B-DENSE: BRANCHING FOR DENSE ENSEMBLE NETWORK SUPERVISION EFFICIENCY

Cherish Puniani, Tushar Kumar, Arnav Bendre, Gaurav Kumar, Shree Singhi
Indian Institute of Technology, Roorkee

Roorkee, Uttarakhand, India
cherish_p@me.iitr.ac.in, tushar_k@cy.iitr.ac.in,

arnav_b@hre.iitr.ac.in, gaurav_k@mfs.iitr.ac.in,
shree_s@mfs.iitr.ac.in

ABSTRACT

Inspired by non-equilibrium thermodynamics, diffusion models have achieved state-of-the-art performance in generative modeling. However, their iterative sampling nature results in high inference latency, which is typically required to maintain image quality. While recent efforts in distillation techniques have improved sample quality with fewer steps, they discard intermediate trajectory steps. By discarding intermediate trajectory steps, these methods lose structural information, resulting in significant discretization errors. To mitigate this issue, we propose a novel framework, **B-DENSE**, that leverages multi-branch trajectory alignment. We train the student model using branches that simultaneously map to the entire sequence of the teacher's target timesteps. We modify the student architecture to output K−fold expanded channels. Each channel subset corresponds to a specific branch representing a discrete intermediate step in the teacher's trajectory. By enforcing intermediate trajectory alignment, the student model learns to navigate the solution space from the earliest stages of training, leading to better image generation quality than the baseline distillation frameworks.

1 INTRODUCTION

Diffusion models Ho et al. (2020a); Sohl-Dickstein et al. (2015)have recently come to dominate image synthesis, surpassing previous paradigms such as Generative Adversarial NetworksHeusel et al. (2017) and Variational Autoencoders Kingma & Welling (2013). This is largely attributed to a stable training objective and robust framework that learns to reverse a gradual noise injection process by iterative sampling. By transforming a standard normal prior into high-fidelity data through a sequence of learned Gaussian transitions, these models achieve exceptional sample quality and mode coverage. However, these advantages come at a high computational cost; achieving high-quality results typically requires hundreds or thousands of iterative denoising steps, rendering inference slow and compute-intensive. Numerous studies focus on mitigating this bottleneck by either designing faster ODE solvers that speed up the inference process without changing the diffusion model's parameters (DDIM, Euler, Heun, DPM-Solver++), or by distilling slow but high-quality models into faster models that take far fewer steps. Many distillation methods (Salimans & Ho (2022a); Zhou et al. (2024); Kim et al. (2023) Kim et al. (2024) Xiang et al. (2025))supervise the student only at a sparse set of timestamps, matching the teacher at the endpoints of a collapsed interval. This sparse supervision discards critical trajectory geometry, leading to significant discretization errors and increased sensitivity to step reductions.

We propose B-DENSE, a novel distillation framework that addresses these limitations by modifying the architecture to explicitly align the student with the teacher's entire denoising trajectory over each collapsed interval without any significant computation overhead. Our method modifies the student architecture to output $K \cdot C$ channels, organized into K parallel branches, where each branch predicts the teacher's denoised state at a distinct intermediate timestamp within a teacher interval. During training, the teacher generates the full sequence of intermediate states, and the student is supervised with a multi-branch loss that enforces agreement at all intermediate points, rather than only at the

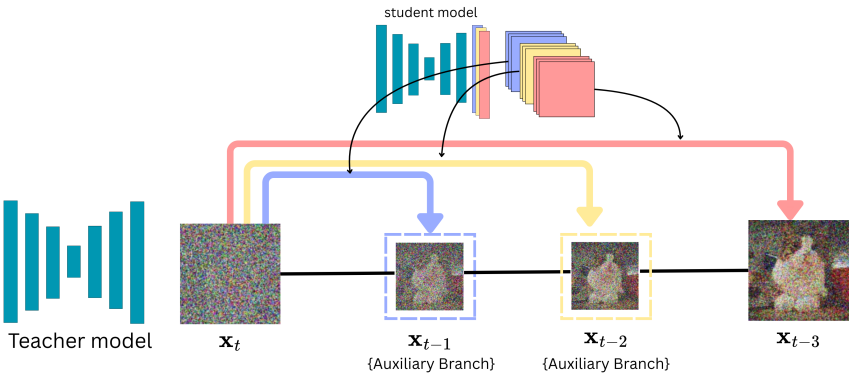

Figure 1: This visualization illustrates a single iteration of the proposed B-DENSE method. Here, x_t represents a noisy image at a random time step t. At the same time, x_{t-3} is output directly mapped by the standard distillation method, and x_{t-1} and x_{t-2} are consecutive intermediate denoised outputs that will be skipped by the standard distillation method. The blue, yellow, and red channels indicate branched channels for $K = 3$. Blue, yellow, and red arrows illustrate the mapping of each three-channel configuration to the final output and to an intermediate step used to calculate the reconstruction loss for each corresponding branch.

