head that a REPA-style alignment loss has already trained, so it cannot be applied to a model trained without such a loss. What it removes is the waste in discarding that head, not the need for alignment in the first place. More in Appendix F

8 Conclusion

This paper challenges the practice in representation alignment methods that the projection head g_ϕ should be discarded after training. This is based on our observation that the output of g_ϕ is very semantical: it aligns with the vision encoder more strongly than any of the diffusion transformer layers. Discarding it means we forfeit useful and also free cues. We propose representation conditioning (RECON), which supplies output of g_ϕ to the later layers. With no added parameters and no change in training loss, RECON simply lets later layers decide how to use the conditioning semantics. We report good improvement over REPA in various settings. We hope that RECON was effective in conveying and bringing up the following message: *what matters in representation alignment is not only acquiring a representation, but also keeping it accessible in generation.*

AI USE STATEMENT

In this work, we used Claude Code (Anthropic, 2026) to assist with implementing our methods, preparing experiment configurations, and running experiments. Additionally, we used Claude Code to draft the paper outline, check the grammar of the manuscript, and improve the clarity of the writing. We also used it to generate most of the figures and tables from our experiment logs and checkpoints, following the colour palette we provided. We have reviewed all AI-assisted work, including the code, experiment configurations, figures, tables, and text. We take responsibility for the final content of this work and the associated codebase, including all materials produced with AI assistance.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide all implementation details and hyperparameters in AppendixB

REFERENCES

Anthropic. Claude. https://claude.ai, 2026.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, 2023.

Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A ViT backbone for diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

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. In *International Conference on Machine Learning (ICML)*, 2026.

Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, Le Xue, Caiming Xiong, and Ran Xu. BLIP3-o: A family of fully open unified multimodal models – architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

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

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

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning (ICML)*, 2024.

David Fan, Shengbang Tong, Jiachen Zhu, Koustuv Sinha, Zhuang Liu, Xinlei Chen, Michael Rabbat, Nicolas Ballas, Yann LeCun, Amir Bar, and Saining Xie. Scaling language-free visual representation learning. In *International Conference on Computer Vision (ICCV)*, 2025.

Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Masked diffusion transformer is a strong image synthesizer. In *International Conference on Computer Vision (ICCV)*, 2023.

Yasaman Haghighi, Bastien van Delft, Mariam Hassan, and Alexandre Alahi. LayerSync: Self-aligning intermediate layers. In *International Conference on Learning Representations (ICLR)*, 2026.

Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. DiffiT: Diffusion vision transformers for image generation. In *European Conference on Computer Vision (ECCV)*, 2024.

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

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

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, 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 (JMLR)*, 2022.

Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning (ICML)*, 2023.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. Position: The platonic representation hypothesis. In *International Conference on Machine Learning (ICML)*, 2024.

Dengyang Jiang, Mengmeng Wang, Liuzhuozheng Li, Lei Zhang, Haoyu Wang, Wei Wei, Guang Dai, Yanning Zhang, and Jingdong Wang. Representation alignment for diffusion transformers without external components. In *International Conference on Learning Representations (ICLR)*, 2026.

Diederik P. Kingma and Ruiqi Gao. Understanding diffusion objectives as the ELBO with simple data augmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning (ICML)*, 2019.

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

Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. REPA-E: Unlocking VAE for end-to-end tuning of latent diffusion transformers. In *International Conference on Computer Vision (ICCV)*, 2025.

Xingjian Leng, Jaskirat Singh, Zhanhao Liang, Ethan Smith, Martin Bell, Aninda Saha, Yuhui Yuan, and Liang Zheng. DiffusionBench: On holistic evaluation of diffusion transformers. *arXiv preprint arXiv:2606.24888*, 2026.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

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

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.

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 (ECCV)*, 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. *Transactions on Machine Learning Research (TMLR)*, 2024.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021.

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)*, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Ge Wu, Shen Zhang, Ruijing Shi, Shanghua Gao, Zhenyuan Chen, Lei Wang, Zhaowei Chen, Hongcheng Gao, Yao Tang, Jian Yang, Ming-Ming Cheng, and Xiang Li. Representation entanglement for generation: Training diffusion transformers is much easier than you think. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

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 *International Conference on Learning Representations (ICLR)*, 2025.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *International Conference on Learning Representations (ICLR)*, 2026.

Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar. Fast training of diffusion models with masked transformers. *Transactions on Machine Learning Research (TMLR)*, 2024.

Rui Zhu, Yingwei Pan, Yehao Li, Ting Yao, Zhenglong Sun, Tao Mei, and Chang Wen Chen. SD-DiT: Unleashing the power of self-supervised discrimination in diffusion transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

A Extended related work

Section 2 covers representation alignment. Here we extend it with the diffusion backbones we compare against, other ways of bringing pretrained representations into diffusion models, and train–inference gaps.

Diffusion transformers. DiT (Peebles & Xie, 2023) replaced the U-Net of latent diffusion (Rombach et al., 2022) with a transformer and showed that generation quality improves with model compute. SiT (Ma et al., 2024) keeps the DiT architecture and trains it with the interpolant framework, which is the setting we follow. Other works change the architecture or add objectives during training: U-ViT (Bao et al., 2023) adds long skip connections between shallow and deep blocks, DiffiT (Hatamizadeh et al., 2024) makes self-attention time-dependent, and MDT (Gao et al., 2023), MaskDiT (Zheng et al., 2024) and SD-DiT (Zhu et al., 2024) add masking or self-supervised objectives. For text-to-image generation, MMDiT (Esser et al., 2024) keeps text and image tokens in separate streams that interact through joint attention.

Pretrained representations in the latent space. Instead of aligning the generator, some works build the semantics into the latent space itself. VA-VAE and REPA-E (Section 2) align or fine-tune the VAE. RAE (Zheng et al., 2026) goes further and replaces the VAE encoder with a frozen pretrained representation encoder such as DINOv2, training only a decoder and running diffusion in that representation space. These approaches are complementary to RECON: in our tokenizer experiments (Fig. 8b), RECON still improves generation on VA-VAE, although by a smaller margin than on SD-VAE.

Train–inference gaps and forcing strategies. Conditioning a model on information during training that is not available at inference is a well-known problem. In autoregressive video diffusion, teacher forcing condition on context that the model must later produce itself, and Self Forcing (Huang et al., 2025) closes the resulting gap by rolling out the model during training. RECON has the same kind of gap: it uses teacher forcing, conditioning on vision encoder features during training and on the output of g_ϕ at inference. We reduce this gap with time-dependent scheduling for semantic token injection (Section 5).

B Hyperparameter and more implementation details

Table 9: **Hyperparameters, class-conditional.** ImageNet 256×256.

	SiT-B/2	SiT-L/2	SiT-XL/2
Architecture			
Input dim.		$32 \times 32 \times 4$	
Num. layers	12	24	28
Hidden dim.	768	1024	1152
Num. heads	12	16	16
Alignment (REPA)			
λ		1.5	
Alignment depth		8	
Similarity		cos. sim.	
Vision encoder		DINOv2-B	
RECON			
Sem. tokens k		1	
$t_{\max}$ (training)		0.5	
Dim. matching	identity	zero-pad	zero-pad
Optimization			
Iterations	100K	100K	400K–2.4M
Batch size		256	
Optimizer		AdamW	
Learning rate		10^{-4}	
(β_1, β_2)		(0.9, 0.999)	
Interpolant and sampling			
Objective		v-prediction	
Sampler		Euler–Maruyama	
Sampling steps		250	

Table 10: **Hyperparameters, text-to-image.** Settings not listed are as in Table 9.

