Local Expert Diffusion Models for Efficient Training in Denoising Diffusion Probabilistic Models

Anonymous submission

Abstract

Diffusion models have emerged as a new standard technique in generative AI due to their huge success in various applications. However, their training can be prohibitively resource- and time-consuming, resulting in high-carbon footprint. To address this issue, we propose a novel and practical training strategy that significantly reduces the training time, even enhancing generation quality. We observe that diffusion models exhibit different convergence rates and training patterns at different time steps, inspiring our MDM (**M**ulti-expert **D**iffusion **M**odel). Each expert specializes in a group of time steps with similar training patterns. We can exploit the variations in iteration required for convergence among different local experts to reduce total training time significantly. Our method improves the training efficiency of the diffusion model by (1) reducing the total GPU hours and (2) enabling parallel training of experts without overhead to further reduce the wall-clock time. When applied to three baseline models, our MDM accelerates training $\times$2.7 - 4.7 faster than the corresponding baselines while reducing computational resources by 24 - 53%. Furthermore, our method improves FID by 7.7% on average, including all datasets and models.

Introduction

Diffusion models have emerged as a powerful new family of generative models for both conditional (Li et al. 2022; Lugmayr et al. 2022; Nichol et al. 2022; Rombach et al. 2022; Saharia et al. 2022) and unconditional (Ho, Jain, and Abbeel 2020; Nichol and Dhariwal 2021; Song et al. 2021b) generation tasks, offering notable advantages over existing models, such as generative adversarial networks (GANs (Goodfellow et al. 2014)). The advancements in model design and training strategies (Dhariwal and Nichol 2021; Karras et al. 2022; Nichol and Dhariwal 2021) have led diffusion models to beat the current state-of-the-art in several fields (Deng et al. 2009; Yu et al. 2015).

However, training large-scale diffusion models is extremely expensive and time-consuming. Training time increases quadratically by the resolution of the dataset. For instance, training a diffusion model on 512×512 ImageNet (Deng et al. 2009) dataset using a single V100 GPU (Dhariwal and Nichol 2021) takes up to 1914 days. This substantial training expenses cause the emission of a huge amount of carbon dioxide, resulting in environmental destruction. In this paper, our research objective centers on improving the training efficiency of diffusion models.

The training efficiency can be evaluated from two perspectives: (1) the total cost of fully training a model (**TC**), measured in GPU days, and (2) the actual training time (wall-clock time, **WCT**), measured in days. The relationship between them can be expressed as TC $=$ WCT $\times$ RT, where **RT** denotes resource throughput, representing the number of distributed GPUs or nodes employed. For example, if a model takes 100 V100 days (TC) to converge, it takes 25 days (WCT) with four V100 GPUs (RT), assuming ideally distributed training. Considering both TC and WCT are essential when evaluating training efficiency.

To reduce WCT, we can increase the RT of the model by parallelizing the training process across multiple modules. However, computational overhead from communication between devices (Shi, Wang, and Chu 2018; Wu et al. 2022) slows down training. For example, if the batch size is split in half between two GPUs, the WCT should be 50% compared to training with single GPU. However, the actual WCT is around 58% due to the computational overhead. If this situation is extended from inter-GPU to inter-node, this overhead significantly increases.

With this objective in mind, we focus on the inherent property of time-independent training in diffusion models. Training each time step x_t is conducted independently (Song, Meng, and Ermon 2020) across the entire time step range $t \in (0, T]$ (where $t = T$ represents the fully noisy step). Our investigation reveals that there is a significant variation in convergence speed among each timestep. Based on this observation, we propose a multi-expert diffusion model (MDM), an algorithm that accelerates training via time step-adaptive local experts. We carefully identify three time intervals, each exhibiting a similar training pattern based on an activation analysis. Then, we train three experts independently, each responsible for each interval. Since MDM consists of multiple independent experts, it naturally aligns with exploiting sufficiently large RT with negligible overheads. This effectively reduces WCT by using a large RT while keeping TC fixed. To further reduce TC, we allocate different resources (i.e., iterations) to each expert to take advantage of their varying convergence speeds. This simple modification accelerates overall convergence. We interpret that fast convergence can be achieved by minimizing

