ByteDance | Seed

Expert Race: A Flexible Routing Strategy for Scaling Diffusion Transformer with Mixture of Experts

Yike Yuan[1,*], **Ziyu Wang**[1,2,*], **Zihao Huang**[1], **Defa Zhu**[1], **Xun Zhou**[1], **Jingyi Yu**[2,†], **Qiyang Min**[1,†]

[1]ByteDance Seed, [2]ShanghaiTech University

[*]Equal Contribution, [†]Corresponding Authors

arXiv:2503.16057v1 [cs.CV] 20 Mar 2025

Abstract

Diffusion models have emerged as mainstream framework in visual generation. Building upon this success, the integration of Mixture of Experts (MoE) methods has shown promise in enhancing model scalability and performance. In this paper, we introduce Race-DiT, a novel MoE model for diffusion transformers with a flexible routing strategy, Expert Race. By allowing tokens and experts to compete together and select the top candidates, the model learns to dynamically assign experts to critical tokens. Additionally, we propose per-layer regularization to address challenges in shallow layer learning, and router similarity loss to prevent mode collapse, ensuring better expert utilization. Extensive experiments on ImageNet validate the effectiveness of our approach, showcasing significant performance gains while promising scaling properties.

Date: March 21, 2025

Correspondence:
Qiyang Min at minqiyang@bytedance.com,
Jingyi Yu at yujingyi@shanghaitech.edu.cn

Contents

1 Introduction . . . 3

2 Related Work . . . 5
- 2.1 Mixture of Experts . . . 5
- 2.2 Multiple Experts in Diffusion . . . 5

3 Preliminaries . . . 5
- 3.1 Diffusion Models . . . 5
- 3.2 Mixture of Experts . . . 6
- 3.3 The Rationality of Using MoE in Diffusion Models . . . 6

4 Taming Diffusion Models with Expert Race . . . 6
- 4.1 General Routing Formulation . . . 6
- 4.2 Expert Race . . . 7

5 Load Balancing via Router Similarity Loss . . . 8
- 5.1 Mode Collapse in Balancing Loss. . . . 8
- 5.2 Router Similarity Loss. . . . 9

6 Per-Layer Regularization for Efficient Training . . . 9

7 Experiments . . . 10
- 7.1 Implementation Details . . . 10
- 7.2 Routing Strategy . . . 11
- 7.3 Gating Function . . . 11
- 7.4 Load Balance . . . 12
- 7.5 Core Components . . . 12
- 7.6 Scaling Law . . . 13
- 7.7 Extended Routing Strategy . . . 14
- 7.8 More Results on ImageNet $\mathbf{256 \times 256}$ . . . 15

8 Conclusion . . . 15

A Implemention of the Per-Layer Regularization . . . 19

B Analysis of Router Similarity Loss . . . 19

C Combination Usage . . . 21

D Additional Comparisons with DiT-MoE . . . 21

E Additional Image Generation Results . . . 22

1 Introduction

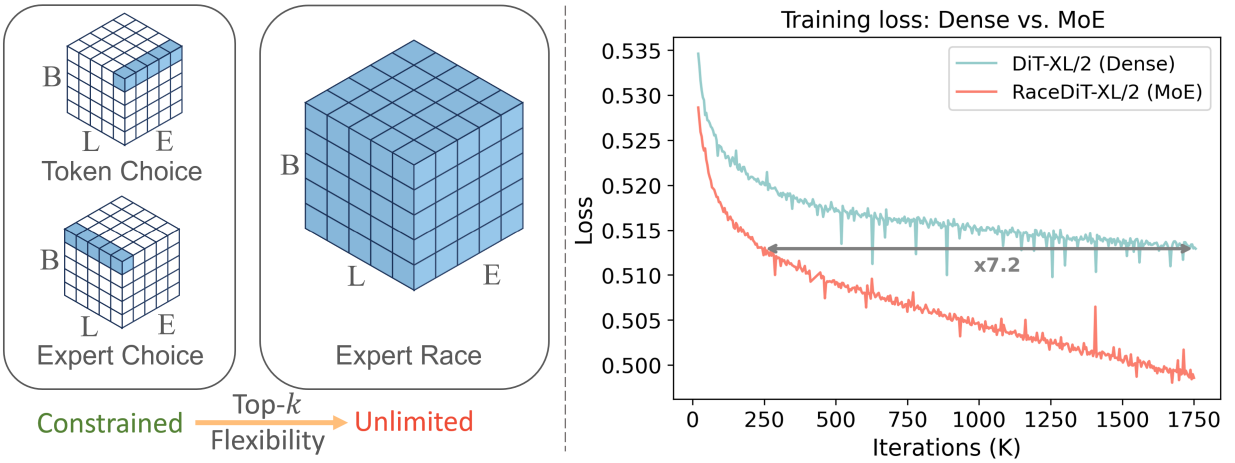


Figure 1 (Left) Our proposed **Expert Race** routing employs Top-k selection over full token-expert affinity logits, achieving the **highest flexibility** compared to prior methods like Token Choice and Expert Choice.
(Right) Training curve comparisons between DiT-XL [25] and ours. Our model, with **equal number of activated parameters**, achieves a **7.2× speedup** in iterations when reaching the same training loss.

Recent years have seen diffusion models earning considerable recognition within the realm of visual generation. They have exhibited outstanding performance in multiple facets such as image generation [7, 23, 28, 32, 33], video generation [3, 13], and 3D generation [2, 41]. Thus, diffusion models have solidified their position as a pivotal milestone in the field of visual generation studies. Mimicking the triumph of transformer-based large language models (LLMs), diffusion models have effectively transitioned from U-Net to DiT and its variants. This transition yielded not only comparable scaling properties but also an equally successful pursuit of larger models.

In the quest for larger models, the Mixture of Experts (MoE) approach, proven effective in scaling large language models (LLMs) [18], exhibits promising potential when incorporated into diffusion models. Essentially, MoE utilizes a routing module to assign tokens among experts (typically, a set of Feed-Forward Networks (FFN)) based on respective scores. This router module, pivotal to MoE's functionality, employs common strategies such as token-choice and expert-choice.

Meanwhile, we observe that the visual signals processed by diffusion models exhibit two distinct characteristics compared to those in LLMs. First, **visual information tends to have high spatial redundancy.** For instance, significant disparity in information density exists between the background and foreground regions, with the latter typically containing more critical details. Second, **denoising task complexity exhibits temporal variation across different timesteps.** Predicting noise at the beginning of the denoising process is substantially simpler than predicting noise towards the end, as later stages require finer detail reconstruction. These unique characteristics necessitate specialized routing strategies for visual diffusion models.

Consider these characteristics under the MoE, the presence of a routing module can adaptively allocate computational resources. By **assigning more experts to challenging tokens** and fewer to simpler ones, we can enhance model utilization efficiency. Previous strategies like expert-choice anticipated this, but their routing design limit the assignment flexibility to image spatial regions without considering temporal denoising timestep complexity.

In this paper, we introduce Race-DiT, a novel family of MoE models equipped with enhanced routing strategies, Expert Race. We find that simply increasing strategy flexibilities greatly boost the model's performance. Specifically, we conduct a "race" among tokens from different samples, timesteps, and experts, and select the top-k tokens from all. This method effectively filters redundant tokens and optimizes computational resource deployment by the MoE.

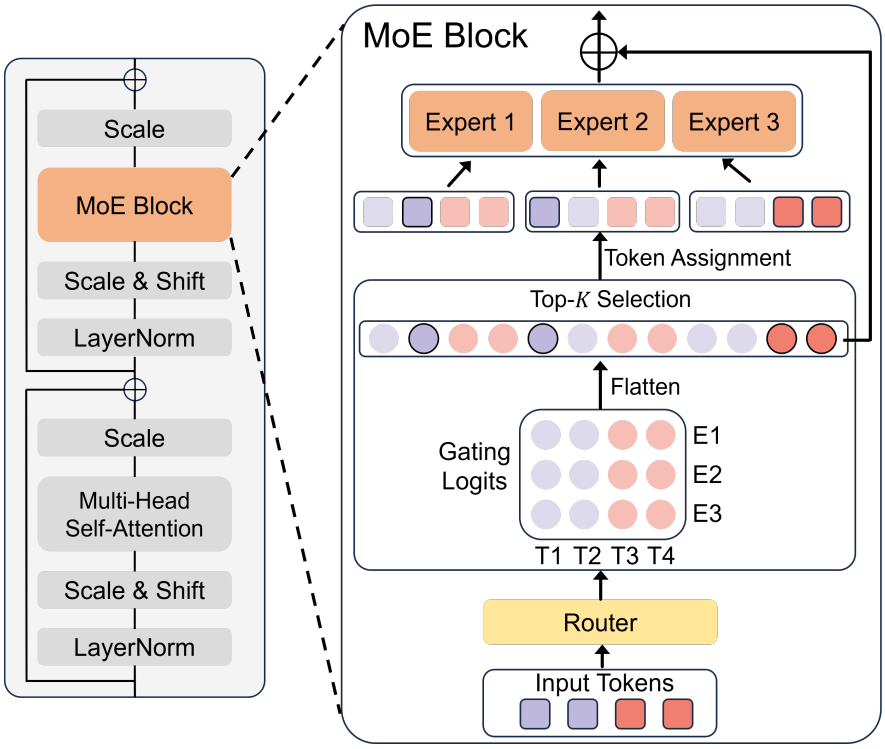


Figure 2 The Race-DiT Architecture. We replace the Multi-Layer Perceptron (MLP) with the MoE block, which consists of a Router and multiple Experts. In Race-DiT, the token assignment is done once for all. Each token can be assigned to any number of experts, and each expert can process any number of tokens (including zero).