	MS-COCO	BLIP3-o
Architecture		
Backbone	MMDiT	NanoGen
Num. layers[†]	24	28 + 2
Hidden dim.[†]	768	1152 / 2048
Num. heads	24	16
Text conditioning		
Text encoder	CLIP ViT-L/14	Qwen3-0.6B
Max. text tokens	77	256
Timestep tokens	0 (adaLN)	4
CFG dropout	0.1	0.1
Alignment (REPA)		
λ	1.5	1.5
Optimization		
Iterations	150K	100K
Batch size	256	1024
Optimizer	AdamW	AdamW
Learning rate	10^{-4}	2×10^{-4}
(β_1, β_2)	(0.9, 0.999)	(0.9, 0.95)
EMA decay	0.9999	0.9995
Interpolant and sampling		
Time sampling	uniform	logit-normal
Sampler	ODE (Heun)	ODE (Euler)
Sampling steps	50	50
Guidance scale w	1, 2	1, 8
Data		
Training images	$\sim$83K	$\sim$36M

[†]NanoGen: encoder + decoder. (A text-to-image variant of DDT)

B.1 Class-conditional generation

Further implementation details. We build on the official REPA implementation, which follows SiT (Ma et al., 2024). We use AdamW with a constant learning rate of 10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.999)$ and no weight decay, and train with mixed precision (fp16) and gradient clipping. We pre-compute the SD-VAE latents and apply no data augmentation. As in REPA, the projection head g_ϕ is a three-layer MLP with SiLU activations. Table 9 lists the full setup.

RECON settings. Pool in Eq. 5 is average pooling of the vision encoder tokens to k tokens. The semantic tokens are concatenated with the patch tokens after the alignment depth and dropped before the final block (Section 5). Let D be the feature dimension of the vision encoder and C the hidden dimension of the diffusion transformer. The dimension-matching function m is the identity when $D = C$, zero-padding when $D < C$, and a linear layer when $D > C$; for DINOv2-B ($D = 768$), this gives the entries in Table 9. During training, the tokens come from the vision encoder and are injected only for $t < t_{\max}$. During inference, they come from $g_\phi(h)$ and are injected at every step (Table 8).

Patch size per tokenizer. For the tokenizer study (Table 12), we keep the XL backbone fixed and only change its patch size, so that every run processes the same number of tokens. SD-VAE (Rombach et al., 2022) downsamples the 256×256 image by 8 and gives a $32 \times 32 \times 4$ latent. We therefore use a patch size of 2 (SiT-XL/2). VA-VAE (Yao et al., 2025) and InVAE (Leng et al., 2025) downsample by 16 and give a $16 \times 16 \times 32$ latent. Following REPA-E, we use a patch size of 1 for them (SiT-XL/1). In both cases, the transformer processes $16 \times 16 = 256$ patch tokens, so the sequence length and compute per step are the same across the three tokenizers.

Pretrained encoders. DINOv2-B (Oquab et al., 2024) is the default target. For the encoder study (Fig. 8c), we also use I-JEPA-H (Assran et al., 2023), MoCo-v3-B (Chen et al., 2021), SigLIP2-B (Tschannen et al., 2025) and WebSSL-DINO-1B (Fan et al., 2025), all with their official checkpoints.

Epochs and iterations. ImageNet-1K has about 1.28M training images, so at a batch size of 256 one epoch is about 5K iterations. Hence, 100K, 200K, 400K, 1M and 2.4M iterations correspond to 20, 40, 80, 200 and 480 epochs, respectively. All checkpoints are evaluated with the protocol of Section 6.1.

Computing resources. All models are trained on 4 NVIDIA H100 GPUs. In our default setting (SiT-XL/2, batch size 256), training runs at about 4.10 iterations per second. A 400K-iteration run (80 epochs) therefore takes about 27 hours (about 110 GPU-hours), and the longest 2.4M-iteration run (480 epochs) about 6.8 days (about 650 GPU-hours). RECON adds only $k = 1$ semantic token to the 256 patch tokens of the later layers, so its training cost is almost the same as that of REPA.

B.2 Text-to-image generation

Table 10 lists the settings.

MS-COCO. We follow the text-to-image setup of U-ViT (Bao et al., 2023). We train MMDiT (Esser et al., 2024) from scratch on the training split of MS-COCO (Lin et al., 2014) and evaluate on the validation split.

BLIP3-o. We use the NanoGen configuration of DiffusionBench (Leng et al., 2026) and train on BLIP3-o (Chen et al., 2025), about 36M image–text pairs at 256×256.