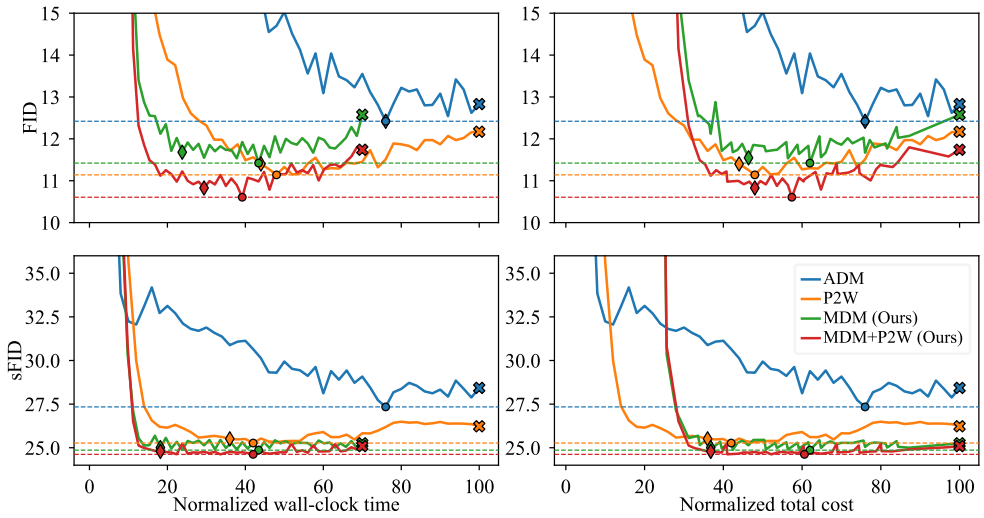

Figure 1: Quantitative evaluation for the normalized WCT (NWCT) and normalized TC (NTC) axes. The total WCT and TC of training the baseline for 500K iterations are set to 100%NWCT and 100%NTC, respectively. The best FID value for each model is denoted by '$\circ$' markers and its value as horizontal dotted lines. The termination point for full iteration is denoted as '$\times$' marker. We determine the model convergence point as the first point where the score difference between adjacent points is smaller than 0.1 for three consecutive sampling points, and at the same time, the score gap to the best FID value is smaller than 0.3 as marked with '$\diamondsuit$'.

negative interactions across different time intervals. Consequently, MDM can reduce both WCT and TC by early stopping rapidly converging experts.

We thoroughly investigate the advantages of the multi-expert approach by analyzing training patterns of diffusion models along with different time intervals. In our experiments, we apply MDM on several baseline models and demonstrate the effect of MDM in terms of efficiency (i.e., training time) and performance (i.e., generation quality). Overall, our method improves FID by 7.7% on average, including all datasets and baselines. Furthermore, MDM offers $\times$2.7 - 4.7 faster training and reduces TC by 24 - 53% to reach the best baseline score.

Related works

Denoising diffusion probabilistic model. Diffusion models (Ho, Jain, and Abbeel 2020; Dhariwal and Nichol 2021; Nichol and Dhariwal 2021; Song, Meng, and Ermon 2020) aim to generate data through a learned denoising process. Starting from a Gaussian noise x_T, they iteratively denoise x_t to x_{t-1} using a denoising autoencoder until obtaining a final image x_0. We discuss theoretical backgrounds in Appendix. ADM (Dhariwal and Nichol 2021) proposes the optimized network architecture and proves that the diffusion model can achieve higher image sample quality than state-of-the-art GANs in several benchmark datasets (Deng et al. 2009; Yu et al. 2015). We use ADM as our baseline to investigate the training dynamics of diffusion time steps in diffusion model dissection section.

Several works focus on the time steps of the diffusion model to improve sample quality. P2W (Choi et al. 2022) identifies that diffusion models learn coarse features in later time steps, rich contents at medium, and finally, remove remaining noise at early time steps. They propose a new weighting scheme for the training objective by assigning small weights to the unnecessary noise-removal stage while assigning higher weights to the others. Since the diffusion model exhibits an unstable denoising process nearly at $t = 0$ (infinite signal-to-noise ratio), both discrete and continuous time-based diffusion models (Ho, Jain, and Abbeel 2020; Song et al. 2021b,a) truncate the smallest time step (early-stopping denoising process before it reaches $t = 0$). Soft-truncation (Kim et al. 2021) sets the truncation hyperparameter to be randomly chosen at every optimization step to secure both NLL and FID. P2W and Soft-truncation improve the image quality by regularizing the model along time steps. However, based on our observation, they train the entire time steps at once, causing a negative influence among different time steps. Unlike these methods, our work identifies and then effectively eliminates such negative influences.

Also, several researchers have attempted to enhance the training efficiency of diffusion models. LDM (Rombach et al. 2022) seeks to reduce the parameter size of the model by reducing data resolution via autoencoders. Patch-Diffusion (Wang et al. 2023) proposes a data- and resource-efficient diffusion model by generating images in a patch-wise manner. Their focus is to improve model efficacy by changing from natural images to patch images in the data domain. These approaches are orthogonal to our method as they modify the domain of data distribution. Several methods (Balaji et al. 2022; Feng et al. 2023) utilize multi-

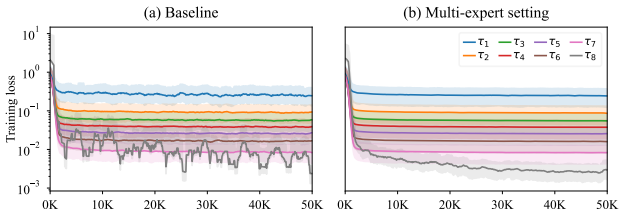

Figure 2: Training losses for (a) the baseline ADM (Dhariwal and Nichol 2021) and (b) the eight-expert settings. To improve visualization, we average five adjacent points to filter out the noise in the graph. The color-shaded area depicts the range between minimum and maximum values for adjacent points.

expert fine-tuning on a pre-trained text-to-image diffusion model to seamlessly reflect the text-conditional signal. Although they utilize a multi-expert strategy (Artetxe et al. 2021; Shazeer et al. 2017; Riquelme et al. 2021), their experts share the same pretrained model as initial points for fine-tuning. This approach limits training efficiency since those require a resource-intensive pretraining stage. Furthermore, they focus on conditional generation scenarios, which enhance text-and-image alignment through fine-tuning. We (1) do not deal with pre-trained models but the training efficiency of the model when training from scratch and (2) target unconditional diffusion models, which affects various applications. More importantly, our method works as an add-on module to these researches to enhance their training efficiency.

