

Pretrained Reversible Generation as Unsupervised Visual Representation Learning

Rongkun Xue^{♣*}, Jinouwen Zhang^{♡*}, Yazhe Niu^{♣♣}, Dazhong Shen[◇], Bingqi Ma[♠], Yu Liu[♠], Jing Yang^{♠✉}

[♠]Xi’an Jiaotong University [♡]Shanghai AI Laboratory [♠]SenseTime

[♣]The Chinese University of Hong Kong [◇]Nanjing University of Aeronautics and Astronautics

xuerongkun@stu.xjtu.edu.cn, zhangjinouwen@pjlab.org.cn, niuyazhe314@outlook.com

Abstract

Recent generative models based on score matching and flow matching have significantly advanced generation tasks, but their potential in discriminative tasks remains underexplored. Previous approaches, such as generative classifiers, have not fully leveraged the capabilities of these models for discriminative tasks due to their intricate designs. We propose Pretrained Reversible Generation (PRG), which extracts unsupervised representations by reversing the generative process of a pretrained continuous generation model. PRG effectively reuses unsupervised generative models, leveraging their high capacity to serve as robust and generalizable feature extractors for downstream tasks. This framework enables the flexible selection of feature hierarchies tailored to specific downstream tasks. Our method consistently outperforms prior approaches across multiple benchmarks, achieving state-of-the-art performance among generative model based methods, including 78% top-1 accuracy on ImageNet at a resolution of 64×64 . Extensive ablation studies, including out-of-distribution evaluations, further validate the effectiveness of our approach. PRG is available at <https://opendilab.github.io/PRG/>.

1. Introduction

“What I cannot create, I do not understand.”

— Richard Feynman

Generative models formulated as continuous-time stochastic differential equations, including diffusion and flow models [2, 37, 39, 51, 55], have shown remarkable proficiency in multi-modal content generation tasks [27, 45, 53, 67], as well as in scientific modeling [1, 75]. By effectively learning high-dimensional distributions, these models can generate new, high-quality samples that resemble the original data.

While generative tasks require models to reconstruct the

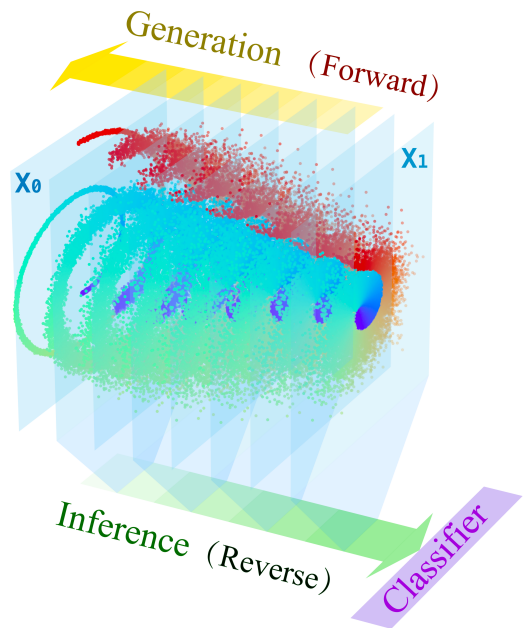


Figure 1. Overview of the PRG as Unsupervised Visual Representation pipeline. Swiss-roll data is generated using $(x, y) = (t \cos(t), t \sin(t))$ with $t \in [0, 3\pi]$, where the color gradient from blue to red corresponds to t increasing from 0 to 3π . The yellow trajectory represents the generative (forward) process, with color intensity indicating direction. Green arrows illustrate the inference (reverse) generation process. Using the pretrained model, each step along the green path can be fine-tuned for downstream tasks.

entire data distribution, discriminative tasks (e.g., image classification) rely on learning representations where some parts capture the essential underlying data structure to produce discrete or continuous predictions. Previous works [17, 23] have explored various approaches to enhance representation learning, with growing focus on unsupervised visual representation learning [10, 12]. These methods construct generic proxy tasks to efficiently extract and utilize data for pre-training, yielding robust representations for diverse

*Equal contribution.

✉Corresponding author.

downstream tasks [5, 10, 13, 16, 52, 61, 68].

Due to the extraordinary performance of diffusion models in generative tasks, recent studies have explored their potential for discriminative tasks [14, 35]. These studies often rely on extracting or combining representations from the internal layers of pretrained diffusion models, involving intricate and poorly generalizable designs, yet usually lacking a clear explanation. For example, [68] selects a specific layer in a UNet-based diffusion model as the feature representation. Moreover, the performance of these methods exhibits significant room for improvement compared to current state-of-the-art techniques.

Inspired by the reversible property [11, 58] of continuous-time stochastic flow models, we introduce Pretrained Reversible Generation as Unsupervised Visual Representation Learning (PRG). Our approach applies generative models to discriminative tasks by reversing the generative process into a meaningful inference process. Specifically, we pretrain a reversible generative model via flow matching to maximize the lower bound of mutual information between the original image and its optimal representation, providing a strong initialization. We then reverse the generative process and leverage it for downstream training to improve task performance. This model-agnostic method does not require access to any internal features of a pretrained generative model, eliminating the need for fixed network modules to capture data representations. As a pretrain-finetune paradigm, it fully leverages the powerful capabilities of various pretrained generative models (e.g., diffusion models, flow models). During the reversed process, it utilizes features from different hierarchical levels, enabling efficient adaptation to downstream tasks. Through theoretical analysis Sec. 3.2 and verification experiments Sec. 4.3, we explain why reversing a pretrained generative process serves as an effective feature extractor for downstream tasks. Our main experiments further demonstrate that this method achieves competitive performance in image classification and OOD detection while effectively adapting community-pretrained text-to-image models for downstream training.

Our contributions can be summarized as follows:

- We propose a general, model-agnostic method for applying pretrained generative models to discriminative tasks.
- We analyze the theoretical foundation of the method and provide empirical findings demonstrating its efficacy.
- Extensive experiments demonstrate the robustness and generalizability of PRG in many image classification datasets.

2. Related Work

Generative versus Discriminative Classifiers Generative classifiers, first proposed in [8, 9, 14, 35, 50, 77], model the conditional distribution $p(x | y)$ and use Bayes' rule to compute $p(y | x) = \frac{p(x|y) \cdot p(y)}{p(x)}$. These classifiers are

typically trained by maximizing the conditional logarithmic likelihood: $\max_{\theta} \mathbb{E}_{(x,y) \sim p(x,y)} \log p(x|y)$.

When using conditional diffusion models, $p(x|y)$ can be computed with the instantaneous change-of-variable formula [11, 22] or approximated by ELBO [26, 57]. Duvenaud et al. [18] advocate an energy-based model to improve calibration, robustness, and out-of-distribution detection. Yang et al. [70] propose HybViT, which combines diffusion models and discriminative classifiers to predict $p(y|x)$. Zimmermann et al. [77] introduce score-based generative classifiers but find them vulnerable to adversarial attacks. Clark and Jaini [14] and Li et al. [35] explore zero-shot generative classifiers using pretrained models like Stable Diffusion [53] to denoise images based on text labels.

In contrast, our approach fine-tunes a classifier on top of the generative model, making it a discriminative classifier that directly estimates $p(y|x)$. Discriminative classifiers are often preferred for classification tasks [48, 62]. Fetaya et al. [19] suggest a trade-off between likelihood-based modeling and classification accuracy, implying that improving generation may hinder classification for generative classifier. However, our two-stage training scheme shows that enhancing the generative model in the pretraining stage can improve discriminative performance during the fine-tuning stage. This suggests a complementary relationship between likelihood-based training and classification accuracy, indicating that discriminative models can benefit from generative pretraining. Thus, we demonstrate a viable approach to leveraging generative models for discriminative tasks.

Representation Learning via Denoising Autoencoders

Generative models have been widely explored for visual representation learning [21, 30, 61]. Denoising autoencoders (DAEs) [6, 20, 64, 65] learn robust representations by reconstructing corrupted input data, which can be interpreted through a manifold learning perspective [63]. Our method extends Vincent et al. [65] by leveraging reversible flow-based generative models like diffusion models.

Diffusion models [26, 28, 57, 59] have shown promise in generation, yet their potential for feature extraction is underexplored. Chen et al. [10] pretrain a Transformer for autoregressive pixel prediction, demonstrating competitive results in representation learning. Their two-stage approach (pre-training a generative model followed by full fine-tuning) closely resembles ours. Additionally, their downsampling of images to low resolution mimics the effect of adding noise, analogous to denoising. However, their model and autoregressive generation scheme are unsuitable for reverse generation and lack efficiency. He et al. [24] pretrain a masked autoencoder, effectively performing denoising by reconstructing incomplete data. Baranchuk et al. [4] show diffusion models improve semantic segmentation, particularly with limited labeled data. Sinha et al. [56] propose

Diffusion-Decoding with contrastive learning to enhance representation quality, while Mittal et al. [46] introduce a Diffusion-based Representation Learning (DRL) framework, leveraging denoising score matching.