Expert Race introduces a high degree of flexibility in token allocation within the MoE framework. However, there are several challenges when extending DiT to larger parameter scales using MoE. First, we observe that routing in the shallow layers of MoE struggles to learn the assignment, especially with high-noise inputs. We believe this is due to the weakening of the shallow components in the identity branch of the DiT framework. To address this, we propose an auxiliary loss function with layer-wise regularization to aid in learning. Second, considering the substantial expansion of the candidate space, to prevent the collapse of the allocation strategy, we extend the commonly used balance loss from single experts to combinations of experts. This extension is complemented by our router similarity loss, which ensures effective expert utilization by regulating pairwise expert selection patterns.

To validate the proposed method, we conducted experiments on ImageNet [5], performing detailed ablations on the proposed modules and investigating the scaling behaviors of multiple factors. Results show that our approach achieves significant improvements across multiple metrics compared to baseline methods.

In summary, our main contributions include

- Expert Race, a novel MoE routing strategy for diffusion transformers that supports high routing allocation flexibility in both spatial image regions and temporal denoising steps.
- Router similarity loss, a new objective that optimizes expert collaboration through router logits similarity, effectively maintaining workload equilibrium and diversifying expert combinations without compromising generation fidelity.
- Per-layer Regularization that ensures effective learning in the shallow layers of MoE models.
- Detailed MoE scaling analysis in terms of hidden split and expert expansion provides insights for extending this MoE model to diverse diffusion tasks.

2 Related Work

2.1 Mixture of Experts

Mixture of Experts (MoE) improves computational efficiency by activating only a subset of parameters at a time and forcing the other neurons to be zero. Typically, MoE is used to significantly scale up models beyond their current size leveraging the natural sparsity of activations. This technique has been widely applied in LLMs first [8, 20] and then extended to the vision domain [29]. The most commonly used routing strategy in MoE is token-choice, in which each token selects a subset of experts according to router scores. For its variants, THOR [46] employs a random strategy, BASELayer [21] addresses the linear assignment problem, HASHLayer [30] uses a hashing function, and MoNE [16] uses greedy top-k. All of these methods allocate fixed number of experts to each token. DYNMoE [12] and ReMoE [38] activates different number of experts for each token by replacing TopK with threshold and using additional regularization terms to control the total budget. Also, some auxiliary regularization terms are applied to constrain the model to activate experts uniformly [4, 36, 45]. Expert choice [44] has been proposed to avoid load imbalance without additional regularizations and enhance routing dynamics, but due to conflicts with mainstream causal attention, it is less commonly applied in large language models (LLMs).

2.2 Multiple Experts in Diffusion

Diffusion follows a multi-task learning framework that share the same model across different timesteps. Consequently, many studies have explored whether performance can be enhanced by disentangling tasks according to timesteps inside the model. Ernie [10] and e-diff [1] manually separate the denoising process into multiple stages and train different models to handle each stage. MEME [19] uses heterogeneous models and DTR [24] heuristically partitions along the channel dimension. DyDiT [42] introduce nested MLPs and channel masks to fit varing complexities across time and spatial dimensions. DiT-MoE [9], EC-DiT [35], and Raphael [39] have applied MoE architectures, learning to assign experts to tokens in an end-to-end manner. Compared with previous works, the methods proposed in this paper builds on the MoE but further enhancing its flexibility on dynamically allocate experts on all dimensions to unleash its potential.

3 Preliminaries

Before introducing our MoE design, we briefly review some preliminaries of diffusion models and mixture of experts.

3.1 Diffusion Models

Diffusion models [15] are a class of generative models that sample from a noise distribution and learn a gradual denoising process to generate clean data. It can be seen as an interpolation process between the data sample x_0 and the noise ϵ. A typical Gaussian diffusion models formulates the forward diffusion process as

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon \tag{1}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ is the Gaussian noise and $\bar{\alpha}_t$ is a monotonically decreasing function from 1 to 0. Diffusion models use the neural networks to estimate the reverse denoising process $p_\theta(x_{t-1}|x_t) = \mathcal{N}\left(\mu_\theta(x_t), \Sigma_\theta(x_t)\right)$. They are trained by minimizing the following objectives:

$$\min_\theta \mathbb{E}_{x_0,t,\epsilon}\left[\|\mathbf{y} - F_\theta(x_t; c, t)\|^2\right], \tag{2}$$

where t is the timestep which uniformly distributed between 0 to T , c is the condition information. e.g. class labels, image or text-prompt. The training target $\mathbf{y}$ can be a Gaussian noise ϵ, the original data sample x_0 or velocity $v = \sqrt{1 - \bar{\alpha}}\epsilon - \sqrt{\bar{\alpha}}x_0$.

Early diffusion models used U-net [6, 31] as their backbone. Recently, Transformer-based diffusion models [25] with adaptive layer normalization (AdaLN) [26] have become mainstream, showing significant advantages in scaling up.

3.2 Mixture of Experts

Mixture-of-Experts (MoE) is a neural network layer comprising a router $\mathcal{R}$ and a set $\{E_i\}$ of N_E experts, each specializing in a subset of the input space and implemented as FFN. The router maps the input $X \in \mathbb{R}^{B\times L\times D}$ into token-expert affinity scores $\mathbf{S} \in \mathbb{R}^{B\times L\times E}$, trailed by a gating function $\mathcal{G}$:

$$\mathbf{S} = \mathcal{G}(\mathcal{R}(x)). \tag{3}$$

The input will be assigned to a subset of experts with top-k highest scores for computation and its output is the weighted sum of these experts' output. A unified expression is as follows:

$$\mathbf{G} = \begin{cases} \mathbf{S}, & \text{if } \mathbf{S} \in \texttt{TopK}\left(\mathbf{S}, \mathcal{K}\right) \\ 0, & \text{Otherwise} \end{cases} \tag{4}$$

$$\text{MoE}(X) = \sum_{i\in N_E} G_i(X) * E_i(X) \tag{5}$$

where $\mathbf{G} \in \mathbb{R}^{B\times L\times E}$ is the final gating tensor and $\texttt{TopK}(\cdot, \mathcal{K})$ is an operation that build a set with $\mathcal{K}$ largest value in tensor.

To maintain a constant number of activated parameters while increasing the top-k expert selection, the MoE model often splits the inner hidden dimension of each expert based on the top-k value, named fine-grained expert segmentation [4]. In the subsequent discussions, an "**x-in-y**" MoE means there are y candidate experts, with the top-x experts activated, and the hidden dimension of expert's intermediate layer will be divided by x.

3.3 The Rationality of Using MoE in Diffusion Models

Diffusion models possess several distinctive characteristics.

- Multi-task in nature, tasks at different timesteps predicting the target are not identical. Prior works like e-diff [1] validate this dissimilarity.
- Redundancy of image tokens. The information density varies across different regions, leading to unequal difficulties in generation.

Given these traits, MoE presents a suitable architecture for diffusion models. Its routing module can flexibly allocate and combine tokens and experts based on the predicted difficulties. We consider the allocation process as a distribution of computational resources. More challenging timesteps and complex image patches should be allocated to more experts. Achieving this requires a routing strategy with sufficient flexibility to distribute resources with broader degrees of freedom. Our method is designed following this principle.

4 Taming Diffusion Models with Expert Race

4.1 General Routing Formulation

For computational tractability, we decompose the original score tensor $\mathbf{S}$ into two operational dimensions through permutation and reshaping, obtaining matrix $\mathbf{S}' \in \mathbb{R}^{D_A\times D_B}$, where

- D_B: Size of the expert candidate pool;
- D_A: Number of parallel selection operations.

This dimensional reorganization enables independent top-k selection within each row while preserving cross-row independence.

