MD-DiT: Step-aware Mixture-of-Depths for Efficient Diffusion Transformers

Mingzhu Shen[1†] **Pengtao Chen**[2†] **Peng Ye**[34*] **Guoxuan Xia**[1]
Tao Chen[2] **Christos-Savvas Bouganis**[1] **Yiren Zhao**[1]
[1]Imperial College London [2]Fudan University [3]Chinese University of Hong Kong
[4]Shanghai AI Laboratory `m.shen23@imperial.ac.uk, pengt.chen@gmail.com`

Abstract

Diffusion models (DMs) excel in vision generation tasks such as Text-to-Image but face high computational demands due to their large timestep dimensions. While reducing the number of timesteps has been the primary focus of previous studies, our research aims to optimize DM inference efficiency by reconfiguring the model architecture, particularly for diffusion transformers (DiT). Drawing inspiration from mixture-of-depth (MD) models, we account for the *computational asymmetry* across different timesteps, acknowledging that each computational block contributes differently at each time step. This observation leads us to explore strategies to bypass certain computational blocks (block skipping) or reuse the results from previous timesteps (block caching). To this end, We introduce MD-DiT, a unified framework that optimizes diffusion transformers by integrating block skipping and caching through gradient-free search, allowing the model to select blocks at varying timesteps for improved inference efficiency. Our findings demonstrate a 20% reduction in computational cost for a 4-step Latent Consistency Model (LCM) and a 59% reduction in a 40-step setup. MD-DiT exceeds the performance of state-of-the-art *training-free* methods, such as DeepCache, TGATE, and T-Stitch.

1 Introduction

Diffusion models have achieved remarkable success in a wide array of text-to-image generation tasks, including GLIDE [23], Imagen [32], DALL·E [28], and Stable Diffusion [26, 30, 34]. Recent research has focused on developing efficient noise schedulers [11, 13, 19, 20] that can significantly reduce the number of timesteps, such as reducing the timesteps from 1000 to just 10 steps. Further advancements have also enabled diffusion models to generate reasonable results even in a single timestep through distillation techniques, such as the consistency loss [21, 38] and adversarial distillation [33, 34]. While the majority of diffusion models are based on the Convolutional UNet [31] architecture, more recent models have transitioned to transformer-based models, which offered a better scalability [2, 3, 25]. Besides, numerous studies have targeted efficiency improvements through these optimizations, including quantization [35], pruning [44], novel model design [14, 49] and caching strategies [16, 22, 41, 47]. However, the majority of these methods concentrate on UNet-based diffusion models instead of transformer-based architectures.

The recent work OMS-DPM [17] and T-Stitch [24] introduced a novel approach where different models are assigned at various timesteps for the sampling process, as depicted in Figure 1 for transformer based diffusion. They demonstrate that models with different capabilities can be applied at different timesteps without compromising performance. In contrast, DDSM [43] utilizes a single model trained with adjustable widths, though this approach incurs significant training costs to obtain subnets. In contrast to previous studies, we adopt an entirely different perspective on this problem – *we view a diffusion transformer as a mixture-of-depth model*, where at each timestep, only

*Corresponding Author. †Equal Contribution.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

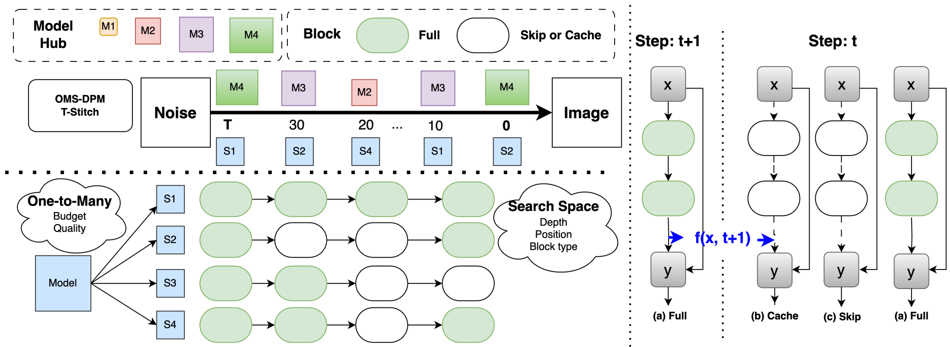


Figure 1: Existing works OMS-DPM [17] and T-Stitch [24] propose a mixture of models downloaded from the model hub and use them at different timesteps in the sampling process. We propose to use **One Model** to produce different subnets (such as S2, S3) by skipping or caching certain blocks, allowing for varying depths tailored to different computational requirements.