Multi-expert diffusion model

We propose the Multi-expert Diffusion Model (MDM) as an efficient training solution for diffusion models. Our approach centers around two objectives: (1) partitioning the model for independent training that better aligns with a large resource throughput (RT) and (2) improving the convergence speed of each expert to reduce the total cost (TC). Our investigation reveals distinct training patterns within the diffusion model, characterized by three groups of time steps exhibiting similar training patterns. Based on this observation, we introduce a training strategy that involves three experts, each responsible for training a specific group of three-time step intervals: τ_A, τ_B, τ_C.

Following Ho, Jain, and Abbeel (2020), we employ a denoising autoencoder to model the reverse process of the diffusion model. The learnable parameters $\theta(t)$ of MDM, given by a denoising autoencoder $f_{\theta(t)}(x_t, t)$, can be expressed as:

$$\theta(t) = \begin{cases} \theta_A, & t \in \tau_A, \\ \theta_B, & t \in \tau_B, \\ \theta_C, & t \in \tau_C. \end{cases} \tag{1}$$

τ_A, τ_B, and τ_C vary depending on the baseline model and the image resolution. The range of each interval determined for each experiment is specified in implementation details section. The experts in MDM (f_{θ_A}, f_{θ_B}, and f_{θ_C}) are trained independently within their designated time interval. For a fair comparison with the baseline, we initially set the maximum number of iterations $\mathbb{I}_e$ for each expert equally to $(|\tau_e|/T)\mathbb{I}_{\text{baseline}}$, $e \in \{A, B, C\}$. In this context, $|\tau_e|$ denotes the number of time steps within the interval τ_e, and $\mathbb{I}_{\text{baseline}}$ indicates the total iterations for training the baseline model. Then, we assign additional iterations to the expert with a relatively slower convergence while maintaining the sum of all $\mathbb{I}_e$ equal to $\mathbb{I}_{\text{baseline}}$. Each expert's architecture remains consistently the same.

Remarks on training efficiency. Our multi-expert approach offers two advantages: (1) utilizing a large RT with negligible overhead and (2) faster convergence to optimal performance for each expert. These two advantages reduce WCT and TC, respectively.

Firstly, training multiple experts independently empowers us to effectively reduce WCT by employing a large RT while minimizing additional overhead. Although the baseline model can be trained on multiple GPUs (or nodes), it is limited by practical resistance, such as finite batch size (which limits the maximum number of devices used) and communication overhead between devices. In contrast, our model has three independent experts, allowing us to increase RT more effectively than training the baseline with multiple nodes, with negligible practical resistance (see overhead analysis).

Secondly, our method trains each time interval independently, thereby focusing on each distinct training pattern. This mitigates the potential negative interactions among different time steps when training the entire time step simultaneously (Fig. 2). As a result, we consistently observe that any of the three experts in MDM reach optimal parameters faster than the baseline model. Furthermore, we assign additional iterations to the experts in τ_A and τ_C due to their slower convergence compared to the expert in τ_B. Our strategic allocation of training resources to the slower experts accelerates the overall convergence, reducing TC.

Overhead analysis. MDM utilizes three experts, resulting in three times the number of parameters compared to the baseline. However, the model capacity remains unchanged in terms of vRAM (or other equivalent limiting devices), serving as a true bottleneck in computing resources, as training and inferring each expert is independent of each other. The additional storage space required to store the parameters can be achieved with more affordable and sufficient options, such as flash memory. The slight increase in loading time required to transfer the model to vRAM is negligible compared to TC. Therefore, from a practical standpoint, the resource overhead associated with our method is manageable.

Diffusion model dissection

In this section, we delve into the detailed process of dissecting the time steps of diffusion models into three main groups for our MDM. We analyze the training patterns of the diffusion model and conclude that the standard method of training all time steps at once hinders fast convergence. We divide whole time steps into three groups for efficient training based on activation analysis.

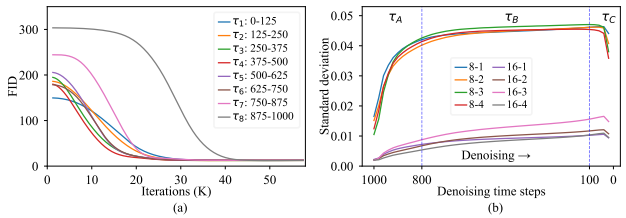

Figure 3: ADM time steps analysis. (a) Performance convergence in FID of eight experts for each time step range τ_i. (b) Concentration of attention weights during the denoising process. Each legend 'r-o' indicates o-th $r \times r$ attention layer.

Method	FID↓	sFID↓	Precision↑	Recall↑
Uniform	**12.42**	**27.34**	**0.5777**	0.6247
Importance	18.35	34.77	0.5532	**0.6355**

Table 1: Comparison between uniform and importance sampling strategies for time step sampling in the CIFAR-10 dataset. We train the baseline (Dhariwal and Nichol 2021) using uniform sampling and importance sampling and evaluate them after 300K iterations.