Following the sparse gating paradigm in [20, 44], we control the MoE layer sparsity through parameter k, which specifies the expected number of activated experts per token. To satisfy system capacity constraints,

the effective selection size per candidate pool is defined as:

$$\mathcal{K} = \frac{k}{N_E} \cdot D_B. \tag{6}$$

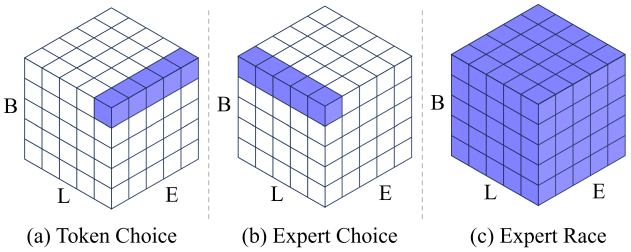
The routing objective, aligning with the optimization framework in [21], formalizes as the maximization of aggregated gating scores:

$$\max \sum_{i=1}^{D_A} \sum_{j \in \mathcal{T}_i} \mathbf{S'}_{i,j}, \tag{7}$$

where $\mathcal{T}_i$ denotes the set of indices corresponding to the top-$\mathcal{K}$ values in the i-th row of $\mathbf{S}'$.

Suboptimal in Conventional Strategies As shown in figure 3, the unified framework generalizes existing routing methods through top-$\mathcal{K}$ selection in $\mathbf{S}'$. However, standard row-wise approaches like Token-Choice and Expert-Choice exhibit inherent sub-optimality. These selection methods struggle to achieve optimal allocation in practice, as the required uniform distribution of top $\mathcal{K} \times D_A$ elements across rows, which is necessary for attaining the theoretical optimum in equation (7), rarely holds with real-world data distributions.

In practical scenarios like image diffusion model training, generation complexity varies across two key dimensions: denoising timesteps (B) and spatial image regions (L). To address this computational heterogeneity, the routing module must dynamically allocate more experts to tokens with greater generation demands. However, the token-choice strategy, since D_A is constituted by dimensions $B\&L$, both dimensions will receive an identical amount of activation experts. Expert-Choice mitigates this issue but remains constrained by its L-dimensional top-$\mathcal{K}$ selection, limiting optimal allocation potential.



(a) Token Choice (b) Expert Choice (c) Expert Race

Method	D_A	D_B	$\mathcal{K}$
Token-choice	$B * L$	E	k
Expert-choice	$B * E$	L	$k * L/E$
Expert-Race	1	$B * L * E$	$B * L * k$

Figure 3 Top-$\mathcal{K}$ Selection Flexibility and Specifications. B: batch size; L: sequence length; E: the number of experts. (a) Token Choice selects top-$\mathcal{K}$ experts along the expert dimension for each token. (b) Expert Choice selects top-$\mathcal{K}$ tokens along the sequence dimension for each expert. (c) Expert Race selects top-$\mathcal{K}$ across the entire set.

4.2 Expert Race

To address these limitations, we propose Expert-Race, which performs global top-$\mathcal{K}$ selection across all gating scores in a single routing pass. The "Race" mechanism provides an optimal solution to equation (7) by setting $D_A = 1$, ensuring the selected $\mathcal{K}$ elements are globally maximal. This design maximizes router flexibility to learn adaptive allocation patterns, enabling arbitrary expert-to-token assignments and dynamic allocation based on computational demands. However, applying Expert-Race naively presents two challenges.

Gating Function Conflict. While softmax over the expert dimension is standard for score normalization in existing routing strategies, it disrupts cross-token score ordering in Race. Additionally, applying softmax across the full sequence incurs high computational costs and risks numerical underflow as sequence length grows. We therefore explore alternative activation functions, finding through table 1 that the `identity` $\mathcal{G}(x) = x$ yields improved results.

Training-Inference Mismatch. Batch-wise candidate aggregation creates a fundamental mismatch between training and inference. During training, samples influence each other's routing selection and timesteps are randomly sampled per batch, whereas inference operates on independent samples with consistent timesteps. Since timesteps directly control noise mixing levels, this inconsistency degrades generation quality and can

Algorithm 1 Pytorch-style Pseudocode of Expert-Race

```
# m: momentum
# tau: ema updated threshold
# x: input of shape (B, L, D)
# experts: a list of FFN
# router: a linear layer

# Compute router logits for each token
logits = router(x)    # (B, L, E)
score = logits.flatten()
gates = nn.Identity()(logits) # activation
expect_k = B * L * k

# Get kthvalue and update threshold
if training:
    kth_val = torch.kthvalue(score, k=expect_k)
    mask = score >= kth_val
    tau = m * tau + (1. - m) * kth_val
else:
    mask = score >= tau

# Process tokens by each expert
expert_outputs = torch.stack([expert(x) for expert in experts], dim=-1)

# Aggregate the output by mask
gates = gates * mask
output = torch.sum(gates.unsqueeze(-1) * expert_outputs, dim=-1)
```

lead to model failure. At the same time, the mutual influence between samples during routing selection causes unstable inference. To mitigate these effects, we propose a learnable threshold τ that estimates the $\mathcal{K}$-th largest value through exponential moving average (EMA) updates during training.

$$\tau \leftarrow m\tau + (1-m) \cdot \frac{1}{D_A} \sum_{i=1}^{D_A} \mathbf{S'}_{i,\mathcal{K}}, \tag{8}$$

where $\mathbf{S'}_{i,\mathcal{K}}$ represents the $\mathcal{K}$-th largest element in the i-th row of $\mathbf{S'}$. This adaptive threshold is directly applied during inference, ensuring sample independence and consistent performance.

Pseudocode. We provide core pseudocode in PyTorch style in algorithm 1, illustrating how the expert selects the k-th largest logits and updates the threshold. Our algorithm is easy to implement, requiring only minor modifications to the existing MoE framework.

5 Load Balancing via Router Similarity Loss

In MoE systems, balanced token allocation across experts remains a critical engineering challenge. For our proposed Race strategy, the increased policy flexibility imposes greater demands on routing balancing.

5.1 Mode Collapse in Balancing Loss.

The conventional balancing loss [8, 34], originally designed for token-choice, promotes load balance by enforcing uniform token distribution across experts, thereby preventing dominance by a small subset of experts. However, by only constraining the marginal distribution of scores per expert, this approach fails to prevent collapse between experts with similar selection rules. As shown in figure 4, if multiple experts follow the same rules for selecting tokens, they are downgraded to one wider expert. Although such configurations satisfy balance loss constraints, they undermine the specialization benefits of fine-grained expert design [4], ultimately degrading overall performance.

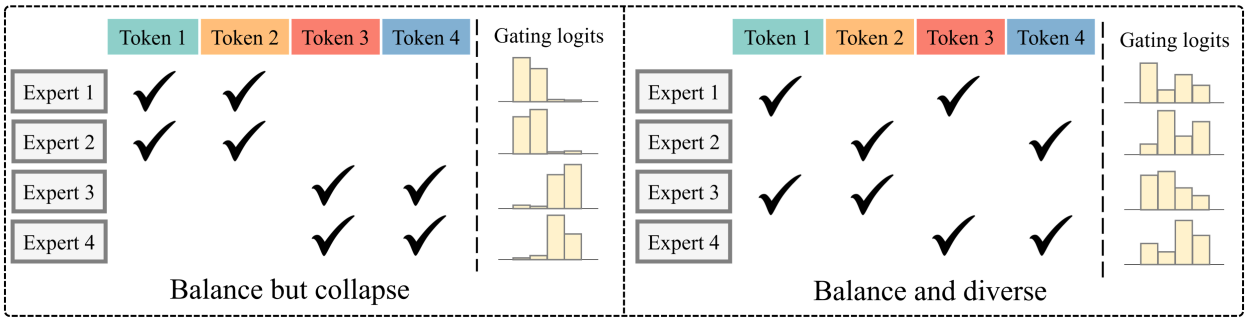


Figure 4 Toy examples of token assignment. Both of the two cases show perfect load balance that each expert process two tokens. But in the case above, experts 1 and 2 are assigned the same token, as are experts 3 and 4, where the 2-in-4 MoE collapse into 1-in-2. The example below shows a more diverse assignment, making full use of the expert specialization.

5.2 Router Similarity Loss.

To tackle this issue, we propose maximizing expert specialization by promoting pairwise diversity among experts. Specifically, inspired by [40], we compute cross-correlation matrices and minimize their off-diagonal elements to encourage expert differentiation. Given the router logits $S \in \mathbb{R}^{(B\times L)\times E}$, we apply softmax along the expert dimension to obtain normalized probabilities P, and compute two correlation matrices

$$M' = M^T M, \quad P' = P^T P \tag{9}$$

where M is the indicator matrix that $M_{i,j} = 1$ if expert j selects the i-th token and 0 otherwise.