interval endpoints. B-DENSE recovers the fine-grained teacher updates typically discarded during distillation. By treating these branches as an ensemble of local predictors, the model approximates the full trajectory more accurately. This dense alignment minimizes discretization errors.

Across experiments, B-DENSE demonstrates better FID scores, with notable improvements in low-step sampling regimes. Importantly, the method incurs minimal additional overhead; the multi-branch output requires $(K - 1)$ additional channel dimensions, while teacher target generation remains the dominant cost, making the approach broadly compatible with existing U-Net based diffusion architectures. B-DENSE provides a computationally efficient mechanism for dense trajectory supervision in diffusion distillation. We provide a theoretical interpretation of the method as a piecewise quadrature approximation of the probability-flow ODE, which helps explain its reduced discretization errors and better generation quality when aggressively reducing the number of sampling steps. Through empirical evaluation, we found that B-DENSE achieves better sample quality than established baselines, particularly in low-step regimes, while maintaining the same inference-time complexity.

2 Background

2.1 Diffusion framework

Diffusion models are generative models inspired by non-equilibrium statistical physics Sohl-Dickstein et al. (2015). They progressively inject Gaussian noise into the data, destroying its structure through an iterative forward diffusion process. The models then learn a reverse process that reconstructs the original image from pure noise. Theoretically, these models can be described using stochastic differential equations (SDEs), where the forward process is defined as:

$$dx = f(x, t)dt + g(t)dw_t$$

In this context, $f(\cdot, t)$ and $g(\cdot)$ denote the drift and diffusion coefficients, respectively, while w_t represents the Wiener process. The generative model approximates the intractable reverse process using a reverse-time SDE to sample data from noise, defined as follows:

$$dx = [f(x, t) - g^2(t)\nabla_x \log p_t(x)]dt + g(t)d\bar{w}_t$$

Here $\nabla_x \log p_t(x)$ is known as the score Song & Ermon (2019), which represents the gradient of the log-density of the noisy data at each timestep t

To implement the reverse process, we parameterize a neural network $\epsilon_\theta(x, t)$ referred to as the noise-prediction model to estimate the score function. The model is trained by optimizing a variational

lower bound, which simplifies to a weighted mean squared error (MSE) between the actual noise ϵ and the predicted noise:

$$L_{simple} = E_{x_0,\epsilon,t}[\|\epsilon - \epsilon_\theta(x_t, t)\|^2]$$

While the SDE framework is mathematically elegant, it presents several significant issues, such as slow sampling speed and stochasticity, because it requires many small discretization steps. Every time we sample from the same starting noise x_T, we get a different result. While diversity is usually beneficial, it makes precise latent-space manipulations impossible. To eliminate this stochasticity, the diffusion process can be reformulated as a deterministic Probability Flow Ordinary Differential Equation (ODE) Song et al. (2021), also often called probability flow ODE, is defined as

$$dx = \left[f(x,t) - \frac{1}{2} g(t)^2 \nabla_x \log p_t(x) \right] dt$$

The probability flow ODE transforms the stochastic process into a deterministic mapping. This ensures that a specific initial noise vector x_T always collapses to the same image x_0. Consequently, the generative trajectory can be subsampled, enabling high-fidelity sample synthesis in as few as 10–50 steps rather than the canonical 1000.

2.2 Distillation and Related Work

Samplers are numerical algorithms designed to compute the reverse-time generation process, typically by formulating it as a PF-ODE that maps noise back to data. Because the standard denoising process is sequential and requires many iterations to resolve the continuous trajectory, solvers help speed up diffusion by employing more efficient discretization techniques (such as higher-order approximations like Heun or DPM-Solver) to estimate the correct path with fewer steps. This more efficient discretization reduces the NFE required to generate a high-quality image, thereby lowering the sampling latency without the need to retrain the underlying model.