Compared to previous methods, **our approach proposes to invert a pretrained continuous-time flow/diffusion generator: by running the model backward, it produces multi-level features that, after light fine-tuning, serve as an unsupervised representation extractor.**

3. Method

To address the issues discussed in Section 2, We propose an approach that follows the conventional two-stage training scheme commonly used in generative modeling applications. In the first stage, we pretrain diffusion or flow models in an unsupervised manner. In the second stage, we fine-tune the models for discriminative tasks by conducting inference in the reverse direction of the generative process.

3.1. Pretrained Reversible Generation Framework

We consider two continuous-time stochastic flow generative models: diffusion and flow models. Let t denote time, where $t \in [1, 0]$ corresponds to the generation, and $t \in [0, 1]$ to its reverse. We denote the data and their transformations as x_t , with x_0 as the original and x_1 as the terminal state.

A diffusion model reverses a forward diffusion process described by $p(x_t | x_0) \sim \mathcal{N}(x_t | \alpha_t x_0, \sigma_t^2 I)$, where α_t is a scaling factor and σ_t^2 denotes the variance. We adopt the Generalized VP-SDE, which uses a triangular function as coefficients to define the diffusion path, referred to as **PRG-GVP** with $\alpha_t = \cos(\frac{1}{2}\pi t)$ and $\sigma_t = \sin(\frac{1}{2}\pi t)$. Due to the Fokker-Planck equation, an equivalent ODE can be derived that shares the same marginal distribution, known as the Probability Flow ODE [58]:

$$\frac{dx_t}{dt} = v(x_t) = f(t)x_t - \frac{1}{2}g^2(t)\nabla_{x_t} \log p(x_t). \quad (1)$$

A flow model deterministically transforms x_0 to x_1 via a parameterized velocity function $v(x_t)$. Inspired by [60], we adopt the flow path formulation, referred to as **PRG-ICFM**, with $p(x_t|x_0, x_1) = \mathcal{N}(x_t|tx_1 + (1-t)x_0)$, and velocity function: $v(x_t|x_0, x_1) = x_1 - x_0$.

We also employ the Optimal Transport Conditional Flow Matching (OTCFM) model [60]. Specifically, we jointly sample (x_0, x_1) from the optimal transport plan π to construct the 2-Wasserstein target distribution. The corresponding velocity field $v(x_t | x_0, x_1)$ is then used to approximate dynamic optimal transport, referred to as **PRG-OTCFM**. In practice, we parameterize the velocity model $v(x_t)$ as v_θ using a network to model the above variants of PRG.

$$\mathcal{L}_{\text{FM}} = \frac{1}{2} \mathbb{E}_{p(x_t)} [\lambda_{\text{FM}}(t) \|v_\theta(x_t) - v(x_t)\|^2] dt \quad (2)$$

After completing pre-training via Eq. (2), and given the invertibility of Eq. (1), we obtain x_1 for a data point x_0 through the same reversal. We denote $x_t = F_\theta(x_0)$ as the extracted features for downstream discriminative tasks, such as classification or regression.

This paper focuses on supervised learning for downstream tasks while simultaneously improving the lower bound of mutual information. Suppose we have a labeled dataset $\mathcal{D}_{\text{finetune}} = \{(x_i, y_i)\}_{i=1}^N$, where x_i is a data point and y_i is its label, potentially differing from the pre-training dataset $\mathcal{D}_{\text{pretrain}}$. Thus, during the fine-tuning stage, we introduce a classifier $p_\phi(y | z)$, parameterized by ϕ , minimizing the cross-entropy loss and flow-matching loss:

$$\mathcal{L}_{\text{total}} = - \sum_{i=1}^N \log p_\phi(y_i | F_\theta(x_i)) + \beta \mathcal{L}_{\text{FM}}(x). \quad (3)$$

Classifier Design We found that the features x_t are well-structured, allowing a simple two-layer MLP with tanh activations to perform well. Notably, enlarging the classifier yielded minimal gains, suggesting that the reversed generative process already captures highly informative features.

Diffusion Step Design To reduce computational costs, the representations used for inference do not necessarily need to traverse the entire process. Instead, our framework allows fine-tuning and inference to start from any selected point along the trajectory. Moreover, more complex downstream tasks typically require greater training/inference steps. Further details are provided in Sec. 4.3.2 and Appendix C.

3.2. Pretrained Reversible Generation Mechanism

pre-training In the traditional autoencoder paradigm, representation learning aims to learn a network that maps the input data X into a useful representation Z for downstream tasks. We parameterize the encoder $p_\theta(z|x)$ with parameters θ . A good representation should retain sufficient information about the input X . In information-theoretic terms [36, 65], this corresponds to maximizing the mutual information $\mathcal{I}(X, Z)$ between the input random variable X and its representation Z . Following the reasoning of Vincent et al. [65], we show that pre-training diffusion or flow models maximizes the mutual information. Specifically, we aim to find the optimal θ^* that maximizes $\mathcal{I}(X, Z)$:

$$\begin{aligned} \theta^* &= \arg \max_{\theta} \mathcal{I}(X, Z) = \arg \max_{\theta} [\mathcal{H}(X) - \mathcal{H}(X|Z)] \\ &= \arg \max_{\theta} [-\mathcal{H}(X|Z)] = \arg \max_{\theta} \mathbb{E}_{p(z,x)} [\log p(x|z)]. \end{aligned} \quad (4)$$

Suppose we have a decoder θ' , which approximates $p_{\theta'}(x|z)$ to recover data x from the latent variable z . By the non-negativity of the KL divergence, we have:

$$\mathbb{D}_{\text{KL}}[p(x|z) || p_{\theta'}(x|z)] \geq 0. \quad (5)$$

$$\mathbb{E}_{p(z,x)}[\log p(x|z)] \geq \mathbb{E}_{p(z,x)}[\log p_{\theta'}(x|z)]. \quad (6)$$

Thus, the right term serves as a lower bound on $\mathcal{I}(X, Z)$.

In our setting, we consider deterministic mappings for both the encoder and decoder, which are invertible. Let us denote the mapping $z = F_{\theta}(x)$, which implies $p_{\theta}(z|x) = \delta(z - F_{\theta}(x))$, which is Dirac delta function. Under these assumptions, we can rewrite the RHS of Eq. (6) as:

$$\begin{aligned} \mathbb{E}_{p(z,x)}[\log p_{\theta'}(x|z)] &= \mathbb{E}_{p(x)}[\log p_{\theta'}(x|z = F_{\theta}(x))] \\ &= -\mathbb{D}_{\text{KL}}(p(x)||p_{\theta'}(x|z = F_{\theta}(x))) - \mathcal{H}(X) \end{aligned} \quad (7)$$

where $p(x)$ is an unknown data distribution that we can sample from. Eq.7 is for maximizing the log likelihood of reconstructing data X , or equivalently, minimizing KL divergence between the data and the generated distribution.

In continuous-time stochastic flow generative models, the inference and generation process are reversible via a same model, which means the decoder parameter θ' and encoder parameter θ are shared parameters. However, we still use these two notation in the later part of this paper for distinguishing the inference and generation process through ODE solver with θ and θ' respectively. For preventing any confusion, we denote x_0 being data variable x and x_1 being the inferred latent variable z . In practice, training generative models using score matching or flow matching objectives does not directly maximize the likelihood of reconstructing x_0 from x_1 . However, as shown in previous works [42, 59, 76], these methods provide a lower bound on the likelihood. Specifically, they establish that:

$$\begin{aligned} \mathbb{D}_{\text{KL}}(p(x_0)||p_{\theta'}(x_0)) &= \mathbb{D}_{\text{KL}}(p(x_1)||p_{\theta'}(x_1)) + \mathcal{L}_{\text{ODE}} \\ &\leq \mathbb{D}_{\text{KL}}(p(x_1)||p_{\theta'}(x_1)) + \sqrt{\mathcal{L}_{\text{SM}}}\sqrt{\mathcal{L}_{\text{Fisher}}}. \end{aligned} \quad (8)$$

The term $\mathbb{D}_{\text{KL}}(p(x_1)||p_{\theta'}(x_1)) \approx 0$ because both $p(x_1)$ and $p_{\theta'}(x_1)$ are approximately the same Gaussian distribution. The losses \mathcal{L}_{ODE} , \mathcal{L}_{SM} and $\mathcal{L}_{\text{Fisher}}$ are defined as:

$$\begin{aligned} \mathcal{L}_{\text{ODE}} &= \frac{1}{2} \mathbb{E}_{p(x_t)} [g^2(t)(s_{\theta}(x) - \nabla \log p(x_t))^T \\ &\quad (\nabla \log p_{\theta'}(x_t) - \nabla \log p(x_t))] dt \\ \mathcal{L}_{\text{SM}} &= \frac{1}{2} \mathbb{E}_{p(x_t)} [g^2(t) \|s_{\theta}(x) - \nabla \log p(x_t)\|^2] dt \\ \mathcal{L}_{\text{Fisher}} &= \frac{1}{2} \mathbb{E}_{p(x_t)} [g^2(t) \|\nabla \log p_{\theta'}(x_t) - \nabla \log p(x_t)\|^2] dt, \end{aligned} \quad (9)$$

where $g(t)$ is a special weighting function, $s_{\theta}(x_t)$ is the score function parameterized by θ , and $p(x_t)$ is the data distribution at time t along forward diffusion process.