C Representation analysis

C.1 CKNNA

Centered Kernel Nearest-Neighbor Alignment (CKNNA) (Huh et al., 2024) measures whether two representations give the same *local geometry* to the same set of images. It is the local counterpart of Centered Kernel Alignment (CKA) (Kornblith et al., 2019).

Let $\phi_i \in \mathbb{R}^{d_\phi}$ and $\psi_i \in \mathbb{R}^{d_\psi}$ be the two representations of image i, for $i = 1, \dots, n$. With the linear kernel, their Gram matrices and row-centered entries are

$$K_{ij} = \langle \phi_i, \phi_j \rangle, \qquad \bar{K}_{ij} = K_{ij} - \frac{1}{n} \sum_{\ell=1}^{n} K_{i\ell}, \tag{9}$$

$$L_{ij} = \langle \psi_i, \psi_j \rangle, \qquad \bar{L}_{ij} = L_{ij} - \frac{1}{n} \sum_{\ell=1}^{n} L_{i\ell}. \tag{10}$$

CKA correlates these centered similarities over all image pairs. CKNNA keeps only pairs that are neighbours in *both* spaces. If $\mathcal{N}_K^k(i)$ and $\mathcal{N}_L^k(i)$ are the k nearest neighbours of image i under K and L, define

$$\alpha_{ij}^{K,L} = \mathbf{1}\big[i \neq j,\ j \in \mathcal{N}_K^k(i) \cap \mathcal{N}_L^k(i)\big], \qquad A_k(K,L) = \sum_{i,j} \alpha_{ij}^{K,L} \bar{K}_{ij} \bar{L}_{ij}. \tag{11}$$

The normalized score is

$$\mathrm{CKNNA}_k(K,L) = \frac{A_k(K,L)}{\sqrt{A_k(K,K)\, A_k(L,L)}}. \tag{12}$$

Thus a high score means that images grouped together by one representation are also grouped together by the other, with similar centered affinities. Unlike a direct feature distance, CKNNA remains meaningful when $d_\phi \neq d_\psi$; its normalization also removes a common rescaling of either representation. With the unbiased convention used here, $k = n - 1$ admits every non-diagonal pair and recovers CKA, whereas a small k focuses the comparison on local semantic structure.

For every CKNNA result in this paper, we randomly sample $n = 10{,}000$ ImageNet validation images and use $k = 10$, following REPA. We spatially average the dense patch features to obtain one vector per image, discard RECON's semantic tokens before pooling, and ℓ_2-normalize the vectors before forming the linear kernels. Scores use the unbiased estimator. The reference is the clean-image DINOv2-B representation; the layer-wise results use the 400K-iteration EMA checkpoint at $t = 0.25$.

C.2 Linear probing

We follow the linear probing protocol of MAE (He et al., 2022). For each layer, we take the frozen features of the 400K-iteration EMA checkpoint at $t = 0.25$. We use the null class label, so that no label information leaks into the features. As for CKNNA, we average the patch tokens to get one vector per image, and discard RECON's semantic tokens before pooling. The classifier is a BatchNorm layer without affine parameters, followed by a linear layer. We train it on the ImageNet-1K training set and report top-1 accuracy on the validation set. Unlike the original recipe, which trains for 90 epochs, we train the classifier for 20 epochs to reduce the cost of probing every layer. REPA and RECON use the same schedule, so the comparison between them is fair.

D Evaluation details

For ImageNet, we follow the evaluation setup of ADM (Dhariwal & Nichol, 2021) and use its reference batches. For text-to-image generation, we compute FID against MS-COCO validation split, and on 50,000 samples for BLIP3-o. The metrics are:

- **FID** (Heusel et al., 2017) measures the distance between the feature distributions of real and generated images. It uses the Inception-v3 network (Szegedy et al., 2016) and models both distributions as Gaussians.
- **IS** (Salimans et al., 2016) also uses Inception-v3. It is high when the class prediction for each generated image is confident and the predictions over all images are diverse.
- **Precision and recall** (Kynkäänniemi et al., 2019) measure the fraction of generated images that fall on the real data manifold, and the fraction of the real data manifold that the generated images cover, respectively.
- **CLIP score** (Hessel et al., 2021) measures how well a generated image matches its text prompt, as the similarity of the two in the embedding space of CLIP (Radford et al., 2021).
- **CKNNA and linear probing** measure the semantics of the diffusion features (Appendix C).