Distillation of diffusion models is a training approach aimed at reducing the high computational costs associated with the standard iterative sampling process. This process usually involves hundreds or thousands of steps to generate a single image. In this framework, a 'student' model is trained to replicate the behavior of a pre-trained 'teacher' model but is optimized to follow the denoising trajectory using significantly fewer timesteps.

Progressive Distillation Salimans & Ho (2022b) accelerates diffusion model sampling by iteratively training a student model to compress two consecutive deterministic steps of a teacher model, typically using a DDIM sampler, into a single step. To ensure stability as the number of steps decreases, the student model is often reparameterized to predict the clean data x or the velocity v directly rather than the noise ϵ, and is trained using a Truncated SNR loss that maintains training signals even at zero signal-to-noise ratios. The student learns to predict a precise target value derived by analytically inverting the diffusion process to match the teacher's trajectory, allowing the sampling process to be recursively reduced from thousands of steps to as few as four with minimal loss in generation quality.

Simple and Fast Distillation (SFD) Zhou et al. (2024) streamlines diffusion model acceleration by addressing the 'step mismatch' inefficiency, grounded in the theoretical observation that optimizing the gradient field at specific timestamps smoothly enhances the score function at neighbouring times, rendering dense fine-tuning unnecessary. Unlike standard 'local' distillation methods that minimize error at each step independently, SFD adopts a 'global' trajectory-matching framework in which the student model generates and imitates the complete sampling trajectory of a high-order teacher (e.g., DPM-Solver++), enabling it to correct accumulated discretization errors dynamically. By restricting optimization to the timestamps used during inference and using a simplified L1 loss, SFD eliminates the need for complex objectives while enabling variable-step sampling via step-conditioning .

Consistency Models Song et al. (2023) accelerate the generative process by learning a function that maps any point along the probability flow ODE trajectory directly to its original clean data, enabling single-step generation. Unlike standard diffusion models that require iterative denoising, this approach enforces a self-consistency property, ensuring that widely separated points on the same trajectory are mapped to the exact same initial data point x_0. This formulation allows the model to generate high-quality samples from pure noise in a single forward pass, while also supporting multi-step sampling that trades increased computation for improved sample fidelity.

Algorithm 1 PD

Require: Trained teacher model $\hat{x}_\eta(\mathbf{z}_t)$
Require: Data set $\mathcal{D}$
Require: Loss weight function $w()$
Require: Teacher steps to approximate N

for K iterations **do**
  $\phi \leftarrow \eta$ ▷ Init student from teacher
  while not converged **do**
    $\mathbf{x} \sim \mathcal{D}$
    $\epsilon \sim \mathcal{N}(0, I)$
    $t \sim [1, T]$
    $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \epsilon$
    # 2 steps of DDIM with teacher
    $\hat{x}_{t-2} \rightarrow$ latent
    $\lambda_t = \log[\alpha_t^2/\sigma_t^2]$
    $\mathcal{L}_\Phi = w(\lambda_t)\|\hat{x}_{t-2} - \tilde{x}_\phi(\mathbf{z}_t)\|_2^2$
    $\phi \leftarrow \phi - \gamma \nabla_\phi \mathcal{L}_\phi$
  end while
  $\eta \leftarrow \phi$ ▷ Student becomes teacher
  $T \leftarrow T/N$
end for

Algorithm 2 PD + B-DENSE

Require: Trained teacher model $\hat{x}_\eta(\mathbf{z}_t)$
Require: Data set $\mathcal{D}$
Require: Loss weight function $w()$
Require: Teacher steps to approximate N
Require: Weights $\{\lambda_k\}_{k=0}^{K-1}$
for K iterations **do**
  $\phi \leftarrow \eta$ ▷ Init student from teacher
  while not converged **do**
    $\mathbf{x} \sim \mathcal{D}$
    $\epsilon \sim \mathcal{N}(0, I)$
    $t \sim [1, T]$
    $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \epsilon$
    # K steps of DDIM with teacher
    Latents $\rightarrow [\hat{x}_t, \ldots, \hat{x}_{t-K-1}]$
    Student outputs $\rightarrow [\tilde{x}_t, \ldots, \tilde{x}_{t-K-1}]$
    $\mathcal{L}_\phi = \sum_{k=0}^{K-1} \lambda_k \cdot \|\hat{x}_{t-k} - \tilde{x}_{t-k}(\mathbf{z}_t)\|_2^2$
    $\phi \leftarrow \phi - \gamma \nabla_\phi \mathcal{L}_\phi$
  end while
  $\eta \leftarrow \phi$ ▷ Student becomes teacher
  $T \leftarrow T/N$
end for