From Eq. (8), we observe that using score matching objectives during training can help increase the data likelihood. Flow matching differs from score matching by employing a slightly different weighting factor in the loss function, so it can also be used to increase the likelihood, as shown in

Eq. (2). Therefore, pre-training diffusion or flow models via score matching or flow matching can be viewed as maximizing the mutual information between the data and its representation encoded through the reverse generative process. This relationship between the mutual information and the training loss is guaranteed by the Eq. (8). This process transforms the data distribution into a Gaussian distribution while preserving as much information as possible.

Fine-tuning by Reversing Generation Generative features may not be optimal for discriminative tasks, as reconstruction requires more detail than comprehension. Hence, fine-tuning is necessary to adapt the model for downstream tasks. **A key question is whether the flow model should also be fine-tuned alongside the classifier.** Due to the flow model’s large capacity, reusing the pretrained model seems preferable. Initially, we explored freezing the model while fine-tuning only the classifier. However, this approach underperformed, indicating that fine-tuning both the flow model and the classifier is necessary. Otherwise, a large and complex classifier would be required, reducing efficiency. To fine-tune the entire model, we leverage neural ODEs [11] for efficient gradient computation, optimizing both θ and ϕ via backpropagation. We observed that distinct discriminative tasks correspond to different parts of generative features Appendix D. Thus, fine-tuning can selectively prune or enhance relevant features as needed. In addition, we assess the robustness and generalization of these features against out-of-distribution samples through further experiments.

3.3. Advantages

Model-Agnostic Flexibility Our approach is agnostic to the choice of network architecture for the generative model: it can be a U-Net, Transformer, or any other model. The latent variable Z remains stationary with respect to the data X when using an ODE solver due to the static nature of continuous-time stochastic flow models [58]. This allows different architectures to encode the data while yielding the same latent variable Z . In contrast, methods like DDAE [68] are model-dependent, with the latent variable Z tied to specific network architectures and derived from intermediate activations, making them less flexible and non-stationary.

Infinite-Layer Expressiveness PRG leverages the infinite-layer structure of continuous-time stochastic flow models [11], providing high expressiveness with relatively small parameter sizes. This structure underpins the success of modern generative models like SD3 [53] and DALL-E 3 [7]. By using a reversible process, our method allows for the simultaneous training of discriminative and generative models, offering excellent performance with enhanced capacity.

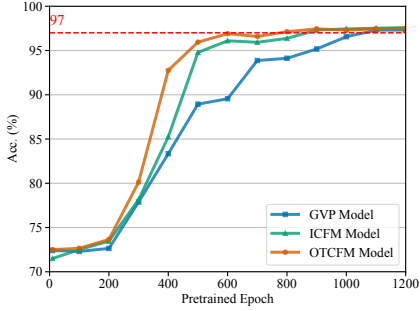


Figure 2. CIFAR-10 Accuracy (y-axis) after fine-tuning checkpoints from different pre-training stages (x-axis).

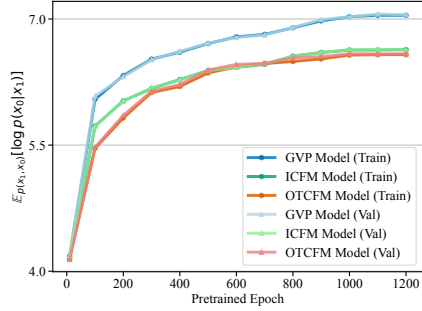


Figure 3. Evaluation of the indicator (y-axis) $\mathbb{E}_{p(x_1, x_0)}[\log p(x_0|x_1)]$ from different pre-training stages (x-axis).

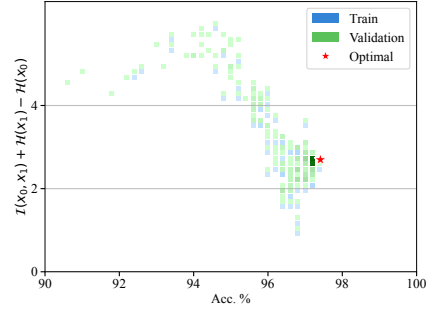


Figure 4. The relationship between y-axis $\mathcal{I}(X_0, X_1) + \mathcal{H}(X_1) - \mathcal{H}(X_0)$ and accuracy (x-axis) during fine-tuning.

Robustness and Generalizability Features extracted by reversing the generative process remain stable along the reverse trajectory, allowing feature use from any point $t \in [0, 1]$ for downstream tasks, not just x_1 . These features are robust to various discretization schemes (e.g., Euler, Runge-Kutta) and time steps (10^3 to 10^1). Our method generalizes well across datasets, enabling fine-tuning of pretrained models for new tasks. Notably, it applies to all diffusion and flow models and has been successfully transferred to community-developed text-to-image models in Sec. 4.4.3.

A more detailed analysis of why PRG is effective can be found in Appendix A.

4. Experiments

4.1. Settings

For generative model selection, we adopt three diffusion and flow-based models: GVP, ICFM, and OTCFM. For pre-training, we followed the protocol from [49], using the Adam optimizer [31] with a fixed learning rate of 1×10^{-4} for 1,200 epochs on the respective training sets. For downstream image classification, we adopted the configuration from [40], using AdamW for 200 epochs with a cosine decay learning rate schedule and a 5-epoch linear warm-up. The training used a batch size of 128 and an initial learning rate of 0.001. We followed DDAE [68]’s setting (64 resolution, simple data augmentation, no mixup [74], no cutmix [72]) for fair comparison, yet our method achieves SOTA diffusion classifier performance. Further details, **including hyperparameters and efficiency statistics for training and inference**, are provided in Appendices B.1 to B.3.

4.2. baselines

We select state-of-the-art (SOTA) representation models as baselines to evaluate PRG on CIFAR-10 [33], Tiny-ImageNet [34], and ImageNet [54]. Baselines include classical discriminative models like WideResNet-28-10 [73] and ResNeXt-29-16 \times 64d [69], as well as diffusion-based gen-

erative models such as GLOW [19], Energy Model [18], SBGC [77], HybViT [70], and DDAE [68]. For out-of-distribution (OOD) evaluation, we compare PRG with PGD [44], PLAT [66], AugMix [25], AutoAug [15], SBGC, and PDE+ [71] on CIFAR-10-C and Tiny-ImageNet-C. All results are from their original papers.

4.3. Verification Analysis

4.3.1. Better Pre-training Lead to Better Fine-tuning?

To explore whether better pre-training improves fine-tuning, we compared the classification accuracies of models fine-tuned from different pre-training epochs on CIFAR-10 (Fig. 2). We also computed the mutual information during pre-training, which increases monotonically, as shown in Fig. 3. During pre-training, we have

$$\mathbb{E}_{p(x_1, x_0)}[\log p(x_0 | x_1)] = \mathcal{I}(X_0, X_1) - \mathcal{H}(X_0),$$

where $\mathcal{H}(X_0)$ is the (constant) entropy of the data distribution and $\mathcal{H}(X_1)$ is the (constant) entropy of the Gaussian distribution. Hence, $\mathbb{E}_{p(x_1, x_0)}[\log p(x_0 | x_1)]$ faithfully reflects the mutual information $\mathcal{I}(X_0, X_1)$ during pre-training. In contrast, during fine-tuning, $\mathbb{E}_{p(x_1, x_0)}[\log p(x_0 | x_1)]$ no longer represents $\mathcal{I}(X_0, X_1)$, because $\mathcal{H}(X_1)$ is no longer the entropy of a Gaussian distribution. Nonetheless, the quantity $\mathcal{I}(X_0, X_1) + \mathcal{H}(X_1) - \mathcal{H}(X_0)$ still captures the change in $\mathcal{I}(X_0, X_1)$ to a large extent. We compute log-probability density $\log p(x_0|x_1)$ using the adjoint method or Neural ODEs [11] for computational feasibility:

$$\begin{bmatrix} x_0 - x_1 \\ \log p(x_0) - \log p(x_1) \end{bmatrix} = \int_{t=1}^{t=0} \begin{bmatrix} v_\theta(x_t, t) \\ -\text{Tr} \left(\frac{\partial v_\theta}{\partial x_t} \right) \end{bmatrix} dt \quad (10)$$

For efficient computation of $\log p(x_0|x_1)$, we employ the Hutchinson trace estimator [22, 29]:

$$\text{Tr} \left(\frac{\partial v_\theta}{\partial x_t} \right) = E_{p(\epsilon)}[\epsilon^T \frac{\partial v_\theta}{\partial x_t} \epsilon] \quad (11)$$

where ϵ is a standard Gaussian random vector.