a subnetwork with selected depth configuration is activated and running. As depicted in Figure 1, the potential subnetwork in operation can vary significantly across different timesteps $(0...T)$. The only remaining question is to identify what is the optimal subnetwork to employ at each timestep.

To this end, we introduce a *training-free* search mechanism that generates distinct models with different depths for each timestep, in a similar spirit to once-for-all networks [1, 46]. Consequently, we name our framework Mixture-of-Depths Diffusion Transformers (MD-DiT). As shown at the bottom of Figure 1, we adeptly tailor **many Subnets** from **one parent Model** (**One-to-Many**), providing a range of models with diverse generation capabilities and runtime characteristics. In this paper, we introduce two distinct strategies to vary depths: skipping and caching. Skipping blocks offers a direct computation reduction but can lead to significant divergence from the original computations. To mitigate this, we draw inspiration from recently proposed caching techniques [16, 22, 41, 47]. By caching and reusing previous block computations, we can approximate the current block with minimal additional computation, effectively offering a 'free lunch' that significantly boosts generation performance. We then integrate skipping, caching, and full computation into a unified block definition, establishing the MD-DiT framework. MD-DiT can choose skipping, caching, or full computation for each block in DMs. The resulting search space is vast, with a model of N blocks leading to a search space of 3^N. Thus, we then employ a gradient-free search algorithm in the search space. This strategy outperforms manually preset patterns and achieves superior outcomes.

To summarize, the contributions of our paper are threefold: (1) We introduce MD-DiT, a **one-to-many** unified framework that realizes a **mixture-of-depths** across different timesteps via the incorporation of block skipping and caching techniques. (2) Our research investigates various search space trimming guidelines such as depth allocation in each timestep, offering valuable insights into the design principles of accelerating diffusion transformers. Furthermore, we can identify a more compact model that further enhances efficiency by employing efficient gradient-free optimization methods. (3) Through extensive experiments, we have successfully compressed the LCM-4Step [3] model with a 20% reduction in Multiple-Accumulate Operations (MACs). This achievement is further amplified in a 40-step setting, where we have accomplished a 59% reduction. These results surpass the performance of existing state-of-the-art **training-free** acceleration methods.

2 Method

2.1 Step-aware Mixture-of-Depths

Block Definition. From a block perspective, the full computation of a block is defined as $y_i^t = x_i^t + f(x_i^t)$, where y_i^t and x_i^t represent the output and input of the block, respectively. The most direct way to skip a block's computation is to simply omit it, as shown in Figure 1. However, this can result in significant degradation of the generated output. Inspired by recent cache-based methods [16, 22], we propose caching the incremental change of a block, termed $f(x_i^{t+1})$, which can be considered a "free lunch" for improving performance as it does not introduce additional computation.

This approach allows us to balance computation and generation quality according to different scenarios. Specifically, we define three possible strategies, where λ_i^t refers to the i-th block at timestep t: (1) **Skipping** ($\lambda_i^t = 0$): This completely skips the block, but may result in a discrepancy between the original and current results. (2) **Caching** ($\lambda_i^t = 1$): The cached feature map from the previous generation step is used as an approximation, offering a cost-free solution. However, for certain critical blocks, this may require compensation to avoid large deviations. (3) **Full** ($\lambda_i^t = 2$): The block performs its full computation, which ensures optimal generative quality but incurs the highest computational cost. This framework allows for a flexible and efficient search across these three options. More details can be found in the Appendix.