3 METHODOLOGY

3.1 B-DENSE

Most Diffusion Distillation frameworks only train on a subset of the teacher's outputs and discard intermediate trajectory steps. Consequently, these frameworks miss valuable supervision signals, leading to suboptimal convergence. B-DENSE leverages these intermediate teacher outputs, which are already available, to provide dense supervision to the student model. This reduces discretization errors and improves the quality of the final image.

B-DENSE can be interpreted as a more accurate instance of numerical integration over the generative trajectory defined by the teacher diffusion model. Instead of viewing denoising steps as discrete transitions, we treat them as evaluations of a continuous vector field $f(x, t)$ derived from the reverse-time probability flow ODE. By using the K sub-intervals of the teacher's decomposed sampling interval $[t_{n+1}, t_n]$ and training the student to match the teacher's predictions at each sub-interval boundary, the student approximates the total integral of f via a structured piecewise quadrature.

This method is a foundational architectural design that can be integrated into various distillation pipelines to ensure the student model navigates the solution space effectively from the earliest stages of training. We begin B-DENSE by using a pretrained diffusion model as the teacher rather than training it from scratch. Then, the student model is initialized as a copy of the teacher, with an important architectural change: to approximate K teacher steps, the student's final layer weights are initialized by repeating the teacher's weights K times, to generate $K \cdot C$ channels instead of C, where C represents the original channel dimension. The output tensor is partitioned into K successive groups, each serving as an auxiliary "branch" that predicts a specific intermediate state.

As in standard training, we sample data and add noise; however, the key difference lies in the training target, which is obtained by running the teacher model for K inference steps and storing the full sequence of intermediate denoised images. To modulate the supervision signal strength across the trajectory, we assign weights $\{\lambda_i\}_{i=0}^{k-1}$ to the reconstruction loss of each corresponding branch, i.e. the new combined objective loss becomes $\lambda_0 \mathcal{L}(x_0, \hat{x}_0) + \lambda_1 \mathcal{L}(x_1, \hat{x}_1) + \cdots + \lambda_{k-1} \mathcal{L}(x_{k-1}, \hat{x}_{k-1})$

where $\mathcal{L}$ is the objective loss of the underlying distillation algorithm, x_i is the ith target output generated by teacher and $\hat{x}_i$ is the ith student prediction. A comprehensive derivation regarding the selection of these weights is provided in the Appendix subsection A.1. Inference is performed using the weights corresponding to the final channel dimensions, which learns to map to the teacher's interval endpoint.

Since the primary bottlenecks, teacher target generation and student gradient computation, remain unchanged, the multi-branch expansion is nearly cost-free. The architectural expansion to a multi-channel output ($K \cdot C$) is confined to the final layer, requiring only $K - 1$ additional convolutional filters. This introduces a negligible computational overhead, increasing total FLOPs in the order of $\sim 0.01\%$ relative to the standard U-Net or Transformer backbone. Consequently, B-DENSE offers a high-efficiency pathway to increase training efficiency with minimal cost, making it broadly applicable across diverse distillation frameworks. We apply the B-DENSE framework on the Progressive Distillation, as shown in Algorithms 1 and 2, and the SFD algorithm, as shown in Algorithms 3 and 4.

3.2 Theoretical Analysis

The effectiveness of B-DENSE can be formally justified by viewing the sampling process through the lens of the Probability Flow ODE.

Theoretical Foundation The generation process in diffusion models is fundamentally defined as solving the reverse-time Probability Flow ODE. Under this formulation, the data trajectory is governed by the following ODE:

$$dx = \epsilon_\theta(x, t)dt \tag{1}$$

where $x(t)$ represents the data state and ϵ_θ denotes the learned vector field. To transition from a noisy state at time t_{n+1} to a cleaner state at t_n, the ideal solver must compute the integral of this vector field:

$$x(t_n) = x(t_{n+1}) + \int_{t_{n+1}}^{t_n} \epsilon_\theta(x(u), u)du \tag{2}$$

The accuracy of the generative process depends entirely on the fidelity with which this integral is approximated during sampling.