E Additional results

E.1 Full metrics

Fig. 8 reports only FID. Here we give FID, IS, precision and recall for the same runs: model scale (Table 11), tokenizer (Table 12) and target encoder (Table 13).

Table 11: **Model scale.** 20 epochs on ImageNet 256×256 with SD-VAE latents, no CFG.

Model	Method	FID↓	IS↑	Pre.↑	Rec.↑
SiT-B/2	REPA	46.35	30.53	0.477	**0.608**
	RECON	**26.08**	**54.63**	**0.620**	0.577
SiT-L/2	REPA	22.95	59.17	0.618	**0.610**
	RECON	**11.24**	**104.44**	**0.742**	0.550
SiT-XL/2	REPA	18.88	69.34	0.647	**0.608**
	RECON	**9.25**	**118.80**	**0.760**	0.543

Table 12: **Tokenizer.** SiT-XL, 80 epochs on ImageNet 256×256, no CFG.

Latent	Method	FID↓	IS↑	Pre.↑	Rec.↑
SD-VAE	REPA	7.84	125.43	0.700	**0.654**
	RECON	**3.25**	**204.18**	**0.817**	0.565
VA-VAE	REPA	4.95	146.43	0.761	**0.611**
	RECON	**4.02**	**163.04**	**0.774**	0.604
InVAE	REPA	7.41	125.25	0.712	**0.655**
	RECON	**4.78**	**151.10**	**0.755**	0.627

Table 13: **Target encoder.** SiT-B/2, 20 epochs on ImageNet 256×256 with SD-VAE latents, no CFG.

Encoder	Method	FID↓	IS↑	Pre.↑	Rec.↑
DINOv2-B	REPA	46.35	30.53	0.477	**0.608**
	RECON	**26.08**	**54.63**	**0.620**	0.577
I-JEPA-H	REPA	48.56	28.41	0.465	**0.596**
	RECON	**26.56**	**54.36**	**0.592**	0.574
MoCo-v3-B	REPA	51.98	26.60	0.449	**0.597**
	RECON	**34.45**	**41.88**	**0.584**	0.547
SigLIP2-B	REPA	53.85	25.12	0.440	**0.587**
	RECON	**43.65**	**31.29**	**0.508**	0.550
WebSSL-DINO-1B	REPA	49.61	28.52	0.459	**0.596**
	RECON	**24.92**	**55.74**	**0.656**	0.527

E.2 Recovering heavily corrupted images

In Section 6.3, we show that RECON decides the structure of a sample earlier in generation. Here we test a related question: can the model recover an image from heavy corruption? We take three clean images and corrupt each with heavy noise, moving it along the interpolant to $t = 0.8$, so the input is 80% noise and 20% image. Then we let REPA and RECON (SiT-XL/2) denoise it. Both start from the same noisy latent and share the class label, the sampling noise and the sampler (Euler–Maruyama, 200 steps, without CFG), so any difference comes from the model. Fig. 9 shows the clean-image estimate $\hat{x}_0(t)$ along the way. RECON commits to the subject within the first few steps and then refines it, while the estimates of REPA stay blurry for much longer. As a result, RECON recovers the identity and layout of the original image better. Since most of the input is noise, neither model reconstructs it exactly; the difference is in how fast and how well a plausible image is formed.

F Limitations

Fidelity–diversity balance. RECON strongly improves sample fidelity. At 80 epochs, it increases the precision of SiT-XL/2 ($0.70 \rightarrow 0.81$) and IS ($125.4 \rightarrow 204.2$), while recall decreases ($0.65 \rightarrow 0.56$) (Table 6). Our hunch is that strong semantic conditioning makes the model commit to clear class semantics early, which leaves less room for variation. RAE (Zheng et al., 2026), whose latent space is DINOv2 features, shows the same behavior (precision 0.82 and recall 0.59 at 80 epochs).