Training dynamics analysis

Each time step in the diffusion model is trained independently, and the loss scale diverges as t $\rightarrow$ 0 (Kim et al. 2021). We hypothesize that simultaneously training the entire time steps with varying loss scales (standard method) can hinder the training process. Therefore, we explore the impact of dividing the whole time steps into distinct groups and training each separately.

We divide the time steps into eight sub-intervals $(\tau_i)_{i=1}^{8}$ and assign an expert f_{θ_i}. In this experiment, each expert shares the same architecture of ADM (Dhariwal and Nichol 2021). The expert f_{θ_1} is responsible for generating the final clean image, while f_{θ_8} starts denoising from the noisy latent. We train each f_{θ_i} for $\tau_i = \{t|t \in (125(i-1), 125i]\}$ on CIFAR-10 dataset (Krizhevsky, Hinton et al. 2009). We evenly assign 62.5K iterations per expert, where a total of 500K iterations are used for both MDM and the baseline. We investigate the multi-expert setting in two aspects: convergence speed and training loss.

Convergence speed. To measure the convergence speed of each expert, we vary the iteration for the i-th expert while keeping the other experts fully trained (62.5K iterations). We calculate FID between the sampled 10K images and the 10K images of the CIFAR-10 validation set. Fig. 3(a) visualizes the FID values at each iteration for each expert. Interestingly, we observe different convergence speeds for each expert. The experts of middle time intervals show the rapid convergence at around 26K iterations. In contrast, the expert f_{θ_8} converges at around 45K iterations, demonstrating the slowest convergence speed. The second slowest expert becomes f_{θ_1}, which starts with a lower FID and converges at around 35K iterations.

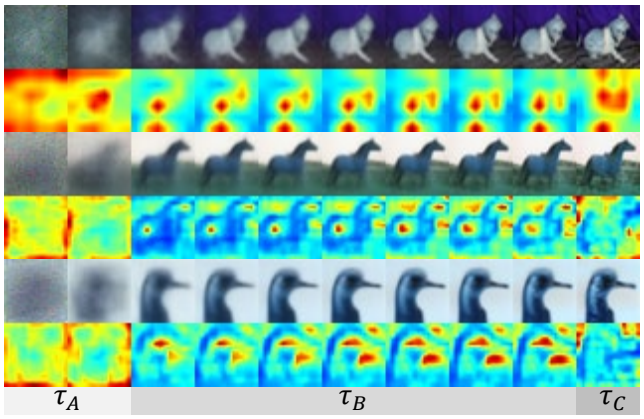

Figure 4: Visualization of the image sampling process and its attention layer weights. Odd rows depict the image prediction samples (x_0) obtained from DDIM (Song, Meng, and Ermon 2020) sampling. Even rows demonstrate the attention layer's activations, normalized by dividing them with the maximum value for improved visualization.

Training losses. We compare the training losses of the baseline (ADM) and the multi-expert setting (Fig. 2). We discover two key findings: (1) Training losses for each time step exhibit different loss lower-bound (Kim et al. 2021), and (2) the loss of the baseline presents fluctuations, especially in τ_8. Investigating each loss, the time range τ_1 produces a significantly higher loss ($\times 20$) than τ_8, thus largely affecting the parameter updates. However, as observed in Fig. 3(a), the time range τ_8 exhibits the slowest convergence, indicating a challenging stage to train. Despite its convergence challenges, the baseline cannot focus on τ_8 due to its low loss scale. In this regard, we recognize that importance sampling proposed by (Nichol and Dhariwal 2021) has a limited impact on performance improvement since it relies more on training time steps with higher losses without considering convergence trends. As a result, when we apply importance sampling to the baseline, we observe the performance degradation (Tab. 1).

The second observation indicates greater instability within each time interval of the baseline model compared to the multi-expert setting. Specifically, the loss for τ_8 depicts significant fluctuations. This result is consistent with the previous observation that τ_8 is the most challenging time interval to train. This phenomenon is significantly reduced in our multi-expert setting (Fig. 2(b)). Here, we speculate that training the entire time steps with a single model could result in sub-optimal performance due to adverse interactions among different time intervals.

Activation analysis for dissection

Our analysis demonstrated that multi-expert training can alleviate the negative impacts among time steps, ultimately improving training efficiency. Now, we arrive at a question: How should we partition the intervals for developing MDM?

We focus on the attention layers within the diffusion

Dataset	Method	Normalized WCT (%)			Normalized TC (%)		
		Converged	Best score	Baseline equiv.	Converged	Best score	Baseline equiv.
CIFAR-10	ADM	76.0	76.0		76.0	76.0	
	+MDM	23.8	43.4	**17.5**	46.4	62.0	**35.7**
	P2W	44.0	48.0		44.0	48.0	
	+MDM	29.4	39.2	**17.9**	48.0	57.4	**36.3**
ImageNet-32	Soft-trunc	-	86.0		-	86.0	
	+MDM	-	31.2	**18.4**	-	67.6	**42.0**

Table 2: Comparison of training time and resource requirements. NWCT (%) and NTC (%) are normalized by 500K training iterations for ADM and P2W, and 5.0M for Soft-truncation. 'Converged' indicates the convergence point of the model. 'Best score' refers to the first WCT and TC, where the model achieves the best FID. 'Baseline equiv.' denotes the first WCT and TC, surpassing the best FID of the baseline model. For ImageNet-32, we omit 'Converged' due to significant performance fluctuations in the baseline model.