The Problem with Sparse Supervision (Baseline Distillation) Standard distillation techniques employ what we classify as Sparse Supervision. In these frameworks, the student model S_ϕ is trained to predict the integral's endpoint $x(t_{n-k})$ directly from the starting point $x(t_n)$ in a single step, minimizing the loss:

$$\mathcal{L}_{\text{sparse}} = \left\| S_\phi(x_{t_n}) - x_{t_{n-k}} \right\|^2 \tag{3}$$

This formulation treats the integral term $\int_{t_{n+1}}^{t_n} \epsilon_\theta du$ as a black box. The student attempts to learn the result of the integral without explicit constraints on the path taken.

As noted in SFD Zhou et al. (2024), diffusion trajectories often exhibit high curvature. In these regions, there exists a manifold of incorrect paths that S_ϕ could hallucinate. This lack of path constraints leads to suboptimal results and high discretization errors, as the student is prone to wandering off the true trajectory manifold during the early stages of training.

B-DENSE as Dense Trajectory Supervision To address the limitations of sparse supervision, we introduce Dense Trajectory Supervision, motivated by the Ensemble Parallel Direction (EPD) solver Zhu et al. (2025). EPD solver demonstrates that utilizing intermediate gradients at times $\tau_k \in [t_{n+1}, t_n]$ significantly reduces the local truncation error of the integral approximation. While EPD calculates these intermediate values in parallel during inference to improve the integration step:

$$x_{t_n} = x_{t_{n+1}} + h \sum_{k=1}^{K} \lambda_k \epsilon_\theta(x_{\tau_k}, \tau_k) \tag{4}$$

B-DENSE forces the student to learn these states during training.

Formulation: We discretize the distillation interval $[t_{n+1}, t_n]$ into K intermediate control points $\tau_0, \tau_1, \ldots, \tau_{K-1}$, where $\tau_K = t_n$ which are teacher model's intermediate steps. The student model architecture is modified to output a set of predictions corresponding to these time steps:

$$S_\phi(x_{t_{n+1}}) = \{\hat{x}_{\tau_0}, \hat{x}_{\tau_2}, ..., \hat{x}_{\tau_{K-1}}\} \tag{5}$$

This mapping is performed while the shared backbone is conditioned on t_{n+1}. By supervising each branch k against its corresponding teacher target at τ_k, the network is forced to internalize the local velocity of the probability-flow ODE, effectively learning the sub-integrals of the vector field. Consequently, we define the Dense Loss Function as:

$$\mathcal{L}_{\text{branch}} = \sum_{k=0}^{K-1} w_k \cdot ||\hat{x}_{\tau_k} - x_{\text{teacher}}(\tau_k)||^2 \tag{6}$$

B-DENSE functions as a pinned numerical integrator. By enforcing the loss at each intermediate τ_k, we constrain the student's learned vector field to satisfy the ODE flow property at multiple points simultaneously If we view the student's prediction for a branch k as an approximation of the sub-integral:

$$\hat{x}_{\tau_k} \approx x(t_{n+1}) + \int_{t_{n+1}}^{\tau_k} \epsilon_{\text{teacher}}(x(u), u)du \tag{7}$$

Our method ensures that the student does not just learn the total integral, i.e, endpoint matching, but implicitly learns the sub-integrals. This constrains the parameter search space of ϕ, effectively forcing the student to align with the teacher's denoising trajectory.

Algorithm 3 SFD

Require: Student ${}^{\phi}$, Teacher η
Require: ODE Solver

while not converged **do**
  Sample ${}_{n+1} \sim \mathcal{N}({}_0; t_{n+1}^2)$
  for $n = N-1$ **to** 0 **do**
    ${}_n^{\phi} = \text{Euler}({}_{n+1}, t_{n+1}, t_n, 1; \phi)$
    $\tilde{}_n = \text{Solver}({}_{n+1}, t_{n+1}, t_n, K; \eta)_K$
    $\mathcal{L}\phi = d({}_n^{\phi}, \tilde{}_n)$
    $\phi \leftarrow \phi - \eta \nabla_{\phi} \mathcal{L}\phi$
    ${}_n = \text{detach}({}_n^{\phi})$
  end for
end while