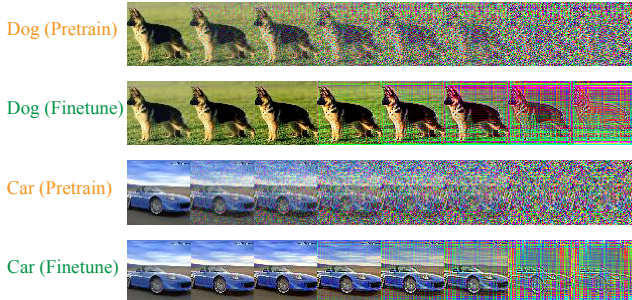


Figure 5. Reverse generation from x_0 to x_1 with full pre-training and fine-tuning.

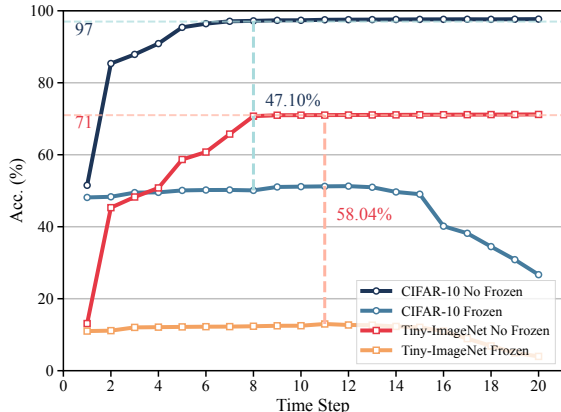


Figure 6. Accuracy on CIFAR-10 and Tiny-ImageNet when fine-tuned from different time steps along the generation trajectory.

Moreover, we analyzed the relationship between accuracy and mutual information during fine-tuning, as shown in Fig. 4. Models without pre-training achieved approximately 73.5% accuracy. In contrast, models with more pre-training exhibit higher mutual information and classification accuracy, suggesting that stronger generative capability enhances fine-tuning performance. During fine-tuning, we observed a decline in $\mathcal{I}(x_0, x_1) + \mathcal{H}(x_1) - \mathcal{H}(x_0)$ for both the training and validation sets, suggesting that the model filters out unnecessary features to improve downstream classification, which can also be observed in Fig. 5 as some visual representations are kept while others are discarded. In summary, **sufficient pre-training is crucial for optimal performance in inverse generative classification.**

4.3.2. What is a Reasonable Fine-tuning Strategy?

We first examine whether a frozen flow model provides meaningful features. For this, we use features x_t along the sampling trajectory from x_0 to x_1 under two conditions: (1) freezing the generative model’s parameters (light lines) and (2) end-to-end training of both the generative model and the classification head (dark lines). As shown in Fig. 6, the performance gap is substantial—47.10% on CIFAR-10 and

58.04% on Tiny-ImageNet—demonstrating the **critical importance of updating the generative model’s parameters during fine-tuning.**

Additionally, on CIFAR-10, initiating fine-tuning from later stages of the sampling trajectory ($x_{1/4}$ to x_1) results in progressively better performance. On TinyImageNet, optimal results are achieved when fine-tuning starts from $x_{1/2}$ to x_1 . This difference likely stems from the complexity of the datasets: **more complex datasets require stronger feature extraction, necessitating longer trajectory lengths for effective fine-tuning.** For an in-depth analysis, see Appendix A.1.1; complementary experiments on optimal fine-tuning strategies are provided in Appendix C.

4.3.3. Is the Model a Continuous Feature Extractor?

Residual networks, such as ResNet-50 [23], achieve feature extraction by constructing discrete-time methods through a series of composite transformations: $\mathbf{h}_{t+1} = \mathbf{h}_t + f(\mathbf{h}_t, \theta_t)$. In contrast, we construct the feature extractor using the ODE specified by the neural network, namely $\frac{d\mathbf{h}(t)}{dt} = f(\mathbf{h}(t), t, \theta)$. Tabs. 1 and 2 demonstrate that although we trained using a discrete sample length of $t_{\text{span}} = 20$, we can evaluate the model using any reasonable inference steps. Furthermore, feature extraction is not limited to a specific training point (such as the midpoint); extracting features within its vicinity (within 20%) still achieves high performance on downstream tasks. **These findings validate our method can serve as a continuous feature extractor.**

Inference Steps	20	100	500	800	1000
PRG-GVP Acc. (%)	97.25	97.28	97.30	97.31	97.31
PRG-ICFM Acc. (%)	97.32	97.33	97.34	97.34	97.34
PRG-OTCFM Acc. (%)	97.42	97.43	97.43	97.44	97.44

Table 1. The checkpoint, fine-tuned at the midpoint of a trajectory with $t_{\text{span}} = 20$, is evaluated on CIFAR-10 using the midpoints of trajectories with varying inference steps (from 20 to 1000).

Offset from Midpoint %	-20	-10	10	20
PRG-GVP Acc. (%)	97.02	97.17	97.12	97.08
PRG-ICFM Acc. (%)	97.23	97.30	97.20	97.10
PRG-OTCFM Acc. (%)	97.31	97.32	97.26	97.13

Table 2. Accuracy at the midpoints of trajectories with varying offsets for $t_{\text{span}} = 1000$ (use the same checkpoint from Tab. 1).

4.4. Main Results

4.4.1. Performance on Image Classification

To comprehensively evaluate our method, we conducted experiments on three image classification datasets, as detailed in Tabs. 3 to 5. Our methods achieved accuracies of

Method	Param. (M)	Acc. (%)
Discriminative methods		
WideResNet-28-10 [73]	36	96.3
ResNeXt-29-16 * 64d [69]	68	96.4
Generative methods		
GLOW [19]	N/A	84.0
Energy model [18]	N/A	92.9
SBGC [77]	N/A	95.0
HybViT [70]	43	96.0
DDAE [68]	36	97.2
Our methods		
PRG-GVP-onlyPretrain	42	54.10
PRG-GVP-S	42	97.35
PRG-ICFM-S	42	97.59
PRG-OTCFM-S	42	97.65

Table 3. Comparison on the CIFAR-10 dataset with various algorithms. All results are reported from their original papers.

Method	Param. (M)	Acc. (%)
Discriminative methods		
WideResNet-28-10 [73]	36	69.3
Generative methods		
HybViT [70]	43	56.7
DDAE [68]	40	69.4
Our methods		
PRG-GVP-onlyPretrain	42	15.34
PRG-GVP-S	42	70.98
PRG-ICFM-S	42	71.12
PRG-OTCFM-S	42	71.33

Table 4. Comparison on the Tiny-ImageNet dataset with various algorithms. All results are reported from their original papers.

97.59%, 71.12%, and 78.1% on CIFAR-10, Tiny-ImageNet, and ImageNet, respectively, surpassing all existing generative approaches, as well as supervised methods like WideResNet-28[73]. However, these results still trail behind recent transformer-based supervised learning architectures like SwinTransformer [40]. We hypothesize that integrating transformer architectures and leveraging VAEs to handle high input resolutions could further enhance the performance. This avenue will be explored in future work.