Dataset	Method	FID↓	sFID↓	Prec.↑	Rec.↑
C-10	ADM	12.42	27.34	**0.5777**	0.6247
	+MDM	**11.42**	**24.86**	0.553	**0.6455**
	P2W	11.14	25.32	0.5405	0.6263
	+MDM	**10.61**	**24.74**	**0.5559**	**0.6569**
IN-32	Soft-trunc	9.18	4.74	**0.6018**	0.5966
	+MDM	**8.25**	**4.24**	0.5879	**0.6020**

Table 3: Quantitative evaluation. All metrics report the best FID score of each model. 'C-10' and 'IN-32' refer to CIFAR-10 and ImageNet-32, and 'Prec.' and 'Rec.' refer to precision and recall, respectively. For CIFAR-10, the best score within the same baseline (ADM or P2W) is in bold. The best score in all experiments is marked with an underline. MDM consistently improves FID, sFID, and Recall when applied to each baseline model.

model to derive distinct intervals of MDM. Previous studies (Caron et al. 2021; Tumanyan et al. 2022) have demonstrated that attention layers provide rich visual information, such as the semantic layout of scenes. Specifically, these attention layers selectively concentrate on structural properties among features (Caron et al. 2021). Motivated by this insight, we analyze the visual information captured by the diffusion model at each time step through attention weight analysis. For that, we leverage softmax weights within the attention layer:

$$Attention(Q, K, V) = softmax(QK^\top/\sqrt{d})V. \quad (2)$$

For each attention layer, we compute the average standard deviation of softmax weights for each image as follows.

$$\mathbb{E}_t\left[\sqrt{\text{VAR}_s(softmax_s(Q_{ct}K_{cs}/\sqrt{d_k}))}\right], \quad (3)$$

where subscripts follow Einstein's summation convention. A low standard deviation implies that the weight distribution is close to the uniform distribution (e.g., 0 if all values are $1/HW$, where H and W are the height and width of the attention map). Conversely, a high standard deviation represents weight concentration in a specific region (e.g., ∞ if the distribution follows the Dirac delta function). Fig. 4 demonstrates the attention layer's activations at each DDIM (Song, Meng, and Ermon 2020) sampling time step. Fig. 3(b) illustrates the average standard deviation of the attention layer's weights from 1K samples at resolutions of 8×8 and 16×16.

Herein, we identify two distinct transitions in terms of attention concentration. As depicted in Fig. 3(b), the first group τ_A consistently increases attention concentration. In this stage, the model generates the overall outline of the resulting image, as also reported in (Choi et al. 2022). In contrast, the second group τ_B shows minimal changes in attention concentration. The outline from the previous stage remains unchanged while incorporating additional details. Lastly, the third group, τ_C, shows a rapid decrease in concentration. This is because it removes an overall noise while adding natural high-frequency details (Balaji et al. 2022). These unique characteristics are used to determine three intervals of $[\tau_A, \tau_B, \tau_C]$, allowing each dedicated expert to handle distinct training patterns. Therefore, MDM assigns three experts for three distinct intervals derived in this study.

Experiments

Implementation details

Dataset. We use CIFAR-10 (Krizhevsky, Hinton et al. 2009) and ImageNet-32 dataset (Chrabaszcz, Loshchilov, and Hutter 2017) to train and evaluate our model. We conduct evaluations with validation sets comprised of 10K images for CIFAR-10 and 50K for ImageNet-32, respectively.

Architecture. We applied MDM on three baselines: ADM (Dhariwal and Nichol 2021), P2W (Choi et al. 2022) and Soft-truncation (Kim et al. 2021). ADM is the representative baseline model with widely used architectures for diffusion models. P2W is a recent training strategy tailored to diffusion models. Soft-truncation represents a universal training technique for score-based models, including both discrete and continuous time-based models. We show that our method can be combined with these baselines to improve the generation quality and reduce training resources. For ADM, we employ three attention layers at resolutions of 32, 16, and 8, with three residual blocks per resolution in Unet (Ronneberger, Fischer, and Brox 2015). The noise schedule is set as cosine. Our model has 128 channels with 32 channels per attention head and a dropout rate of 0.3. The batch size is 128, and the learning rate is 0.0001.

P2W is implemented on top of ADM. We set k=1, γ=1. For sampling, we apply DDIM (Song, Meng, and Ermon 2020) with 50 sampling steps. We set full-time step T to 1000. For the soft-truncation, we follow the identical configuration for ImageNet-32 training that uses DDPM++ (Song et al. 2021a) architecture. For ADM and P2W, we set $\tau_A = \{t|t \in (0.8T, T]\}$, $\tau_B = \{t|t \in (0.1T, 0.8T]\}$, and $\tau_C = \{t|t \in (0, 0.1T]\}$. For Soft-truncation we use $\tau_A = \{t|t \in (0.6T, T]\}$, $\tau_B = \{t|t \in (0.2T, 0.6T]\}$, and $\tau_C = \{t|t \in (0, 0.2T]\}$. We observe that τ_A, τ_B, and τ_C are consistent along with model and image resolution regardless of the training dataset. Furthermore, the attention concentration of the model (Fig.3(b)) depicts similar patterns even when we train the model using only 10% of total iterations. Thus, we can obtain time step intervals without significant overheads.