Algorithm 4 SFD + B-DENSE

Require: Branched model ϕ_{aug}, Teacher η
Require: Weights $\{\lambda_k\}_{k=0}^{K-1}$

while not converged **do**
  Sample $\mathbf{x}_{n+1} \sim \mathcal{N}(\mathbf{x}_0; t_{n+1}^2 \mathbf{I})$
  for $n = N-1$ **to** 0 **do**
    $\mathbf{X}_n^{\phi} = \text{Euler}(\mathbf{x}_{n+1}, t_{n+1}, t_n, 1; \phi_{aug})$
    $\tilde{\mathbf{X}}_n = \text{Solver}(\mathbf{x}_{n+1}, t_{n+1}, t_n, K; \eta)$
    $\mathcal{L}\phi = \sum_{k=0}^{K-1} \lambda_k \cdot d(\mathbf{X}_{n,k}^{\phi}, \tilde{\mathbf{X}}_{n,k})$
    $\phi_{aug} \leftarrow \phi_{aug} - \eta \nabla_{\phi_{aug}} \mathcal{L}\phi$
    $\mathbf{x}_n = \text{detach}(\mathbf{X}_{n,K}^{\phi})$
  end for
end while

4 Experiments

4.1 Experiment Settings

Experimental Settings for PD. The experiments utilize the pretrained model and weights from Ho et al. (2020b) as described in the original DDPM paper. The CIFAR-10 dataset, which contains $60,000$ 32×32 colour images across 10 classes, is used for all experiments. Following standard protocols, we employ ϵ-parameterization and a linear noise schedule to ensure consistency with recent literature. The teacher model initially performs 1024 steps, which are distilled to 128 steps. As the original authors set $K = 2$, the same parameterization is used in B-DENSE. Each distillation iteration involves 50k parameter updates, a batch size of 128, and the AdamW optimizer with a learning rate of 2×10^{-4} . The model's performance proved robust to variations in branch weights; thus, a uniform weighting strategy was used. Distillation experiments are conducted with NVIDIA L4 GPU and the time taken to distill a model is $\sim$ 4-5 hours.

Experimental Settings for SFD. Pre-trained diffusion models provided by EDM Karras et al. (2022) were used. We evaluate results on CIFAR10 32×32 Krizhevsky & Hinton (2009) and ImageNet 64×64 Russakovsky et al. (2015). All training hyperparameters were set to the same values as described by the original authors. Consequently, K was set to 4. The values of $\{\lambda_i\}_{i=0}^{k-1}$ were set to $[0.017, 0.056, 0.191, 0.651]$ for CIFAR10 and $[0.014, 0.056, 0.223, 0.892]$ for ImageNet. Distillation on NFE 2 was done using NVIDIA A100 GPU and the time taken to distill on CIFAR-10 was $\sim$ 44 minutes and ImageNet was $\sim$ 3 hours.

Table 1: Comparison of the FID of both frameworks on CIFAR-10 with various NFE

Methodology	**512**	**256**	**128**
ProgressiveDistillation (Baseline)	11.96	21.52	39.66
B-DENSE (Ours)	**8.92**	**12.04**	**20.81**

4.2 RESULTS

Results on PD: Table 1 shows that our technique achieves better performance than the baseline framework. Figure 2 shows that, as distillation progresses, the FID between the two models diverges. This indicates that our method provides richer supervision by leveraging the discarded intermediate timestep at each iteration, thereby reducing discretization error in the sampling trajectory. Our method achieves this with almost no training time increment.

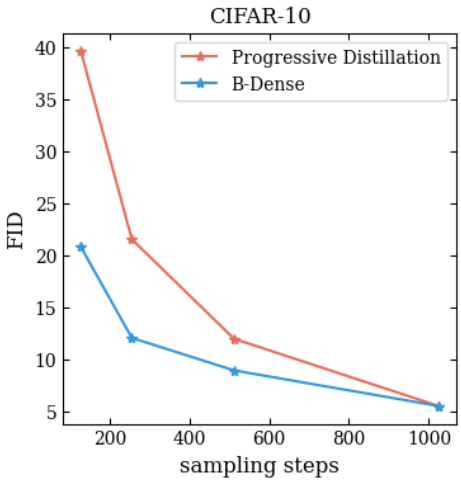

Figure 2: Results on calculating FID for 50k images

Results on SFD: The integration of B-DENSE into the SFD framework yields consistent improvements in image fidelity across different sampling budgets. By providing dense trajectory supervision, the model bridges the gap between the teacher's continuous path and the student's discretized steps.