4.4.2. Out-of-Distribution Robustness

To address the degradation in out-of-distribution (OOD) due to common image corruptions or adversarial perturbations [77], data augmentations and adversarial training are typically employed. However, recent studies [8, 35, 77] have indicated that generative-based classifiers, without requiring additional data, often exhibit superior OOD robustness. As illustrated in Tab. 6 and Appendix C.3, our method showcases remarkable robustness on these two datasets with only

Method	Param. (M)	Acc. (%)
Discriminative methods		
ViT-L/16 (384 ²) [17]	307	76.5
ResNet-152 (224 ²) [23]	60	77.8
Swin-B (224 ²) [40]	88	83.5
Generative methods		
HybViT (32 ²) [70]	43	53.5
DMSZC-DiT XL2 (256 ²) [35]	338	77.5
iGPT-L (48 ²) [10]	1362	72.6
Our methods		
PRG-GVP-onlyPretrain (64 ²)	122	20.18
PRG-GVP-XL (64 ²)	122	77.84
PRG-ICFM-XL (64 ²)	122	78.12
PRG-OTCFM-XL (64 ²)	122	78.13

Table 5. Comparison on the ImageNet dataset with various algorithms. All results are reported from their original papers, the values in parentheses indicate the input image size.

Model	CIFAR-10		CIFAR-10-C	
	Clean	Corr Severity	All	w/ Noise
Adversarial Training				
PGD *[44]	93.91	83.08 ↓ 10.83	82.10 ↓ 11.81	
PLAT * [66]	94.73	88.28 ↓ 6.45	88.56 ↓ 6.17	
Noise Injection				
RSE *[38]	95.59	77.86 ↓ 17.73		N/A
ENResNet *[66]	83.33	74.34 ↓ 8.99		N/A
Data Augment				
AugMix § [25]	95.83	89.09 ↓ 6.74	88.71 ↓ 7.12	
AutoAug *[15]	95.61	85.37 ↓ 10.24	76.47 ↓ 19.14	
Generative Methods				
SBGC [77]	95.04	76.24 ↓ 18.80	75.38 ↓ 19.66	
PDE+ [71]	95.59	89.11 ↓ 6.48	85.59 ↓ 10.00	
Our methods				
PRG-GVP-S	97.35	91.21 ↓ 6.14	92.17 ↓ 5.18	
PRG-ICFM-S	97.59	92.13 ↓ 5.46	93.07 ↓ 4.52	
PRG-OTCFM-S	97.65	92.26 ↓ 5.39	93.10 ↓ 4.55	

Table 6. (OOD: Image Corruptions) Comparison on CIFAR-10-C, including Noise, Blur, Weather, and Digital corruptions. Results are referenced from original papers or [71, 77]. * denotes ResNet-18 as the base model, while § indicates ResNeXt-29.

simple data augmentation.

4.4.3. Transferring Features

A main goal of the pre-training/fine-tuning paradigm is to learn transferrable features. We hypothesize that advancements like powerful pretrained models in the generative model community can contribute to our method. To validate this, we conducted experiments by transferring the pretrained SiT-XL [43] model to two other datasets. Since

Method	Param. (M)	Cifar Acc.	Tiny Acc.
Classic self-supervised learning methods			
MAE (ViT-B/16) [41]	86	96.5	76.5
SimCLR Res-50-4x [12]	375	98.6	N/A
Generative methods			
DDAE-DiT-XL2 [68]	338	98.4	77.8
iGPT-L [10]	1362	99.0	N/A
Our methods			
PRG-SiT-XL2	338	98.72	78.33

Table 7. Comparison of models pretrained on ImageNet-1k and transferred (fine-tuning) to CIFAR-10 and Tiny-ImageNet.

SiT provides only class-conditional checkpoints, we adopted an unconditional approach by setting the label parameter to null. Specifically, we fine-tuned the model for 28 epochs on the CIFAR-10 dataset and for 45 epochs on the Tiny-ImageNet dataset. As demonstrated in Tab. 7, our method shows superior transfer learning performance. The results suggest that our algorithm benefits from larger datasets and the latent space of generative models.

4.5. Ablation Studies

In this section, we conduct three ablation studies to further evaluate the applicability of our method. Additional experiments, including analyses of loss type, ODE solver choice, dual-task supervision, and t_{span} , are provided in Appendix C.

4.5.1. Generative Model Type

We explored various generative model types, particularly focusing on path selection. To assess the impact of these paths on our method, we conducted experiments with three model types on the CIFAR-10 and Tiny-ImageNet datasets. Following [39], we quantified the straightness of various continuously differentiable trajectories. Interestingly, while all three model variants achieved comparable performance (Tab. 8), models with higher straightness required fewer pre-training steps to attain equivalent performance levels and demonstrated potential for superior outcomes.

Generative Model Type	GVP	ICFM	OTCFM
CIFAR-10 (Accuracy)	97.35	97.59	97.65
First to 97% (Epoch)	162	135	128
Straightness	7.36	0.34	0.11
Tiny-ImageNet (Accuracy)	70.98	71.12	71.33
First to 70% (Epoch)	157	125	106
Straightness	6.25	0.54	0.15

Table 8. (**Generative Model Type**) Performance of different generative model types on the CIFAR-10 and Tiny-ImageNet datasets.

4.5.2. Effect of β on Accuracy

During pre-training, we introduced β in Eq. (3) to enhance the mutual information lower bound while balancing the downstream task objective. Table 9 shows PRG scores across datasets for different β values.

β	1	10	100
Acc. (CIFAR/Tiny)	0.973/0.712	0.974/0.711	0.965/0.673

Table 9. (**β values**) Accuracy scores of PRG with varying β values on the CIFAR-10 and Tiny-ImageNet datasets.

4.5.3. Scaling Up Network Parameters

We investigated the impact of scaling up U-Net models of varying network parameter sizes [49]. Table 10 provides a comparative analysis following fine-tuning on the CIFAR-10 and Tiny-ImageNet datasets. The findings suggest that increasing the model size results in a modest enhancement of final classification accuracy.

Model	Param (M)	CIFAR Acc.	Tiny Acc.
PRG-ICFM-B	32	97.35	70.94
PRG-ICFM-S	42	97.59	71.12
PRG-ICFM-L	66	97.75	71.30
PRG-ICFM-XL	122	97.88	71.80

Table 10. (**Scaling Capability**) Performance gains of increasing U-Net model sizes on the CIFAR-10 and Tiny-ImageNet datasets.

5. Conclusion

This paper proposes to turn a pretrained continuous-time flow/diffusion generator upside-down: running the model backward produces multi-level features that, after light fine-tuning, serve as an unsupervised representation extractor. We systematically investigate the necessary designs of the two-stage pre-training and fine-tuning paradigm. Theoretical analysis supports the efficacy of the first-stage generative pre-training, while empirical verifications provide insights for designing the fine-tuning pipeline. Leveraging these simple yet effect techniques and findings, our method achieves state-of-the-art performance in image classification tasks using generative models. Additional experiments, such as those on OOD problems and transfer learning scenarios, further validate the robustness and other advantages of our approach. However, there remains room for improvement, including the integration of large foundation pretrained generative models from the open-source community and optimizing the training epoch/speed for faster adaptation to downstream tasks.

Acknowledgments. This work was supported by the Shanghai Artificial Intelligence Laboratory, in part by the National Key R&D Program of China under Grant No. 2022ZD0119301, and in part by the National Natural Science Foundation of China under Grant No. U21A20485 and No. 62406141. The Work was done during Rongkun Xue’s internship at Shanghai AI Laboratory.

References

- [1] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, pages 1–3, 2024. 1
- [2] Michael Samuel Alberg and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. 1
- [3] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22669–22679, 2023. 13
- [4] Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. *arXiv preprint arXiv:2112.03126*, 2021. 2
- [5] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013. 2
- [6] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. *Advances in neural information processing systems*, 26, 2013. 2
- [7] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science*. <https://cdn.openai.com/papers/dall-e-3.pdf>, 2(3):8, 2023. 4
- [8] Huanran Chen, Yinpeng Dong, Shitong Shao, Zhongkai Hao, Xiao Yang, Hang Su, and Jun Zhu. Diffusion models are certifiably robust classifiers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2, 7
- [9] Huanran Chen, Yinpeng Dong, Zhengyi Wang, Xiao Yang, Chengqi Duan, Hang Su, and Jun Zhu. Robust classification via a single diffusion model. *arXiv preprint arXiv:2305.15241*, 2023. 2
- [10] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020. 1, 2, 7, 8
- [11] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. 2, 4, 5, 12
- [12] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 1, 8
- [13] Xi Chen, Yan Duan, Rein Houthoofd, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, 29, 2016. 2
- [14] Kevin Clark and Priyank Jaini. Text-to-image diffusion models are zero shot classifiers. *Advances in Neural Information Processing Systems*, 36, 2024. 2
- [15] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 113–123, 2019. 5, 7, 14
- [16] Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. *Advances in neural information processing systems*, 32, 2019. 2
- [17] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 7
- [18] David Duvenaud, Jackson Wang, Jorn Jacobsen, Kevin Swersky, Mohammad Norouzi, and Will Grathwohl. Your classifier is secretly an energy based model and you should treat it like one. In *ICLR 2020*, 2020. 2, 5, 7
- [19] Ethan Fetaya, Jörn-Henrik Jacobsen, Will Grathwohl, and Richard Zemel. Understanding the limitations of conditional generative models. *arXiv preprint arXiv:1906.01171*, 2019. 2, 5, 7
- [20] Krzysztof J Geras and Charles Sutton. Scheduled denoising autoencoders. *arXiv preprint arXiv:1406.3269*, 2014. 2
- [21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 2
- [22] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, and David Duvenaud. Scalable reversible generative models with free-form continuous dynamics. In *International Conference on Learning Representations*, 2019. 2, 5
- [23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 6, 7
- [24] 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*, pages 16000–16009, 2022. 2
- [25] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019. 5, 7, 14
- [26] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