Then, we define the **router similarity loss**:

$$\mathcal{L}_{\text{sim}} = \frac{1}{T} \sum_{i,j\in[1,E]} W(i,j) \cdot P'_{i,j} \tag{10}$$

where $W(i,j)$ is the weighting function defined as:

$$W(i,j) = \begin{cases} \frac{M'_{i,j}}{\sum_{i=j} M'_{i,j}} \cdot E, & \text{if } i = j \\ \frac{M'_{i,j}}{\sum_{i\neq j} M'_{i,j}} \cdot (E^2 - E), & \text{if } i \neq j \end{cases} \tag{11}$$

In more detail, the off-diagonal elements denote the similarity between each pair of experts based on token selection patterns in the current batch. From a probabilistic perspective, $P'_{i,j}$ captures the joint probability of a token being routed to both expert i and j. This formulation regularizes consistent co-selection patterns across experts while promoting diverse expert combinations. Regarding the diagonal elements, $P'_{i,i}$ represents a geometric mean version of the balance loss, effectively encouraging individual expert utilization (see appendix B for detailed analysis).

6 Per-Layer Regularization for Efficient Training

DiT employs adaptive layer normalization (adaLN) when introducing conditions. In the pre-normalization (pre-norm) architecture, we observe that adaLN progressively amplifies the outputs of deeper layers. This causes the output magnitudes of shallow layers to be relatively diminished, as illustrated in figure 5. This imbalance results in the learning speed of shallow layers lagging behind that of deeper layers, which is detrimental to the MoE training process. This imbalance has both advantages and disadvantages. On one hand, the outputs from deeper layers are more accurate, and their larger magnitudes make them less susceptible to the substantial noise present in shallow layers, facilitating more precise regression results. On the other hand, due to the presence of the normalization module in the final layer, the component of shallow

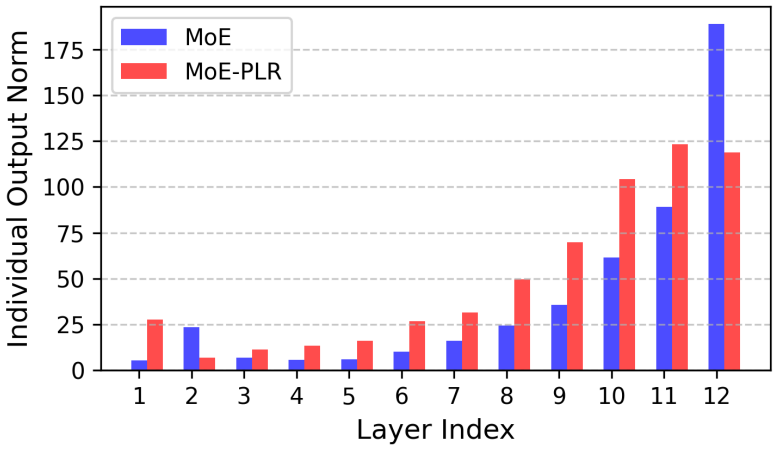


Figure 5 The norm of each block's output before added to the shortcuts. The output norm increases rapidly in deep layers, resulting in the weakening of shallow-layer components. This issue is alleviated with our proposed per-layer regularization.

layers in the residuals is diminished, posing a risk of gradient vanishing and resulting in the learning speed of shallow layers lagging behind that of deeper layers, which is detrimental to the MoE training process.

To mitigate this issue, we introduce a pre-layer regularization that enhances gradients in a supervised manner without altering the core network structure. Specifically, given the hidden output $\mathbf{h}_l$ from the l-th layer, we add a projection layer $\mathcal{H} : \mathbb{R}^{L' \times d} \to \mathbb{R}^{L \times d}$ to predict the final target $\mathbf{y} \in \mathbb{R}^{L \times d}$, where L and L' represent the number of patches before and after the patchify operation []. The pre-layer loss is defined as:

$$\mathcal{L}_{\text{PLR}} = \mathbb{E}_{x_0,t,\epsilon,l}\left[\frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{y}^{[n]} - \mathcal{H}(\mathbf{h}_l)^{[n]}\right\|^2\right] \tag{12}$$

where N is the total number of patches, and n is the patch index. In our implementation, the projection layer is integrated into the MLP router (see appendix A for details). By supervising the projection layer's predictions against final targets, we enhance shallow layer contributions during training, improving overall MoE performance.

7 Experiments

7.1 Implementation Details

Following training configurations in [25], we conduct experiments on ImageNet [5], employ AdamW optimizer, set batch size to 256, and use constant learning rate 1e-4 without weight decay for models of any size. During initialization, we use zero-initialization on all adaLN layers, and xavier initialization of uniform distribution on all linear layers. When training MoE, we substitute all FFNs with a MoE block and make sure they activate same amount of parameters. Specially, we set a smaller initialization range for each expert, enlarging the inner dimension by a factor of k to make the intialization range the same with its dense counterpart. We employ our per-layer regularization with weight 1e-2 and router similarity loss with 1e-4. We maintain an exponential moving average (EMA) of model weights over training and report all results using the EMA model.

The metrics used include FID [14], CMMD [17], and CLIP Score [27]. We present a series of MoE size configurations, denoted as k-in-E where E represents the total number of experts and k indicates the number of average activated experts. Additionally, we set the inner hidden dimension of each expert to be $1/k$ of its dense counterpart to ensure that the number of activated parameters remains the same. In all ablation studies, we train DiT-B and its 2-in-8 MoE variant for 500K iterations unless specified otherwise.

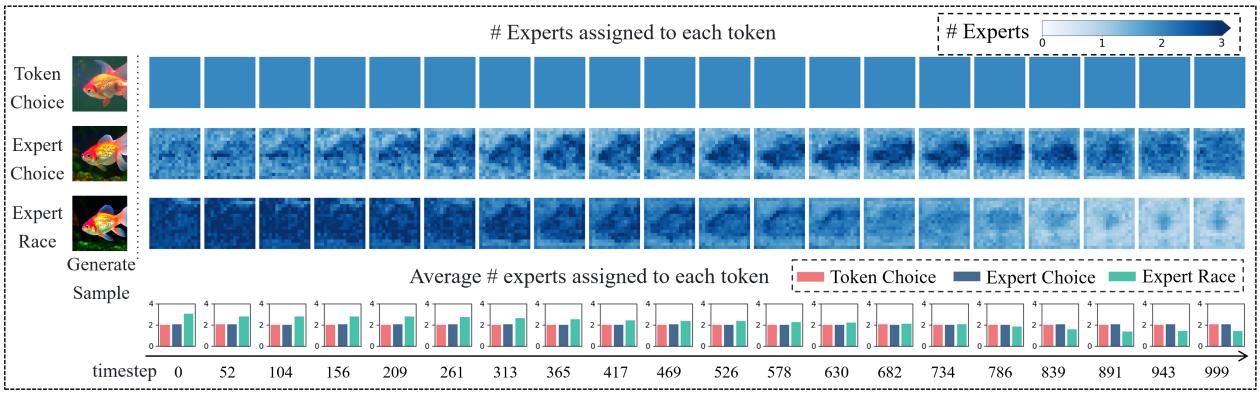


Figure 6 Average token allocation at different time steps. Expert-Race assigns more experts to the more complex denoising time steps, which occur at lower timestep indices that handle finer-grain image details.

Table 1 Ablation study on routing strategy and gating function.

Routing	Gating	FID↓	CMMD↓	CLIP↑
Token Choice	softmax	17.28	.7304	21.87
Expert Choice		16.71	.7267	21.95
Expert Race		16.47	.7104	21.97
Token Choice	sigmoid	15.25	.6956	22.09
Expert Choice		15.73	.6821	22.06
Expert Race		13.85	.6471	22.23
Token Choice	identity	15.98	.6938	22.01
Expert Choice		15.70	.6963	22.04
Expert Race		**13.66**	**.6317**	**22.25**

7.2 Routing Strategy

Expert racing enables extensive exploration within the logit space during training, enabling complexity-aware expert allocation across diffusion timesteps. As shown in figure 6, along the timestep dimension, our method initially uses fewer experts at the beginning of denoising (higher timestep indices) and dynamically assigns more experts to tokens at timesteps requiring higher image detail (lower timestep indices). Along the spatial dimension, the assignment of computation follows a "diffusion process" from the center of image to significant object and then to the entire image. This indicates that the model first focuses on object construction and subsequently refines the details. In contrast, both token-choice and expert-choice strategies maintain fixed average expert allocations per timestep, lacking the temporal dynamic allocation capability. From the result proposed in table 1, expert race outperforms other routing strategies by a significant margin. Within the framework proposed in section 4.1, we conducted ablation studies on routing strategies for combinations of different dimensions. See section 7.7 for more results.