Sample quality metric. We use four metrics to assess the quality of the generated samples. FID measures the symmetric distance on the first raw and second central momentum between the two image distributions in the Inception-V3 (Szegedy et al. 2016) latent space. To capture structural relations between the data distributions more effectively than FID, we utilize sFID (Nash et al. 2021), which evaluates spatial features of Inception-V3. We also report precision and recall on the latent distribution of Inception-V3 (Kynkäänniemi et al. 2019) as FID cannot explicitly measure the distribution coverage of the generated samples.

Computational resources. We train our model with NVIDIA A6000 GPU. Training ADM and P2W on the CIFAR-10 dataset with a batch size of 128 for 500K iteration takes 270 hours. Soft-truncation on the ImageNet-32 dataset with a batch size of 128 for 5.0M iterations requires 462 hours.

Time and resource efficiency evaluation

We evaluate the practical aspects of different models by comparing their training time and resource requirements. Tab. 2 reports WCT and TC at three key points: (1) model convergence, (2) the point of achieving the best FID, and (3) surpassing the baseline model. We set the WCT and TC of the baseline model to 100%NWCT (normalized WCT) and 100%NTC (normalized TC), respectively. We consider the model to be converged when the FID difference between consecutive points is less than 0.1 for three consecutive sampling points. Simultaneously, the FID gap to the best value should be smaller than 0.3 in the CIFAR-10 dataset. ADM converges at 76.0%NWCT, while MDM on ADM converges about 3.2 times faster (23.8%NWCT). Similarly, P2W converges at 44.0%NWCT, whereas MDM on P2W converges at 29.4%NWCT, meaning 1.5 times faster.

We also identify when our MDM reaches its best FID and the best baseline FID score. Surprisingly, MDM-equipped baselines attain the best baseline score at an average of 17.9%NWCT and 38.0%NTC, being up to 4.7 times faster. Then, MDM reaches its best performance at an average of 37.9%NWCT and 62.3%NTC, still less than the baseline best score requirements. The result is visualized in Fig. 1. In conclusion, MDM effectively reduces training time and resources because of (1) higher RT with negligible computational overhead and (2) faster convergence of each expert.

τ_A	$\tau_B \cup \tau_C$	FID↓	sFID↓	Prec.↑	Rec.↑
ADM	ADM	12.42	27.34	**0.5777**	0.6247
MDM	ADM	**11.04**	**25.26**	0.5604	**0.6485**

Table 4: Ablation results on different model combinations along time steps in the CIFAR-10 dataset. 'Prec.' and 'Rec.' refer to precision and recall, respectively. Each experiment uses ADM for $\tau_B \cup \tau_C$ and only differs in the model for τ_A.

Quality evaluation

Tab. 3 presents the quality evaluation results, reporting the minimum FID achieved by each model. Applying MDM consistently improves performance across all baselines. Notably, our approach demonstrates a significant improvement in sFID and recall compared to other metrics. To identify which local expert contributes to our model to cover more diverse structures, we conduct a simple case study. As in Tab. 4, we compare the original ADM with a partially modified ADM where f_{θ_A} of MDM is exclusively applied for time interval τ_A. This investigation shows that f_{θ_A} significantly improves sFID and recall compared to the baseline. This is because the time steps τ_A play a pivotal role in shaping the overall outline, and our independent training strategy allows f_{θ_A} to generate diverse structures without negative impact from other time intervals.

Although we can manipulate precision-recall trade-off via guidance methods (Dhariwal and Nichol 2021; Ho and Salimans 2021) for the diffusion model, increasing recall is known to be a more challenging problem (the guidance can improve precision by sacrificing recall while the opposite is not yet available). In this view, we can conclude that MDM is capable of capturing diverse structures that lead to notable advantages in both sFID and recall.

Conclusion

This paper introduces a multi-expert diffusion model (MDM) as an efficient approach for training diffusion models. MDM capitalizes on the time-independent training nature of the diffusion model. Specifically, we carefully select three-time intervals according to activation analysis and assign a dedicated expert to each interval. Three experts of our model are trained independently on their respective time step groups. This approach allows us to increase resource throughput while minimizing the computational overhead, which effectively reduces the wall-clock time required for training full iterations. Furthermore, our multi-expert strategy enables each expert to focus solely on each designated time step without any negative impacts from other time ranges. This improves overall convergence speed and leads to a significant reduction in the total cost of training the diffusion model. As a result, our model reduces total costs by 24 - 53% and training time by 63 - 79% compared to the baselines, all while achieving the improvement in the average FID by 7.7% over all datasets.

References

Artetxe, M.; Bhosale, S.; Goyal, N.; Mihaylov, T.; Ott, M.; Shleifer, S.; Lin, X. V.; Du, J.; Iyer, S.; Pasunuru, R.; et al. 2021. Efficient large scale language modeling with mixtures of experts. *arXiv preprint arXiv:2112.10684*.