- [27] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems*, pages 8633–8646. Curran Associates, Inc., 2022. 1
- [28] Xirui Hu, Jiahao Wang, Hao Chen, Weizhan Zhang, Benqi Wang, Yikun Li, and Haishun Nan. Dynamicid: Zero-shot multi-id image personalization with flexible facial editability. *arXiv preprint arXiv:2503.06505*, 2025. 2
- [29] M.F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics - Simulation and Computation*, 19(2): 433–450, 1990. 5
- [30] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2
- [31] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5
- [32] Klim Kireev, Maksym Andriushchenko, and Nicolas Flammarion. On the effectiveness of adversarial training against common corruptions. In *Uncertainty in Artificial Intelligence*, pages 1012–1021. PMLR, 2022. 14
- [33] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 5
- [34] Ya Le and Xuan S. Yang. Tiny imagenet visual recognition challenge. 2015. 5
- [35] Alexander C Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *CVPR*, pages 2206–2217, 2023. 2, 7
- [36] Ralph Linsker. An application of the principle of maximum information preservation to linear systems. *Advances in neural information processing systems*, 1, 1988. 3
- [37] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. 1
- [38] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *Proceedings of the european conference on computer vision (ECCV)*, pages 369–385, 2018. 7, 14
- [39] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 1, 8
- [40] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 5, 7
- [41] Zhengqi Liu, Jie Gui, and Hao Luo. Good helper is around you: Attention-driven masked image modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1799–1807, 2023. 8
- [42] Cheng Lu, Kaiwen Zheng, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Maximum likelihood training for score-based diffusion odes by high order denoising score matching. In *International Conference on Machine Learning*, pages 14429–14460. PMLR, 2022. 4
- [43] 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. *arXiv preprint arXiv:2401.08740*, 2024. 7
- [44] Aleksander Madry. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 5, 7, 14
- [45] Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. *arXiv preprint arXiv:2103.16091*, 2021. 1
- [46] Sarthak Mittal, Korbinian Abstreiter, Stefan Bauer, Bernhard Schölkopf, and Arash Mehrjou. Diffusion based representation learning. In *International Conference on Machine Learning*, pages 24963–24982. PMLR, 2023. 3
- [47] Kevin P Murphy. *Probabilistic machine learning: Advanced topics*. MIT press, 2023. 12
- [48] Andrew Ng and Michael Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems*. MIT Press, 2001. 2
- [49] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 5, 8
- [50] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Animashree Anandkumar. Diffusion models for adversarial purification. In *Proceedings of the 39th International Conference on Machine Learning*, pages 16805–16827. PMLR, 2022. 2
- [51] Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky T. Q. Chen. Multisample flow matching: Straightening flows with minibatch couplings. In *Proceedings of the 40th International Conference on Machine Learning*, pages 28100–28127. PMLR, 2023. 1
- [52] Alec Radford. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2
- [53] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022. 1, 2, 4
- [54] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211 – 252, 2014. 5
- [55] Dazhong Shen, Guanglu Song, Zeyue Xue, Fu-Yun Wang, and Yu Liu. Rethinking the spatial inconsistency in classifier-free diffusion guidance. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9370–9379. IEEE, 2024. 1
- [56] Abhishek Sinha, Jiaming Song, Chenlin Meng, and Stefano Ermon. D2c: Diffusion-decoding models for few-shot conditional generation. In *Advances in Neural Information Processing Systems*, 2021. 2
- [57] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using

- nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. [2](#)
- [58] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. [2](#), [3](#), [4](#), [12](#)
- [59] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in neural information processing systems*, 34:1415–1428, 2021. [2](#), [4](#)
- [60] Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. Expert Certification. [3](#)
- [61] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. [2](#)
- [62] Vladimir N. Vapnik. The nature of statistical learning theory. *Springer*, 1995. [2](#)
- [63] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011. [2](#)
- [64] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008. [2](#)
- [65] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010. [2](#), [3](#)
- [66] Bao Wang, Binjie Yuan, Zuoqiang Shi, and Stanley J Osher. Enresnet: Resnets ensemble via the feynman–kac formalism for adversarial defense and beyond. *SIAM Journal on Mathematics of Data Science*, 2(3):559–582, 2020. [5](#), [7](#), [14](#)
- [67] Fu-Yun Wang, Zhaoyang Huang, Alexander Bergman, Dazhong Shen, Peng Gao, Michael Lingelbach, Keqiang Sun, Weikang Bian, Guanglu Song, Yu Liu, et al. Phased consistency models. *Advances in neural information processing systems*, 37:83951–84009, 2024. [1](#)
- [68] 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*, pages 15802–15812, 2023. [2](#), [4](#), [5](#), [7](#), [8](#)
- [69] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. [5](#), [7](#)
- [70] Xiulong Yang, Sheng-Min Shih, Yinlin Fu, Xiaoting Zhao, and Shihao Ji. Your vit is secretly a hybrid discriminative-generative diffusion model. *arXiv preprint arXiv:2208.07791*, 2022. [2](#), [5](#), [7](#)
- [71] Yige Yuan, Bingbing Xu, Bo Lin, Liang Hou, Fei Sun, Huawei Shen, and Xueqi Cheng. Pde+: Enhancing generalization via pde with adaptive distributional diffusion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 16614–16622, 2024. [5](#), [7](#), [14](#)
- [72] Sangdoon Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. [5](#)
- [73] Sergey Zagoruyko. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. [5](#), [7](#)
- [74] Hongyi Zhang. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. [5](#)
- [75] Jinouwen Zhang, Rongkun Xue, Yazhe Niu, Yun Chen, Jing Yang, Hongsheng Li, and Yu Liu. Revisiting generative policies: A simpler reinforcement learning algorithmic perspective, 2024. [1](#)
- [76] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Improved techniques for maximum likelihood estimation for diffusion odes. In *International Conference on Machine Learning*, pages 42363–42389. PMLR, 2023. [4](#)
- [77] Roland S Zimmermann, Lukas Schott, Yang Song, Benjamin A Dunn, and David A Klindt. Score-based generative classifiers. *arXiv preprint arXiv:2110.00473*, 2021. [2](#), [5](#), [7](#)

Pretrained Reversible Generation as Unsupervised Visual Representation Learning

Appendix

A. Why PRG is effective?

What a model can generate, it can understand: pixel-perfect reconstruction means the visual breadcrumbs are all there. It just needs a bit of fine-tuning to tie the generative pathway to the labels. Raw-image baselines often fall prey to "shortcut learning". Because neighboring feature pixels have heavily overlapping receptive fields, the network may rely on trivial cues and end up with weak representations. In contrast, PRG reuses the pretrained reverse generative pathway, recycling each ODE-solver output as the input for the next iteration. This loop drives feature extraction at a semantic level without losing low-level detail and the injected noise perturbations make the representations significantly more robust. Sec. 4.3.2 show that every point along the trajectory possesses strong representational capacity, and the OOD experiments also provide compelling evidence for this claim.

In the first pre-training stage, since no downstream task information is available, it remains unclear which features are most relevant. Consequently, no compression is performed during pre-training. Instead, the model aims to approximate the optimal representation as closely as possible in the absence of downstream information by leveraging unconditional flow matching to learn representations, effectively maximizing the lower bound of mutual information.

According to the manifold assumption [47, Chapter 20], data lies on a low-dimensional manifold \mathcal{M} of intrinsic dimension d^* , which is significantly smaller than the ambient dimension D . Although, in theory, intermediate latents encode the same information as the original data, using these latents as inputs for downstream tasks is more meaningful. A well-trained flow model effectively extracts data from the low-dimensional manifold \mathcal{M} and lifts it to the ambient space \mathbb{R}^D , ensuring that every sample in \mathbb{R}^D remains semantically meaningful. In contrast, directly sampling from the original data space does not necessarily preserve semantic coherence. For instance, generating a 64×64 image by sampling from a 64×64 multivariate Gaussian distribution would typically result in a meaningless noise-like image resembling a snowflake pattern.