7.3 Gating Function

table 1 shows that identity gating outperforms both softmax and sigmoid variants. In this experiment, we isolate other components (learnable threshold and regularizations) to verify the impact of the gating function on performance. We found that identity gating outperforms softmax and sigmoid under expert-race, and it enhances both token-choice and expert-choice compared to softmax. Under expert-race, both sigmoid and identity significantly outperform softmax. We attribute this to the fact that softmax normalizes scores of each token along the expert dimension, disrupting the partial ordering of scores across different tokens. In contrast, sigmoid and identity preserve this partial ordering, ensuring that important token-expert combinations are selected, thereby improving performance.

7.4 Load Balance

table 2 compares our proposed router similarity loss with conventional load balancing loss [4]. This setting is similar to the gating function ablation above, but here the MoE is 4-in-32 to further observe the impact of load balancing. The `MaxVio` [37] metric measures how much the most violated expert exceeds its capacity limit. Combination Usage Ratio, abbreviated as `Comb`, estimates the proportion of activations for each pairwise combination of experts (higher is better). See appendix C for details.

The weights for both the load balance loss and the router similarity loss are set to 1e-4, as this configuration yielded the highest generation quality in our ablation experiments. Results show our method improves expert combination ratio, image generation quality, and load-balancing performance.

figure 7 further demonstrates the evaluation of MoE configurations across multiple scales (4-in-16,32,64,128), highlighting our approach's capability to diversify expert activation pattern compared to existing methods.

Table 2 Load balance for 4-in-32 MoE.

Setting	FID↓	MaxVio↓	Comb↑
No Constraint	11.38	6.383	18.98
Balance Loss	11.67	2.052	72.36
Router Similarity	**10.77**	0.850	**83.10**

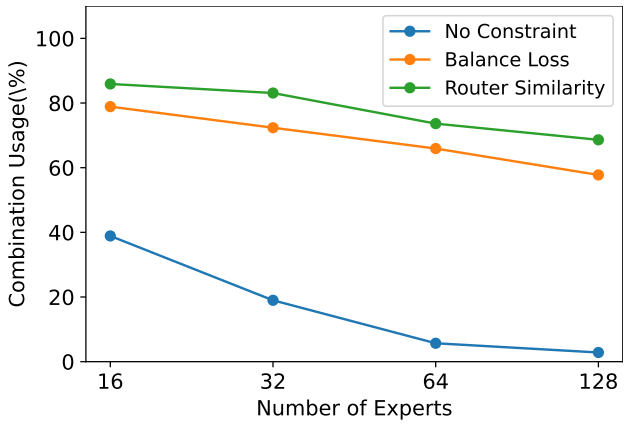


Figure 7 Combination usage comparison between different number of experts.

7.5 Core Components

table 3 demonstrates the improvements brought by each component. Starting from the baseline of expert racing with softmax gating, we incrementally added identity gating, learnable thresholds, per-layer regularization, and router similarity loss. Notably, FID and CMMD consistently decrease and CLIP score increases with the addition of each technique, indicating enhancements in image quality and alignment with conditional distributions.

Table 3 Ablation study of core components.

Setting	FID↓	CMMD↓	CLIP↑
Expert Race (softmax)	16.47	.7104	21.97
+ Identity Gating	13.66	.6317	22.25
+ Learnable Threshold	11.56	.5863	22.56
+ Per-Layer Reg.	8.95	.4847	22.94
+ Router Similarity	**8.03**	**.4587**	**23.09**

7.6 Scaling Law

We first scale the base model sizes in the full pipeline, as detailed in table 4. We have four settings: Base (B), Medium (M), Large (L), and Extra-large (XL). Under the same configuration we compared our 4-in-32 MoE models with their dense counterparts, noting that their activation parameter counts are nearly identical. The experimental results and figure 8 demonstrate that our model significantly outperforms the corresponding Dense models given same activation parameter count. Furthermore, our MoE-4in32 model surpasses the XL-Dense model with less than half the total number of parameters, further showcasing the efficiency of our model design.

Table 4 Model specifications and evaluation results of the comparison between MoE and Dense models.

Model Config.	Total Param.	Activate	# Layers	Hidden	# Heads	FID↓	CMMD↓	CLIP↑
B/2-Dense	0.127B	0.127B	12	768	12	18.03	.7532	21.83
M/2-Dense	0.265B	0.265B	16	960	16	11.18	.5775	22.56
L/2-Dense	0.453B	0.453B	24	1024	16	7.88	.4842	23.00
XL/2-Dense	0.669B	0.669B	28	1152	16	6.31	.4338	23.27
B/2-MoE-4in32	0.531B	0.135B	12	768	12	7.35	.4460	23.15
M/2-MoE-4in32	1.106B	0.281B	16	960	16	5.16	.3507	23.50
L/2-MoE-4in32	1.889B	0.479B	24	1024	16	4.04	.2775	24.12
XL/2-MoE-4in32	2.788B	0.707B	28	1152	16	**3.31**	**.1784**	**24.68**

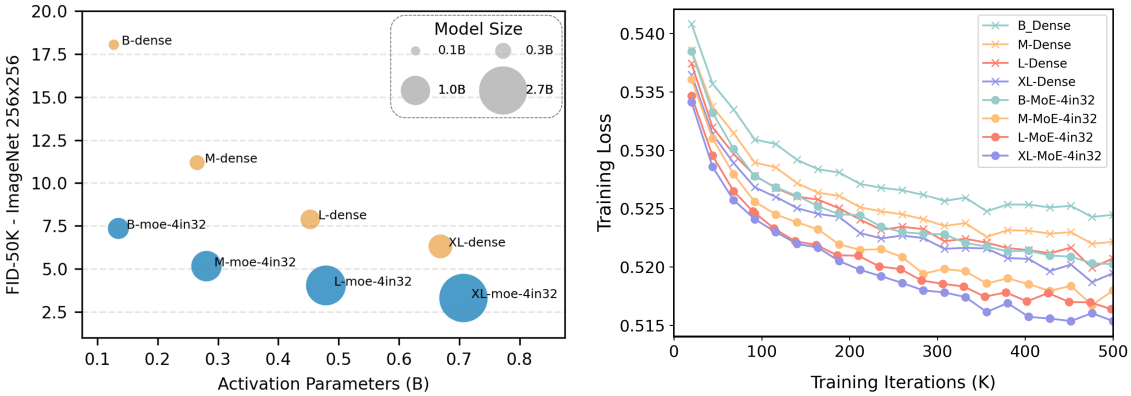


Figure 8 A comparison between Dense and our MoE models. Our MoE models consistently outperform their Dense counterparts across the FID and training curves. Notably, the MoE model with activation size **M** shows better performance compared to the Dense model scaled to size **XL**.

We further expand the model's size while **maintaining the same number of activation parameters** under the model configurations (**B/2-MoE**). The model expansion is achieved by varying the hidden split ratios of experts and increasing the number of candidate experts. As shown in figure 9 and table 5, increasing both the number of candidate experts and splitting the hidden dimensions of MoE leads to improvement in performance, highlighting the potential of our MoE architecture for scaling up.

Table 5 Evaluation results of different MoE configurations with the same number of activation parameters.

Config.	Hidden Split	Total Param.	FID↓	CMMD↓	CLIP↑
1-in-4		0.297B	9.70	.5200	22.82
1-in-8	1	0.524B	9.05	.4976	22.91
1-in-16		0.978B	8.65	.5019	22.92
2-in-8		0.297B	8.03	.4587	23.09
2-in-16	2	0.524B	7.78	.4607	23.06
2-in-32		0.977B	7.57	.4483	23.12
4-in-16		0.297B	7.78	.4628	23.09
4-in-32	4	0.524B	7.35	.4460	23.15
4-in-64		0.977B	6.91	.4244	23.21
8-in-32		0.297B	7.56	.4516	23.11
8-in-64	8	0.524B	6.87	.4263	23.24
8-in-128		0.977B	**6.28**	**.4015**	**23.35**