As shown in 2, B-DENSE significantly improves performance in the ultra-low-step regime (NFE 2) on ImageNet. We speculate that the suboptimal performance for B-DENSE for NFE ≥ 3 on ImageNet is attributed to the fact that the branch weights were chosen according to the CIFAR 10 dataset, as discussed in subsection A.1. Our framework achieves an FID of 4.40, a notable improvement over the baseline's 4.53, proving that intermediate branch supervision helps the model maintain structural integrity even with minimal steps. Across all experiments, we observed virtually the same wall clock time and memory usage with and without B-DENSE.

Table 2: FID comparison on CIFAR-10

Methodology	**NFE 2**	**NFE 3**	**NFE 4**	**NFE 5**
SFD (Baseline)	4.53	3.58	3.24	3.06
B-DENSE (Ours)	**4.40**	**3.52**	**3.21**	**3.01**

Table 3: FID comparison on ImageNet 64x64

Methodology	**NFE 2**	**NFE 3**	**NFE 4**	**NFE 5**
SFD (Baseline)	10.25	**6.35**	**4.99**	**4.33**
B-DENSE (Ours)	**9.57**	6.54	5.97	5.91

The scalability of B-DENSE is further evaluated on the ImageNet dataset. As the image distribution becomes more complex and diverse, dense supervision from auxiliary branches plays a key role in maintaining fine-grained visual structure and stable denoising behavior. Results are summarized in 3.

The results indicate a clear trend: by supervising the student on the trajectory path itself rather than just the final endpoint, B-DENSE achieves a more robust approximation of the Probability Flow ODE, reducing the cumulative error that we usually encounter in low-step regimes.

Figure 3: Comparison of results of B-DENSE (right) and SFD (left)

5 Discussion

The empirical success of B-DENSE across multiple frameworks (Progressive Distillation and SFD) suggests that the missing link in current distillation research is not necessarily the capacity of the student model, but the density of the supervision signal.

Mitigating Discretization Error. Standard distillation forces the student to learn a shortcut between two distant points on the PF-ODE. In a high-curvature vector field, this linear approximation leads to discretization error, which compounds over sampling steps. By enforcing alignment at K intermediate points, B-DENSE effectively constrains the student to follow the teacher's curved trajectory. This reduces the variance of the update even when the number of function evaluations is small.

Computational Efficiency and Scalability. A primary advantage of B-DENSE is its free lunch nature. Since the branched outputs share the entire backbone of the U-NET and are only active during training, the method provides a significant reduction in the discretization errors with near-zero impact on forward and backward pass latency. This makes it particularly attractive for scaling distillation to high-resolution models like Stable Diffusion, where training costs are otherwise prohibitive.

Limitations and future work. While B-DENSE significantly improves sampling quality, it remains tethered to the teacher's trajectory quality; any inherent artifacts in the teacher's sampling path are faithfully replicated by the student. Furthermore, the current dependence on framework-specific branch weights provides a natural entry point for developing more adaptive, self-tuning schedules. Future work will focus on transforming these branch weights into learnable parameters, allowing the model to dynamically balance structural alignment and perceptual refinement. We also aim to validate this framework on Latent Diffusion Models and scale it toward Video and 3D generation, where dense trajectory consistency is paramount.

References

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*, pp. 6626–6637, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020a.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020b. URL https://arxiv.org/abs/2006.11239.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *arXiv preprint arXiv:2310.02279*, 2023.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion, 2024. URL https://arxiv.org/abs/2310.02279.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

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 Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115 (3):211–252, 2015.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022a.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models, 2022b. URL https://arxiv.org/abs/2202.00512.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021. URL https://arxiv.org/abs/2011.13456.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models, 2023. URL https://arxiv.org/abs/2303.01469.

Qianlong Xiang, Miao Zhang, Yuzhang Shang, Jianlong Wu, Yan Yan, and Liqiang Nie. Dkdm: Data-free knowledge distillation for diffusion models with any architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2955–2965, June 2025.

Zhenyu Zhou, Defang Chen, Can Wang, Chun Chen, and Siwei Lyu. Simple and fast distillation of diffusion models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=Ao0FiZqrXa.