Moreover, since the flow trajectories of an ordinary differential equation (ODE) do not intersect [11], points within a given region remain confined, thereby preserving topological relationships throughout the transformation process.

Subsequent fine-tuning for downstream tasks is equally crucial. By adopting techniques such as optimal transport matching and gradient guidance, the model undergoes ef-

ficient adaptation, selectively discarding redundant information while selecting and enhancing task-critical features. As shown in Figure 4, mutual information decreases during fine-tuning, whereas task accuracy improves.

A.1. What's the difference between fine-tuning and training a classifier based on the ODE architecture?

The effectiveness of fine-tuning depends significantly on whether the model undergoes pre-training beforehand. If a classifier is trained directly using the ODE architecture without pre-training, the model must learn meaningful intermediate latents solely through a supervised loss. This approach often fails, as the model struggles to discover good latent representations in the absence of prior knowledge. Our experiments (Appendix A.1.1) also confirm that this method performs poorly.

In contrast, pre-training allows the model to extract a rich set of useful features as intermediate latents. During fine-tuning, the model can then selectively retain and enhance features relevant to downstream tasks while compressing redundant information in the latent space. This process leads to a more effective and structured representation, ultimately improving downstream task performance.

A.1.1. Comparison with Generation without pre-training

We compare PRG with classifiers trained without generative pre-training. To ensure fairness, we train the latter model for 600 epochs until its performance no longer improves. As shown in Tab. 1, PRG without pre-training, relying solely on a single supervision signal, may discard useful information, leading to suboptimal performance. This highlights that the intermediate latent variables of a pretrained flow model provide a strong initialization, requiring only slight fine-tuning for optimal feature extraction.

	CIFAR-10	Tiny-ImageNet	ImageNet
PRG w/o pre-training	0.70	0.35	0.42
PRG	0.97	0.71	0.78

Table 1. **Effect of pre-training:** Performance comparison of PRG with and without pre-training on the CIFAR-10 and Tiny-ImageNet.

A.2. Model-agnosticity

Our method follows the model-agnostic principle as defined in Appendix D.5 of [58], where different architectures

achieve comparable encoding quality through flow matching. This indicates that architectural constraints stem from implementation choices rather than the framework itself.

Notably, while early diffusion models primarily relied on U-Nets for practical stability, recent work [3] has demonstrated that flow matching can be successfully achieved with minimal architectural requirements, such as shallow skip connections. In this work, we achieve state-of-the-art performance across both U-Net (Sec. 4.4.1) and Transformer (Sec. 4.4.3) architectures, further validating the generality of our approach.

A.3. Infinite-Layer Expressiveness

The infinite-layer extractor boosts accuracy by extending training rather than inference steps (Fig.6: CIFAR 0.46 vs. 0.97, Tiny 0.19 vs. 0.71). However, gains plateau after a certain point, with more complex datasets requiring longer training for optimal performance. Meanwhile, the optimal flow-matching setup achieves full experimental coverage in just 5 inference steps. **Full evaluation takes 3 seconds for CIFAR and 9 s for the Tiny-ImageNet testing set.**

B. Additional Implementation Details

B.1. Training Process Details

Dataset	CIFAR-10	Tiny-ImageNet	ImageNet
GPU	8×A100	8×A100	8×A100
Optimizer	Adam	Adam	Adam
LR base	1e-4	1e-4	1e-4
Epochs	1000	1000	2000
Batch Size	256	128	128

Table 2. Experimental settings across datasets for **pre-training**.

To enhance the reproducibility of results across various multi-stage and multi-GPU experiments, we calculate the learning rate using Eq. (1).

$$\begin{aligned}
 \text{LR} &= \text{LR}_{\text{base}} \times \frac{\text{num processes} \times \text{Batch Size}}{512} \\
 \text{Warmup LR} &= \text{Warmup LR}_{\text{base}} \times \frac{\text{num processes} \times \text{Batch Size}}{512} \\
 \text{Min LR} &= \text{Min LR}_{\text{base}} \times \frac{\text{num processes} \times \text{Batch Size}}{512}
 \end{aligned}
 \tag{1}$$

B.2. Evaluation of Training Efficiency

Most research experiments, including the main experiments and ablation studies, are completed within several hours. To demonstrate training efficiency concretely, Tab. 4 reports the run-time per epoch for each data set used in our experiments.

Dataset	CIFAR-10	Tiny-ImageNet	ImageNet
GPU	4×A100	8×A100	32×A100
Optimizer	AdamW	AdamW	AdamW
Eps	1e-8	1e-8	1e-8
Betas	(0.9, 0.999)	(0.9, 0.999)	(0.9, 0.999)
LR base	1.25e-4	1.25e-4	1.25e-4
Weight Decay	0.05	0.05	0.05
Scheduler	CosineLR	CosineLR	CosineLR
Warmup LR base	1.25e-7	1.25e-7	1.25e-7
Min LR base	1.25e-6	1.25e-6	1.25e-6
Epochs	200	200	200
Warmup Epochs	5	10	10
Image Size	32	64	64
Batch Size	256	128	128
T Span	20	32	64
Solver	Euler	Euler	Euler

Table 3. Experimental settings across datasets for **fine-tuning**.

$t_{\text{span}}/t_{\text{cutoff}}$	CIFAR-10	TinyImageNet	ImageNet*
20/5	49s	307s	N/A
20/10	103s	681s	2220s
20	187s	1437s	3480s
32/16	170s	1137s	N/A

Table 4. Mean time cost per epoch during the training process. * means that the model is trained on 4*8 A100 GPUs. t_{span} represents the total sampling length, while t_{cutoff} indicates the point along the trajectory where fine-tuning begins. For example, 20/10 means a trajectory spanning 20 steps from x_0 to x_1 , with fine-tuning starting from the midpoint of the trajectory.

B.3. Evaluation of Inference Efficiency

Table 5 presents the inference efficiency of our method on the CIFAR-10 and Tiny-ImageNet test sets.

Model	CIFAR-10 (s)	Tiny-ImageNet (s)
PRG-GVP-S	4	10
PRG-ICFM-S	3	9
PRG-OTCFM-S	3	8

Table 5. Mean inference time per epoch on the CIFAR-10 and Tiny-ImageNet test datasets.

C. Ablation Studies

C.1. Loss Type

Tab. 6 shows the results of different loss types. Compared to the standard cross entropy loss, label smoothing reduces overconfident predictions, improves model calibration, and improves robustness.

Dataset	LabelSmooth Loss	Cross-Entropy Loss
CIFAR-10	97.59	96.18
TinyImageNet	71.12	70.15

Table 6. (**Loss Type**) Comparison of LabelSmooth Loss and Cross-Entropy Loss on different datasets.

C.2. ODE Solver Type

During fine-tuning, we evaluated different ODE solvers for the reverse process: Euler (first-order), Midpoint (second-order via midpoint evaluations), RK4 (fourth-order Runge-Kutta), and Dopri5 (adaptive step sizes with a fifth-order method). Tabs. 7 and 8 compares their performance on the Cifar-10, Tiny-ImageNet dataset. The results show no significant performance differences, underscoring the method’s consistent effectiveness across various solvers.

solver_type	Euler	Midpoint	RK4	Dopri5
OTCFM	97.65	97.63	97.66	97.72
ICFM	97.59	97.55	97.56	97.61
GVP	97.35	97.30	97.32	97.41

Table 7. (**ODE Solver**) Performance of various solvers on Cifar-10. Different solvers don’t yield obvious differences.

Solver Type	Euler	Midpoint	RK4	Dopri5
PRG-OTCFM	71.33	71.29	71.30	71.36
PRG-ICFM	71.12	71.11	71.13	71.23
PRG-GVP	70.89	70.99	70.84	70.85

Table 8. (**ODE Solver**) Performance of various solvers on Tiny-ImageNet. Different solvers don’t yield obvious differences.

C.3. Details of Out-of-Distribution Experiments

There is no direct correspondence between the test images of Tiny ImageNet and Tiny ImageNet-C, and the images in Tiny ImageNet-C do not overlap with the training images of Tiny ImageNet. We report the comparative results on Tiny ImageNet-C in Tab. 9.

C.4. The Number of Timesteps

Our findings in Tab. 10 show that longer time spans generally lead to better accuracy. On CIFAR-10 with the ICFM flow model, a t -span of 10 achieves accuracy comparable to the best result at $t = 100$. In contrast, TinyImageNet requires a t -span of 15 to achieve similar performance.