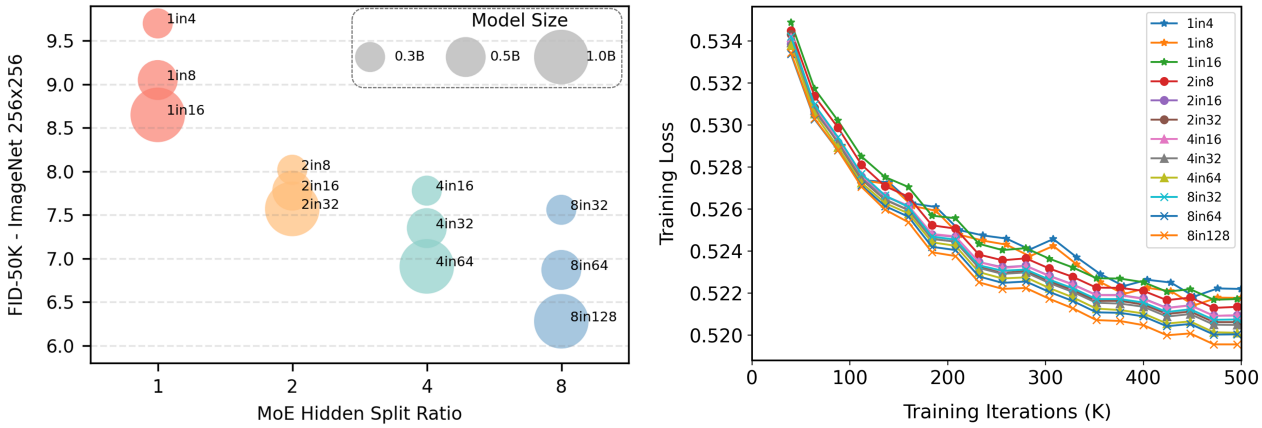
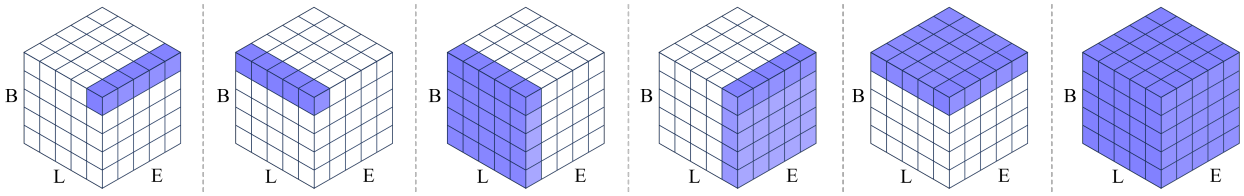


Figure 9 Scaling results of DiT-B in different MoE configurations. Our method demonstrates linear performance improvement when varying expert split ratios and increasing the number of candidate experts.

7.7 Extended Routing Strategy

We extend the token-choice and expert-choice routing strategies by introducing varying degrees of routing selection flexibility, aiming to investigate how training-stage router freedom impacts final model performance. All compared methods are trained for 500K iterations under identical configurations: a full pipeline with DiT-B/2-MoE-2-in-8 architecture. To ensure experimental consistency, all approaches employ learnable thresholds for inference queries. Specifically, we develop three new strategies: BL-Choice, BE-Choice, and LE-Choice, obtained through pairwise combinations of selection dimensions - Batch (B), Sequence Length (L), and Expert count (E), as illustrated in figure 10. Experimental results in table 6 demonstrate that strategies with higher training selection freedom consistently outperform conventional fixed-dimension top-k selection approaches (such as token-choice/expert-choice). Notably, our Expert Race achieves the best performance across all evaluated routing strategies.



(a) Token Choice (b) Expert Choice (c) BL Choice (d) BE Choice (e) LE Choice (f) Expert Race

Figure 10 Top-$\mathcal{K}$ selection flexibility in more extended routing strategies.

Table 6 Design Choices and Evaluation Results of Different Routing Strategies

	Token Choice	Expert Choice	BL Choice	BE Choice	LE Choice	Expert Race
D_A	$B*L$	$B*E$	E	L	B	1
D_B	E	L	$B*L$	$B*E$	$L*E$	$B*L*E$
$\mathcal{K}$	k	$k*L/E$	$B*L*k/E$	$B*k$	$L*k$	$B*L*k$
FID↓	9.50	10.13	9.08	8.28	8.89	**8.03**
CMMD↓	.5202	.5639	.5145	.4636	.4871	**.4587**
CLIP↑	22.81	22.73	22.87	23.05	22.99	**23.09**

7.8 More Results on ImageNet 256×256

Figure 11 Random generated 256×256 samples from RaceDiT-XL/2-4in32. Classifier-free guidance scale is 4.

We present additional results on the ImageNet 256×256. Our model is capable of generating high-quality images, as evidenced by the random samples shown in figure 11 and the single-class generation results presented in appendix E. We further provide a comparison with leading approaches shown in table 7, where *Samples* is reported as *training iterations* $\times$ *batch size*.

In our experiments, we train a vanilla DiT (marked with *) and a MoE model with Expert Race following the training protocol from the original DiT paper [25], except that we use a larger batch size **1024** to improve memory utilization and train for only **1.75M** steps. The learning rate remains unchanged at $1e^{-4}$. We also provide the training curves in figure 1 (Right). Our MoE model, of similar amount of activated parameters, achieves better performance and faster convergence.

Table 7 Comparison with other methods on ImageNet 256x256.

Model Config.	Total	Activated	Samples	FID↓	IS↑	Precision↑	Recall↑
ADM-G [6]	0.608B	0.608B	2.0M × 256	4.59	186.70	0.82	0.52
LDM-8-G [31]	0.506B	0.506B	4.8M × 64	7.76	209.52	0.84	0.35
MDT [11]	0.675B	0.675B	2.5M × 256	2.15	249.27	0.82	0.58
MDT [11]	0.675B	0.675B	6.5M × 256	1.79	283.01	0.81	0.61
DiT-XL/2 [25]	0.669B	0.669B	7.0M × 256	2.27	278.24	0.83	0.57
SiT-XL [22]	0.669B	0.669B	7.0M × 256	2.06	277.50	0.83	0.59
MaskDiT [43]	0.737B	0.737B	2.0M × 1024	2.50	256.27	0.83	0.56
DiT-MoE-XL/2 [9]	4.105B	1.530B	7.0M × 1024	1.72	315.73	0.83	0.64
DiT-XL/2*	0.669B	0.669B	1.7M × 1024	3.02	261.49	0.81	0.51
RaceDiT-XL/2-4in32	2.788B	0.707B	1.7M × 1024	2.06	318.64	0.83	0.60

8 Conclusion

This paper proposes Expert Race, a novel Mixture-of-Experts (MoE) routing strategy that enables stable and efficient scaling of diffusion transformers. Compared to previous methods with fixed degrees of freedom in expert-token assignments, our strategy achieves higher routing flexibility by enabling top-k selection across the full routing space spanning batch, sequence, and expert dimensions. This expanded selection capability provides greater optimization freedom, significantly improving performance when scaling diffusion transformers. To address challenges from increased flexibility, we propose an EMA-based threshold adaptation mechanism that mitigates timestep-induced distribution shifts between training (randomized per-sample timesteps) and inference (uniform timesteps), ensuring generation consistency. Additionally, per-layer regularization improves training stability, while router similarity loss promotes diverse expert combinations and better load balancing, as shown on 256×256 ImageNet generation tasks. As a general routing strategy, future work will extend Expert Race to broader diffusion-based visual tasks.

References

[1] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Qinsheng Zhang, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, et al. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers. arXiv preprint arXiv:2211.01324, 2022.

[2] Raphael Bensadoun, Tom Monnier, Yanir Kleiman, Filippos Kokkinos, Yawar Siddiqui, Mahendra Kariya, Omri Harosh, Roman Shapovalov, Benjamin Graham, Emilien Garreau, et al. Meta 3d gen. arXiv preprint arXiv:2407.02599, 2024.

[3] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024.

[4] Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. arXiv preprint arXiv:2401.06066, 2024.

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

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. Advances in Neural Information Processing Systems (NeurIPS), 34:8780–8794, 2021.

[7] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In International Conference on Machine Learning (ICML), 2024.

[8] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research (JMLR), 23(120):1–39, 2022.

[9] Zhengcong Fei, Mingyuan Fan, Changqian Yu, Debang Li, and Junshi Huang. Scaling diffusion transformers to 16 billion parameters. arXiv preprint arXiv:2407.11633, 2024.

[10] Zhida Feng, Zhenyu Zhang, Xintong Yu, Yewei Fang, Lanxin Li, Xuyi Chen, Yuxiang Lu, Jiaxiang Liu, Weichong Yin, Shikun Feng, et al. Ernie-vilg 2.0: Improving text-to-image diffusion model with knowledge-enhanced mixture-of-denoising-experts. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 10135–10145, 2023.

[11] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Masked diffusion transformer is a strong image synthesizer. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 23164–23173, 2023.

[12] Yongxin Guo, Zhenglin Cheng, Xiaoying Tang, Zhaopeng Tu, and Tao Lin. Dynamic mixture of experts: An auto-tuning approach for efficient transformer models. arXiv preprint arXiv:2405.14297, 2024.

[13] Agrim Gupta, Lijun Yu, Kihyuk Sohn, Xiuye Gu, Meera Hahn, Fei-Fei Li, Irfan Essa, Lu Jiang, and José Lezama. Photorealistic video generation with diffusion models. In European Conference on Computer Vision (ECCV), pages 393–411. Springer, 2024.

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in Neural Information Processing Systems (NeurIPS), 30, 2017.