Balaji, Y.; Nah, S.; Huang, X.; Vahdat, A.; Song, J.; Kreis, K.; Aittala, M.; Aila, T.; Laine, S.; Catanzaro, B.; et al. 2022. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*.

Caron, M.; Touvron, H.; Misra, I.; Jégou, H.; Mairal, J.; Bojanowski, P.; and Joulin, A. 2021. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, 9650–9660.

Choi, J.; Lee, J.; Shin, C.; Kim, S.; Kim, H.; and Yoon, S. 2022. Perception prioritized training of diffusion models. 2022 IEEE. In *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11462–11471.

Chrabaszcz, P.; Loshchilov, I.; and Hutter, F. 2017. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Dhariwal, P.; and Nichol, A. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34: 8780–8794.

Feng, Z.; Zhang, Z.; Yu, X.; Fang, Y.; Li, L.; Chen, X.; Lu, Y.; Liu, J.; Yin, W.; Feng, S.; Sun, Y.; Chen, L.; Tian, H.; Wu, H.; and Wang, H. 2023. ERNIE-ViLG 2.0: Improving Text-to-Image Diffusion Model with Knowledge-Enhanced Mixture-of-Denoising-Experts. In *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative Adversarial Nets. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33: 6840–6851.

Ho, J.; and Salimans, T. 2021. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*.

Karras, T.; Aittala, M.; Aila, T.; and Laine, S. 2022. Elucidating the Design Space of Diffusion-Based Generative Models. In *Proc. NeurIPS*.

Kim, D.; Shin, S.; Song, K.; Kang, W.; and Moon, I.-C. 2021. Soft truncation: A universal training technique of score-based diffusion model for high precision score estimation. *arXiv preprint arXiv:2106.05527*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images. *Technical Report TR-2009*.

Kynkäänniemi, T.; Karras, T.; Laine, S.; Lehtinen, J.; and Aila, T. 2019. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32.

Li, H.; Yang, Y.; Chang, M.; Chen, S.; Feng, H.; Xu, Z.; Li, Q.; and Chen, Y. 2022. Srdiff: Single image super-resolution with diffusion probabilistic models. *Neurocomputing*, 479: 47–59.

Lugmayr, A.; Danelljan, M.; Romero, A.; Yu, F.; Timofte, R.; and Van Gool, L. 2022. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11461–11471.

Nash, C.; Menick, J.; Dieleman, S.; and Battaglia, P. W. 2021. Generating images with sparse representations. *arXiv preprint arXiv:2103.03841*.

Nichol, A. Q.; and Dhariwal, P. 2021. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, 8162–8171. PMLR.

Nichol, A. Q.; Dhariwal, P.; Ramesh, A.; Shyam, P.; Mishkin, P.; Mcgrew, B.; Sutskever, I.; and Chen, M. 2022. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. In *International Conference on Machine Learning*, 16784–16804. PMLR.

Riquelme, C.; Puigcerver, J.; Mustafa, B.; Neumann, M.; Jenatton, R.; Susano Pinto, A.; Keysers, D.; and Houlsby, N. 2021. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34: 8583–8595.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10684–10695.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, 234–241. Springer.

Saharia, C.; Chan, W.; Chang, H.; Lee, C.; Ho, J.; Salimans, T.; Fleet, D.; and Norouzi, M. 2022. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, 1–10.

Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; and Dean, J. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Shi, S.; Wang, Q.; and Chu, X. 2018. Performance Modeling and Evaluation of Distributed Deep Learning Frameworks on GPUs. In *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, 949–957.

Song, J.; Meng, C.; and Ermon, S. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Song, Y.; Durkan, C.; Murray, I.; and Ermon, S. 2021a. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34: 1415–1428.

Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2021b. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

Tumanyan, N.; Bar-Tal, O.; Bagon, S.; and Dekel, T. 2022. Splicing vit features for semantic appearance transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10748–10757.

Wang, Z.; Jiang, Y.; Zheng, H.; Wang, P.; He, P.; Wang, Z.; Chen, W.; and Zhou, M. 2023. Patch Diffusion: Faster and More Data-Efficient Training of Diffusion Models. *arXiv preprint arXiv:2304.12526*.

Wu, C.; Wu, F.; Lyu, L.; Huang, Y.; and Xie, X. 2022. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1): 2032.

Yu, F.; Zhang, Y.; Song, S.; Seff, A.; and Xiao, J. 2015. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv preprint arXiv:1506.03365*.

Appendix

In this document, we provide backgrounds of DDPM (Ho, Jain, and Abbeel 2020) and DDIM (Song, Meng, and Ermon 2020) and qualitative results of our proposed method.