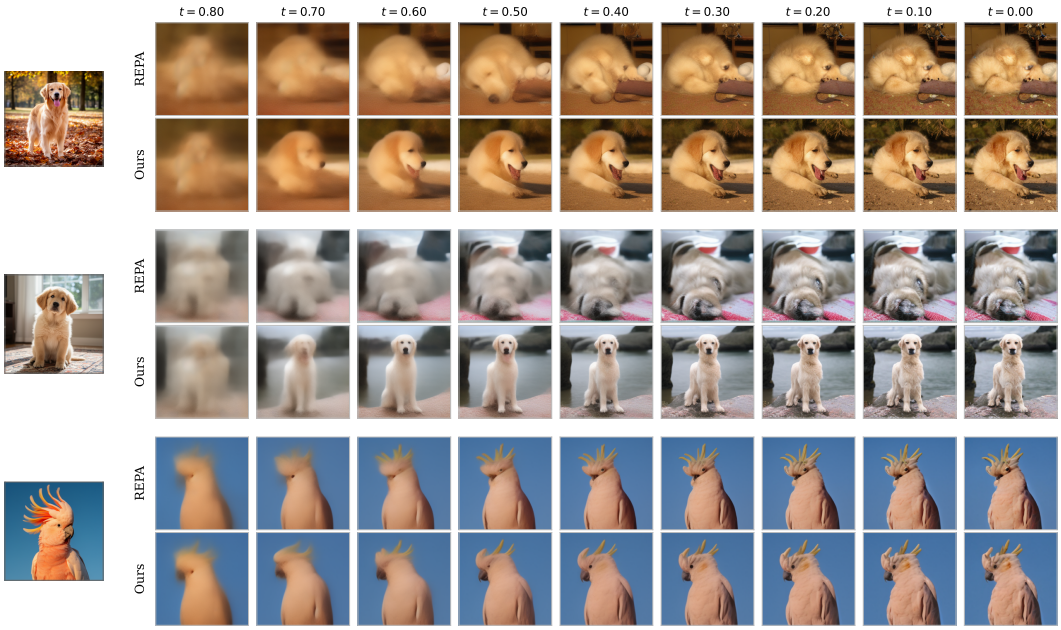


Figure 9: **Denoising heavily corrupted images.** We take three clean images (left) and corrupt each with heavy noise (80% noise and 20% image). Then let REPA and RECON denoise them; each frame shows the clean-image estimate $\hat{x}_0(t)$. RECON recovers the identity and layout of the original image better then REPA, which shows its strength in retaining semantics under heavy noise.

G Discussion and future work

Combining with self-contained alignment. Nothing in RECON depends on where the target representation comes from, only on there being a trained map into a better-organized space. Pairing it with a method that derives its target internally (Jiang et al., 2026; Haghighi et al., 2026; Chefer et al., 2026) would remove the external encoder entirely while keeping the mechanism intact. We regard this as the most promising direction.

Is the gain bottlenecked by the tokenizer? RECON improves generation on all three tokenizers, but by different amounts (Table 12). FID goes from $7.84 \rightarrow 3.25$ on SD-VAE and $7.41 \rightarrow 4.78$ on InVAE, but only from $4.95 \rightarrow 4.02$ on VA-VAE. VA-VAE is the only tokenizer trained with an alignment loss, so its latent already contains part of the semantics that RECON adds. In other words, RECON helps most when the latent has no semantic structure of its own. We do not yet know why the remaining gain on VA-VAE is small. One possibility is that the tokenizer itself limits how much can be gained. Another is that our semantic token, a single pooled summary of the image, is not the right signal for a latent that is already semantically organized, and a signal with information that the latent does not contain could still help. Telling these two apart would show whether RECON is close to its limit on such tokenizers, or whether a different way of injecting semantics could give further gains.

Why no time-dependent scheduling during inference? During inference, for $t < t_{\max}$, the decoder receives the same kind of input as in training. For $t \geq t_{\max}$, we still inject semantic tokens, although $\mathcal{D}_\theta$ does not see the semantic tokens in training, and $g_\phi(h)$ is less accurate at these noise levels. In practice this improves gFID (Table 8), which suggests that $\mathcal{D}_\theta$ applies what it learned at low noise levels across the whole trajectory.