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. Advances in Neural Information Processing Systems (NeurIPS), 33:6840–6851, 2020.

[16] Gagan Jain, Nidhi Hegde, Aditya Kusupati, Arsha Nagrani, Shyamal Buch, Prateek Jain, Anurag Arnab, and Sujoy Paul. Mixture of nested experts: Adaptive processing of visual tokens. arXiv preprint arXiv:2407.19985, 2024.

[17] Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking fid: Towards a better evaluation metric for image generation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 9307–9315, 2024.

[18] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. arXiv preprint arXiv:2401.04088, 2024.

[19] Yunsung Lee, JinYoung Kim, Hyojun Go, Myeongho Jeong, Shinhyeok Oh, and Seungtaek Choi. Multi-architecture multi-expert diffusion models. In AAAI Conference on Artificial Intelligence (AAAI), volume 38, pages 13427–13436, 2024.

[20] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. {GS}hard: Scaling giant models with conditional computation and automatic sharding. In International Conference on Learning Representations (ICLR), 2021.

[21] Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. Base layers: Simplifying training of large, sparse models. In International Conference on Machine Learning (ICML), pages 6265–6274, 2021.

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

[23] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. arXiv preprint arXiv:2112.10741, 2021.

[24] Byeongjun Park, Sangmin Woo, Hyojun Go, Jin-Young Kim, and Changick Kim. Denoising task routing for diffusion models. In International Conference on Learning Representations (ICLR), 2024.

[25] William Peebles and Saining Xie. Scalable diffusion models with transformers. In IEEE International Conference on Computer Vision (ICCV), pages 4195–4205, 2023.

[26] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In AAAI Conference on Artificial Intelligence (AAAI), volume 32, 2018.

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In International Conference on Machine Learning (ICML), pages 8748–8763. PMLR, 2021.

[28] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125, 2022.

[29] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. Advances in Neural Information Processing Systems (NeurIPS), 34:8583–8595, 2021.

[30] Stephen Roller, Sainbayar Sukhbaatar, Jason Weston, et al. Hash layers for large sparse models. Advances in Neural Information Processing Systems (NeurIPS), 34:17555–17566, 2021.

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

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

[33] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. Advances in Neural Information Processing Systems (NeurIPS), pages 36479–36494, 2022.

[34] Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In International Conference on Learning Representations (ICLR), 2017.

[35] Haotian Sun, Tao Lei, Bowen Zhang, Yanghao Li, Haoshuo Huang, Ruoming Pang, Bo Dai, and Nan Du. Ec-dit: Scaling diffusion transformers with adaptive expert-choice routing. arXiv preprint arXiv:2410.02098, 2024.

[36] Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. arXiv preprint arXiv:2408.15664, 2024.

[37] Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. arXiv preprint arXiv:2408.15664, 2024.

[38] Ziteng Wang, Jianfei Chen, and Jun Zhu. Remoe: Fully differentiable mixture-of-experts with relu routing. arXiv preprint arXiv:2412.14711, 2024.

[39] Zeyue Xue, Guanglu Song, Qiushan Guo, Boxiao Liu, Zhuofan Zong, Yu Liu, and Ping Luo. Raphael: Text-to-image generation via large mixture of diffusion paths. Advances in Neural Information Processing Systems (NeurIPS), 36, 2024.

[40] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In International Conference on Machine Learning (ICML), pages 12310–12320. PMLR, 2021.

[41] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. ACM Transactions on Graphics (TOG), 43(4):1–20, 2024.

[42] Wangbo Zhao, Yizeng Han, Jiasheng Tang, Kai Wang, Yibing Song, Gao Huang, Fan Wang, and Yang You. Dynamic diffusion transformer. arXiv preprint arXiv:2410.03456, 2024.

[43] Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar. Fast training of diffusion models with masked transformers. arXiv preprint arXiv:2306.09305, 2023.

[44] Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. Mixture-of-experts with expert choice routing. Advances in Neural Information Processing Systems (NeurIPS), 35:7103–7114, 2022.

[45] Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models. arXiv preprint arXiv:2202.08906, 2022.

[46] Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Jianfeng Gao, and Tuo Zhao. Taming sparsely activated transformer with stochastic experts. In International Conference on Learning Representations (ICLR), 2022.

Appendix

A Implemention of the Per-Layer Regularization

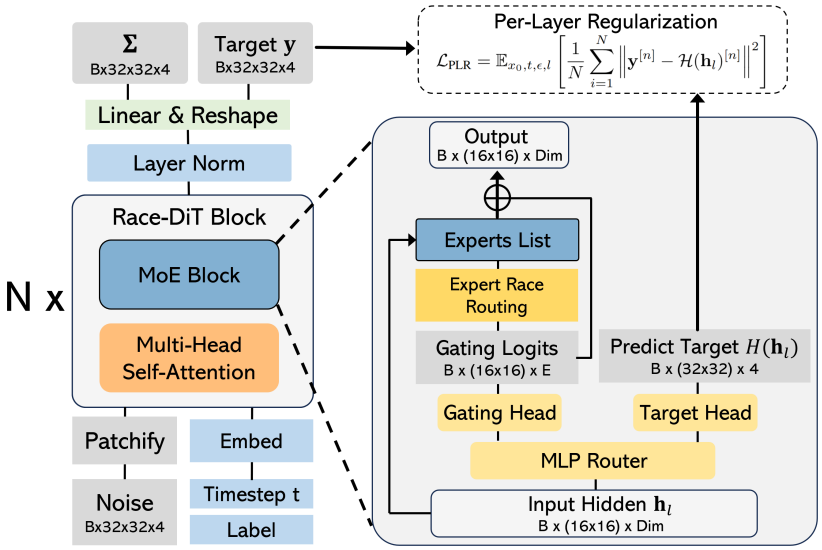


Figure 12 An illustration of the calculation for Per-Layer Regularization.

figure 12 illustrates the Per-Layer Regularization mechanism. The input $\mathbf{h}_l$ at layer l passes through a two-layer MLP router. The first layer applies linear transformation and GELU activation while preserving the original hidden dimension. The second layer branches into two heads: (1) a *gating head* producing routing logits, and (2) a *target head* predicting the output target $\mathbf{y}$ through linear projection. This dual-head design enables concurrent routing and target prediction. The L2 loss is computed per-layer between the target head's prediction $\mathcal{H}(\mathbf{h}_l)$ and ground truth $\mathbf{y}$. By aligning intermediate layer outputs with the final target, this regularization effectively enhances the contribution of shallow MoE layers to the final results in pre-norm architectures, significantly accelerating their optimization.

B Analysis of Router Similarity Loss

In this section, we will demonstrate that the router similarity loss is an extended form of the balance loss, expanding from constraining individual experts to combinations of experts.

Load balance loss [4] are designed to encourage tokens to be evenly distributed across experts. For an MoE configuration with E experts and a top-k value of K, the load balance loss for a batch with T tokens is as follows:

$$\begin{aligned}\mathcal{L}_{blc} &= \sum_{i=1}^{E} f_i \cdot P_i \\ &= \sum_{i=1}^{E}\left(\frac{E}{KT}\sum_{t=1}^{T}\mathbb{1}(i,t)\right)\left(\frac{1}{T}\sum_{t=1}^{T}s(i,t)\right)\end{aligned} \tag{13}$$

where $\mathbb{1}(i,t)$ is the indicator function that equals 1 when token t is assigned to expert i and $s(i,t)$ is the router logits of token t and expert i. Since softmax activation is applied over the expert dimension, $s(i,t)$ can be regarded as the probability that expert i is selected given token t as condition, marked as $p(i|t)$. Considering that each token is uniformly sampled from the dataset,

$$\frac{1}{T}\sum_{t=1}^{T}s(i,t) = \sum_{t=1}^{T}p(i|t)\cdot p(t) = P_i$$

denotes to the marginal probability of choosing expert i. And

$$f_i = \frac{E}{KT}\sum_{t=1}^{T}\mathbb{1}(i,t) = \frac{\sum_{t=1}^{T}\mathbb{1}(i,t)}{\frac{1}{E}\sum_{i=1}^{E}\sum_{t=1}^{T}\mathbb{1}(i,t)}$$

is the ratio of the actual number of tokens assigned to expert i to the expected number of tokens assigned to each expert.

As for router similarity loss, each element of matrix P' in equation (9) can be formulated as below:

$$\begin{aligned}\frac{1}{T}P'_{i,j} &= \frac{1}{T}\sum_{t=1}^{T} s(i,t)\cdot s(j,t)\\ &= \sum_{t=1}^{T} p(i|t)\cdot p(j|t)\cdot p(t)\\ &= p(i,j)\end{aligned} \tag{14}$$

The matrix P' actually represents the probability of the pair of experts i and j being selected. Similarly, each element in M' of equation (9) is the ratio of the actual number of tokens assigned to the expert pair (i,j) to the expected number of tokens assigned to each pairwise combination and thus $W(i,j)$ of equation (10) has a similar form to f_i in balance loss.