Background

Denoising Diffusion Probabilistic Model (DDPM)

DDPM is a latent-variable generative model that gradually transforms a noise distribution into a data distribution $x_0 \sim q(x_0)$ (Ho, Jain, and Abbeel 2020). DDPM consists of a forward process q that iteratively adds a noise on the data distribution, and a reverse process p that iteratively denoises a noise distribution toward a final data distribution. The forward process adds a Gaussian noise to x_t using a Markov process according to a variance schedule $\{\beta_t\}_{t=1}^T$:

$$q(x_{1:T}|x_0) \coloneqq \prod_{t=1}^{T} q(x_t|x_{t-1}), \qquad q(x_t|x_{t-1}) \coloneqq \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{4}$$

Ho et al. state that it is possible to sample x_t from x_0 directly, using the notation $\alpha_t \coloneqq 1 - \beta_t$ and $\bar{\alpha}_t \coloneqq \prod_{s=0}^{t} \alpha_s$:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}) = \sqrt{\bar{\alpha}_t}x_0 + \epsilon\sqrt{1-\bar{\alpha}_t},\ \epsilon \sim \mathcal{N}(0, \mathbf{I}) \tag{5}$$

Using Bayes theorem, posterior $q(x_{t-1}|x_t, x_0)$ is also a Gaussian distribution with mean $\tilde{\mu}_t(x_t, x_0)$ and variance $\tilde{\beta}_t$:

$$\begin{gathered} q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I), \\ \text{where} \quad \tilde{\mu}_t(x_t, x_0) \coloneqq \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \quad \text{and} \quad \tilde{\beta}_t \coloneqq \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \end{gathered} \tag{6}$$

With sufficiently large T and a well defined β_t, the latent x_T becomes nearly an isotropic Gaussian distribution. Assuming this, to sample from the data distribution $q(x_0)$, we can first sample from an isotropic Gaussian distribution and then iteratively apply $q(x_{t-1}|x_t)$ to obtain x_0. However, $q(x_{t-1}|x_t)$ depends on the entire data distribution so it is hard to exactly compute when the data distribution is unknown. As a result, we train a neural network to predict a mean μ_θ and a diagonal covariance matrix Σ_θ:

$$p_\theta(x_{0:T}) \coloneqq p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t), \qquad p_\theta(x_{t-1}|x_t) \coloneqq \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{7}$$

The network is trained by optimizing the usual variational bound on negative log likelihood, L_{vlb}:

$$L_{\text{vlb}} \coloneqq L_0 + L_1 + ... + L_{T-1} + L_T \tag{8}$$

$$L_0 \coloneqq -\log p_\theta(x_0|x_1) \tag{9}$$

$$L_{t-1} \coloneqq D_{KL}(q(x_{t-1}|x_t, x_0) \,||\, p_\theta(x_{t-1}|x_t)) \tag{10}$$

$$L_T \coloneqq D_{KL}(q(x_T|x_0) \,||\, p(x_T)) \tag{11}$$

Ho et al. identify that training the model to predict ϵ in Eq. 5 improves sample quality than directly predicting $\mu_\theta(x_t, t)$. Therefore, L_{vlb} is simplified to:

$$L_{\text{simple}} = E_{t,x_0,\epsilon}\left[||\epsilon - \epsilon_\theta(x_t, t)||^2\right] \tag{12}$$

When the training is done, we can sample from the data distribution by inserting the predicted $\epsilon_\theta(x_t, t)$ to the equation:

$$\boldsymbol{x}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon_\theta\left(\boldsymbol{x}_t\right)\right) + \sigma_t \boldsymbol{z}_t, \tag{13}$$

where $\boldsymbol{z}_t \sim \mathcal{N}(0, \mathbf{I})$ and σ_t^2 is a variance which is set to $\sigma_t^2 = \beta_t$.

DDPM shows a powerful performance on image generation but is has a severe drawback of significantly slow sampling speed. To sample one image, it should feedforward a neural network for each denoising step, total T times. DDIM (Song, Meng, and Ermon 2020) accelerates the sampling speed of DDPM.

Denoising Diffusion Implicit Model (DDIM)

DDIM generalizes DDPM as a class of non-Markovian diffusion processes (Song, Meng, and Ermon 2020):

$$q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}(\sqrt{\alpha_{t-1}}\boldsymbol{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \frac{\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_0}{\sqrt{1-\alpha_t}}, \sigma_t^2 \boldsymbol{I}) \tag{14}$$

Consequently, the reverse process becomes

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\epsilon_t\left(\boldsymbol{x}_t\right)}{\sqrt{\alpha_t}} \right)}_{\text{"predicted } \boldsymbol{x}_0 \text{"}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_t\left(\boldsymbol{x}_t\right)}_{\text{"direction pointing to } \boldsymbol{x}_t \text{"}} + \underbrace{\sigma_t \boldsymbol{z}_t}_{\text{random noise}} \tag{15}$$

When $\sigma_t = \sqrt{\left(1-\alpha_{t-1}\right)/\left(1-\alpha_t\right)}\sqrt{1-\alpha_t/\alpha_{t-1}}$ for all t, the forward process becomes Markovian which means that the reverse process becomes a DDPM. When $\sigma_t = 0$, the forward process becomes deterministic and produces high quality samples much faster.

Qualitative results

We provide qualitative results of MDM applied on ADM (Dhariwal and Nichol 2021), P2W (Choi et al. 2022) and Soft-truncation (Kim et al. 2021).

Figure 5: Image samples of MDM+ADM trained on CIFAR-10 dataset.

Figure 6: Image samples of MDM+P2W trained on CIFAR-10 dataset.

Figure 7: Image samples of MDM+Soft-truncation trained on ImageNet-32 dataset.