D. Reverse Generation Process

Fig. 1 illustrates the reverse generation process from x_0 to x_1 after fine-tuning. Furthermore, Figs. 2 and 3 present the

Tiny-ImageNet-C			
Method	Clean	Average	Corruption-5
Adversarial Training			
PGD [44]	51.08	33.46 ↓ 17.62	24.00 ↓ 27.08
PLAT [32]	51.29	37.92 ↓ 13.37	29.05 ↓ 22.24
Noise Injection			
RSE [38]	53.74	27.99 ↓ 25.75	18.92 ↓ 34.82
ENResNet [66]	49.26	25.83 ↓ 23.43	19.01 ↓ 30.25
Data Augment			
AugMix [25]	52.82	37.74 ↓ 15.08	28.66 ↓ 24.16
AutoAug [15]	52.63	35.14 ↓ 17.49	25.36 ↓ 27.27
Generative Methods			
PDE+ [71]	53.72	39.41 ↓ 14.31	30.32 ↓ 23.40
PRG-ICFM-S (ours)	56.85	46.93 ↓ 9.92	33.32 ↓ 23.53

Table 9. (**OOD: extrapolated datasets**) Performance on Tiny-ImageNet-C. Average represents the accuracy across all corruption levels, with corruption severity ranging from 1 to 5.

T-span	2	5	10	50	100
GVP CIFAR-10	30.54	90.23	93.26	97.50	97.55
GVP Tiny ImageNet	5.06	48.95	53.24	71.05	71.18
ICFM CIFAR-10	31.18	92.35	97.02	97.60	97.61
ICFM Tiny ImageNet	6.01	60.06	65.16	71.20	71.58

Table 10. Comparison of Performance Over Varying Time Spans.

reverse generation results on the TinyImageNet dataset after pre-training and fine-tuning, respectively. Finally, Fig. 4 demonstrates the reverse process before and after applying fog corruption to the images.

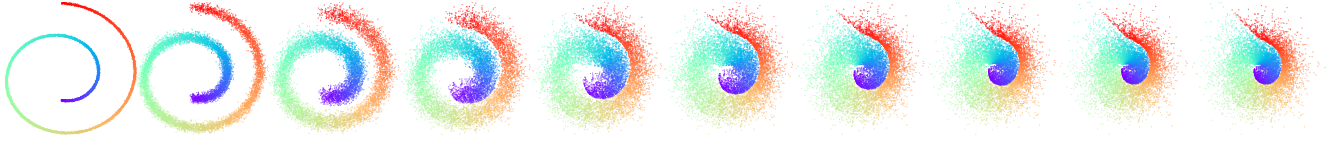


Figure 1. Reverse Generation Process on the Swiss Roll Dataset. Each color represents a different class. After diffusion, samples from the same class become more clustered, while the previously unoccupied white space, corresponding to out-of-class regions, is pushed outward.

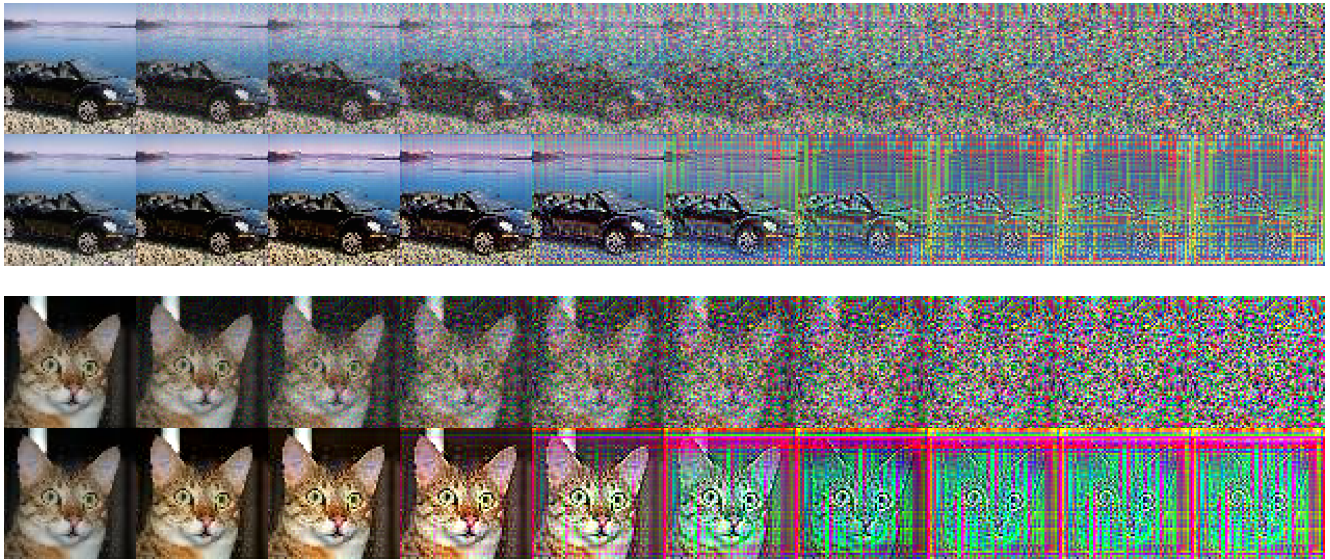


Figure 2. Reverse Generation Process on the TinyImageNet val set. The first row represents the fully pretrained reverse generative process, the second row shows the reverse generative process after extensive fine-tuning.

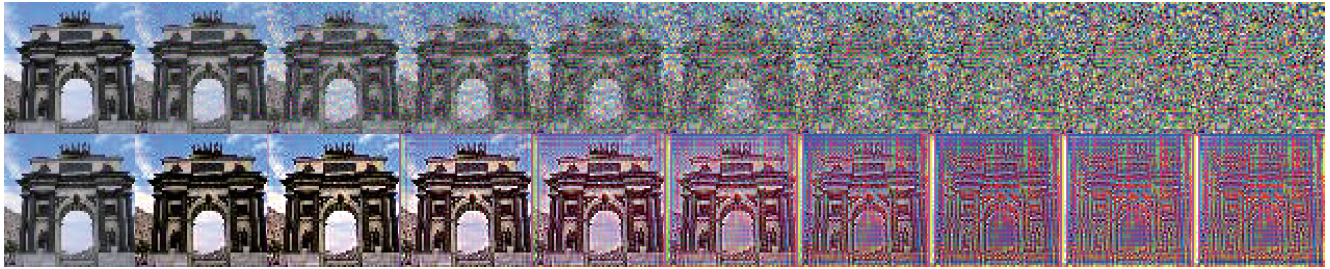


Figure 3. Reverse Generation Process on the TinyImageNet train set. The first row represents the fully pretrained reverse generative process, the second row shows the reverse generative process after extensive fine-tuning.

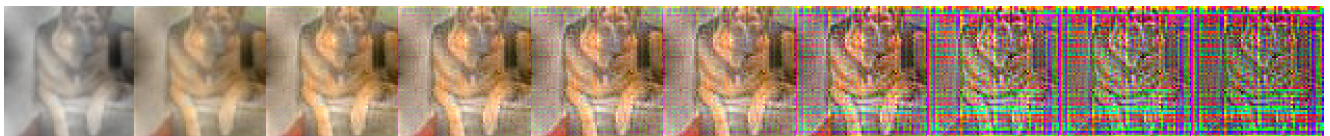
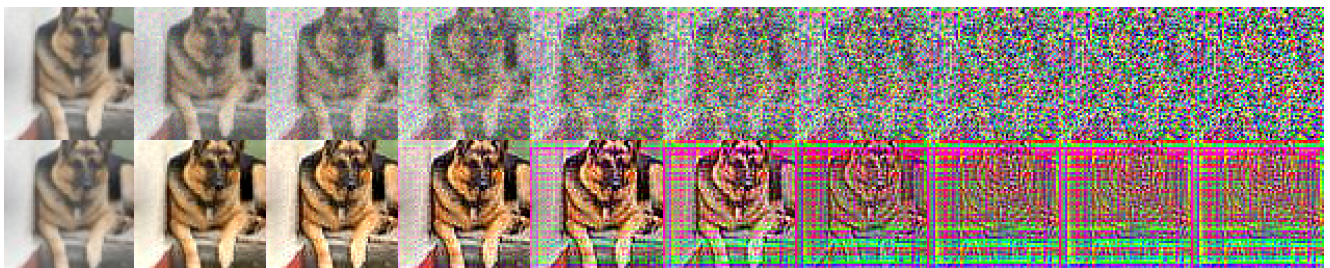


Figure 4. Reverse Generation Process on the TinyImageNet-C Dataset. The first row represents the fully pretrained reverse generative process, the second row shows the reverse process after extensive fine-tuning, and the third row illustrates the reverse generative process under fog corruption.