Noted that the above equations model the probability of sampling with replacement for experts. However, in practical implementation, sampling is performed without replacement, and the case that one expert is selected more than once does not exist. Therefore, in equation (11), we normalize the diagonal and off-diagonal parts of the matrix separately. The diagonal part degenerates into an estimation for individual experts as follows:

$$\begin{aligned}\mathcal{L}_{sim\_diag} &= \frac{1}{T}\sum_{i=1}^{E} W(i,i)\cdot P'_{i,i}\\ &= \frac{E}{T}\cdot\sum_{i=1}^{E}\frac{M'_{i,i}}{\sum_j^E M'_{j,j}}\cdot\sum_{t=1}^{T} s(i,t)^2\\ &= \frac{E}{T}\cdot\sum_{i=1}^{E}\frac{\sum_{t=1}^{T}\mathbb{1}(i,t)^2}{\sum_{j=1}^{E}\sum_{t=1}^{T}\mathbb{1}(j,t)^2}\cdot\sum_{t=1}^{T} s(i,t)^2\\ &= \frac{E}{T}\cdot\sum_{i=1}^{E}\frac{\sum_{t=1}^{T}\mathbb{1}(i,t)}{\sum_{j=1}^{E}\sum_{t=1}^{T}\mathbb{1}(j,t)}\cdot\sum_{t=1}^{T} s(i,t)^2\\ &= \frac{E}{T}\cdot\sum_{i=1}^{E}\frac{\sum_{i=1}^{T}\mathbb{1}(i,t)}{KT}\cdot\sum_{t=1}^{T} s(i,t)^2\\ &= \sum_{i=1}^{E}\left(\frac{E}{KT}\sum_{t=1}^{T}\mathbb{1}(i,t)\right)\left(\frac{1}{T}\sum_{t=1}^{T} s(i,t)^2\right)\end{aligned} \tag{15}$$

Therefore, the diagonal part of router similarity loss represents a geometric mean version of the balance loss as described above. The $W(i,i)$ in the router similarity loss is identical to f_i in the balance loss. Similarly, for the off-diagonal weights $W(i,j)$, we estimate the expected number of times that each expert pair is selected by dividing the sum of all elements, $\sum_{i\neq j} W(i,j)$, by the number of terms, $E^2 - E$. The complete router similarity loss can be viewed as an augmentation of the traditional balance loss, particularly by incorporating the pairwise interaction between experts, to help the model make more effective use of different combinations of experts. It is worth noting that we normalize both the weights and the probabilities, making the calculation of the loss function independent of the number of experts, top-k, and token sequence length. Considering a random assignment case, assume that the scores of the entire score matrix S are the same constant. In this case, $P_{i,j} = \frac{T}{E^2}$, substituting this into the equation (10) yields $\mathcal{L}_{sim} = 1$.

C Combination Usage

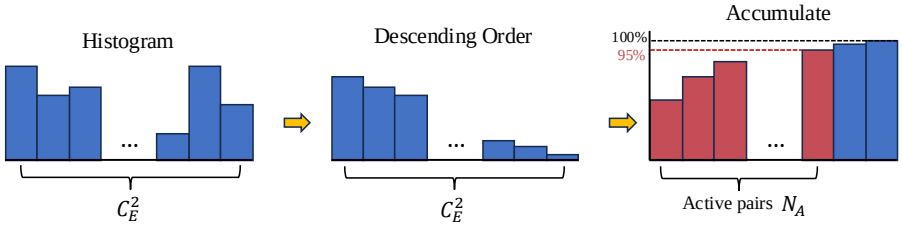


Figure 13 Compute process of Combination Usage.

To estimate the actual number of pairwise expert combinations activated and involved in computation, we use the metric called Combination Usage whose process is shown in the figure 13. From the perspective of pairwise combinations, for E experts, there are a total of $\binom{E}{2}$ pairwise combinations. First, we count the number of times each expert pair is selected across all tokens, resulting in a histogram. Then, we sort the counts in descending order and normalize them to obtain a sorted normalized histogram. Finally, we compute the cumulative sum and count the number of bins whose cumulative sum is less than 95%. Finally, we compute the cumulative sum and calculate the proportion of bins whose cumulative sum is less than 95% relative to the total number of bins, *i.e.*, N_A/C_E^2. This proportion is referred to as the Combination Usage. In other words, those expert pairs with their amount less than 5% of the total count are considered inactive. We set k=4 and conducted experiments with different numbers of experts, as shown in the figure 7. The experiment results indicate that our method effectively enhances the utilization of a larger number of combinations.

D Additional Comparisons with DiT-MoE

Table 8 Comparison to DiT-MoE

Model Config.	Total Param.	Activated Param.	Iters.	FID↓	CMMD↓	CLIP↑
DiT-MoE-B/2-8E2A [9]	0.795B	0.286B	500K	9.06	.5049	22.87
RaceDiT-B/2-2in8	0.297B	0.135B	500K	8.02	.4587	23.09

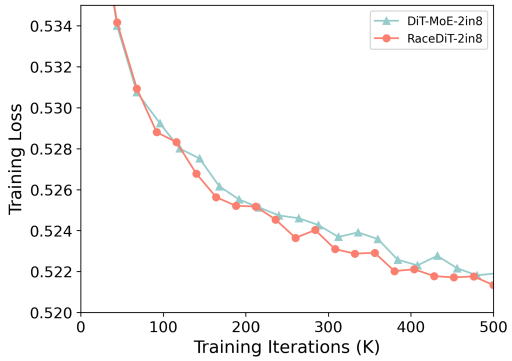


Figure 14 Training curve comparisons between DiT-MoE [9] and our model.

We provide an additional comparison with DiT-MoE [9], our most-related work using its official open-source code. We conduct experiments under the DiT-MoE-B/2-8E2A setting and compared it with our RaceDiT-B/2-2in8 model using the same training configuration. Both models were trained for 500K iterations.

There are several differences between the models: DiT-MoE uses GLU, while our method employs a standard MLP. Additionally, DiT-MoE includes two extra shared experts, which our model does not. As a result, our model has fewer total and activated parameters. Despite these differences, as demonstrated in table 8 and figure 14, our method achieves better training loss and evaluation metrics, even with a smaller number of activated parameters.

E Additional Image Generation Results

Figure 15 Samples from RaceDiT-XL/2-4in32 (256×256).
Classifier-free guidance: 4.0
Label: Macaw (88)

Figure 16 Samples from RaceDiT-XL/2-4in32 (256×256).
Classifier-free guidance: 4.0
Label: loggerhead turtle (33)

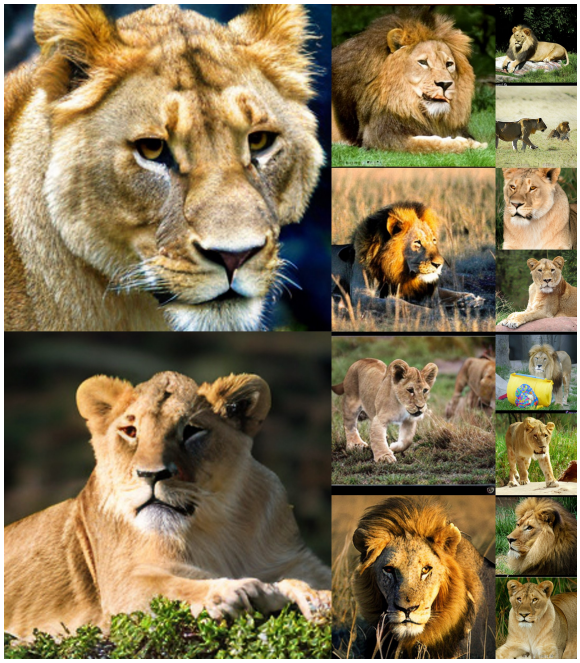


Figure 17 Samples from RaceDiT-XL/2-4in32 (256×256).
Classifier-free guidance: 4.0
Label: lion (291)

Figure 18 Samples from RaceDiT-XL/2-4in32 (256×256).
Classifier-free guidance: 4.0
Label: lesser panda (387)

Figure 19 Samples from RaceDiT-XL/2-4in32 (256×256).
Classifier-free guidance: 4.0
Label: convertible (511)

Figure 20 Samples from RaceDiT-XL/2-4in32(256×256).
Classifier-free guidance: 4.0
Label: balloon (417)

Figure 21 Samples from RaceDiT-XL/2-4in32 (256×256).
Classifier-free guidance: 4.0
Label: lakeside (975)

Figure 22 Samples from RaceDiT-XL/2-4in32 (256×256).
Classifier-free guidance: 4.0
Label: volcano (980)