Beier Zhu, Ruoyu Wang, Tong Zhao, Hanwang Zhang, and Chi Zhang. Distilling parallel gradients for fast ode solvers of diffusion models, 2025. URL https://arxiv.org/abs/2507.14797.

Table 4: Top-5 weight configurations found via Optuna parameter search. The weights follow the form $\lambda_i = \exp(a \cdot i + b)$. We report FID scores at NFE 2 for both ImageNet 64×64 and CIFAR-10 32×32. All configurations outperform the SFD baseline.

		FID (NFE 2)	
a	**b**	**ImageNet 64**	**CIFAR-10**
1.386	-4.274	**9.57**	**4.40**
1.337	-4.195	9.72	4.40
1.230	-4.085	9.84	4.40
1.278	-4.137	9.72	4.41
1.277	-4.207	9.64	4.41
SFD baseline		10.57	4.53

A Appendix

A.1 Choosing Optimal Branch Weights

To determine the optimal contribution of each branch to the total loss, we treated the weighting coefficients $\{\lambda_i\}_{i=0}^{K-1}$ as hyperparameters. Conducting a grid search over four independent continuous variables is computationally expensive and prone to instability. Instead, we assumed that the supervision signal should monotonically increase as the trajectory approaches the final clean data state. This reflects the intuition that errors in the final steps of generation have a more direct impact on perceptual quality than errors in the noisy, early stages, due to the lower Signal-to-Noise-Ratio (SNR).

Based on this assumption, we hypothesized that the optimal weight schedule follows a geometric progression. We parametrized the weights using a linear function in the log-space:

$$\lambda_i = \exp(a \cdot i + b) \tag{8}$$

where $i \in \{0, \ldots, K-1\}$. This formulation reduces the search space to two parameters: a, which controls the rate of growth (slope), and b, which determines the initial magnitude (intercept).

Search Strategy and Constraints. Performing a comprehensive hyperparameter search for Progressive Distillation (PD) is computationally prohibitive, as the pipeline requires multiple sequential training sessions to iteratively halve the step count. To overcome this limitation, we conducted the search using the Simple and Fast Distillation (SFD) framework. We specifically targeted the CIFAR-10 dataset in the NFE 2 (Number of Function Evaluations) regime. This configuration provided the fastest training and evaluation loop, enabling us to efficiently explore the parameter space. We reasoned that weights optimized for the most constrained sampling regime (2 steps) on a standard dataset would transfer effectively to other resolutions and step counts.

We utilized the Optuna framework to search for a and b, ensuring numerical stability by constraining $b \in [-7, -2]$. We then validated the transferability of the optimal weights found on CIFAR-10 by applying them to ImageNet 64×64.

Results. The search identified a distinct 'valley' of optimal configurations where the weights follow a steep exponential curve (high a). This confirms our initial assumption: assigning significantly higher importance to the later branches (closer to x_0) yields better convergence. Table 4 presents the top-5 configurations. While B-DENSE provides the most significant gains at NFE 2, where discretization error is most acute, the dense constraints serve as a powerful regularizer that maintains structural integrity across the low-step regime. The results for B-DENSE on Imagenet for NFE$\geq$ 3 being suboptimal can be largely attributed to lack of hyperparameter search under compute limitations.

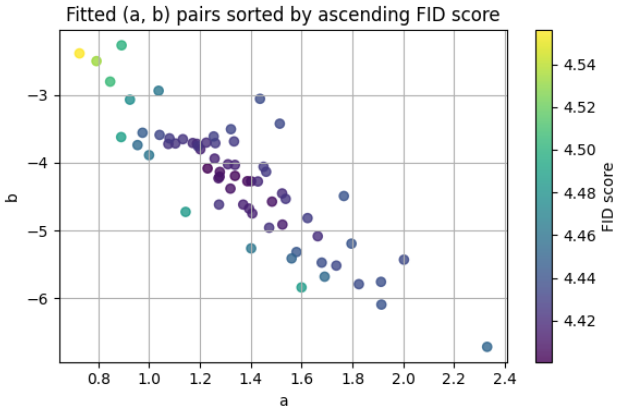

Figure 4: Fitted (a, b) parameter pairs sorted by ascending FID score. The color scale indicates the FID score (lower is better). The clustering of high-performing configurations (dark purple) demonstrates a strong preference for a specific relationship between the growth rate a and intercept b.