The MD-DiT Framework. Existing work with Mixture-of-Depth models dynamically assigns different depths to *different tokens* [29] such as using a router based on the input token x to choose to execute at different depths. In contrast, we intend to assign varying depths to *different timesteps*. Our depth allocation varies only across the timestep dimensions under different computation and generation quality constraints. As shown in Figure 1, given a model m with an maximum depth D, for each timestep $t \leq T$, we assign a subnet (*e.g.* **S1, S2, S3, S4** in Figure 1), that are generated from m with a depth $D^t \leq D$. We can effectively construct these subnets by assigning different λ_i^t values to each computation block, as explained in the previous **Block Definition** section. Consequently, we achieve a **one-to-many** generation: based on a single parent model, we can customize different sub-models for any given computational budget. For a given budget constraint c, the optimization objective is to minimize the loss function $\mathcal{L}$ to improve the generative quality by searching for the optimal λ_i^t values for each block. Therefore, the search process can be defined as:

$$\min_{\lambda_0^T, \dots, \lambda_D^0} \quad \mathcal{L}(m, \{\lambda_0^T, ..., \lambda_i^t, ..., \lambda_D^0\}), \tag{1}$$

In the formulation in Equation (1), the overall search space size is $3^{T\times D}$. For instance, considering a 28-block transformer like Pixart-Alpha [3] with a total of $T = 20$ timesteps, the search space becomes $3^{20\times 28}$, which is excessively large. Therefore, it is essential to consider the search efficiency.

2.2 Gradient-Free Search

Search Space Design. The search problem can be divided into two primary components: the Search Space Design and the corresponding Search Algorithm. For the former, we can reduce the search space size by employing strategic elimination of certain search dimensions. Extensive research [27, 39] has already established various principles for designing efficient search spaces, particularly in the domain of classification tasks. By carefully eliminating non-essential search elements or dimensions, the search space can be reduced by several orders of magnitude, which in turn also allows for a more focused allocation of search resources. Further discussions on these topics are detailed in the Appendix.

Search Algorithm. Regarding the search algorithm, the issue stems from its combinatorial optimization space, necessitating a fine-graned search. More specifically, given a fixed computational budget, the task involves strategically assigning three discrete values of 0, 1, or 2 to each block, to achieve a more refined compressed model. Drawing inspiration from the latest research [12, 18], we propose to adopt a gradient-free optimization technique to identify the most beneficial blocks to either skip or cache. In particular, we employ the Covariance Matrix Adaptive Evolution Strategy (CMA-ES) [7] to conduct this search, leveraging its efficacy in navigating the complex landscape of potential solutions.

3 Experiment

Models, Datasets, and Evaluation Metrics. As we focus on diffusion transformers, we choose DiT-XL [25] and Pixart-Alpha [3], and use their LCM distilled versions [21]. To align with prior research [22, 36, 44], we select three datasets for evaluation: PartiPrompts [45], containing 1.63K prompts, MSCOCO-2017, which includes 5K prompts and images and ImageNet [5] with 5K images. Our assessments are based on the Fréchet Inception Distance (FID) [10] metric and the Clip Score, utilizing the ViT-g/14 architecture as detailed in [9]. To evaluate efficiency, we use Calflops [42] to count Multiple-Accumulate Operations (MACs) and the latency per sample on the Nvidia 3090. To benchmark against state-of-the-art (SOTA) methods, we have faithfully implemented several **training-Free** acceleration baselines, including FasterDiffusion [16], DeepCache [22], TGATE [47], OMS-DPM [17], DDSM [43] and T-Stitch [24].

Table 1: Based on LCM [21] Pixart-Alpha [3], we utilize prompts in PartiPrompt and MSCOCO-2017 5K validation set to generate images at the resolution of 1024. We search for the model with computation that is comparable to or surpasses that of established baseline methods TGATE [47]. For latency, we only count the time to inference transformers, not the whole pipeline.

Method	PartiPrompts			COCO2017		
	MACs↓	Reduction ↑	CLIP Score ↑	Latency (ms)↓	FID ↓	CLIP Score ↑
LCM - 4 steps	8.57T	-	29.67	880	40.43	29.99
TGATE (n=2)	7.94T	7.3%	29.55	820 (1.07×)	42.04	29.91
Ours	7.65T	**10.7%**	29.38	**780** (1.13×)	**40.08**	29.59
TGATE (n=1)	7.62T	11%	28.68	790 (1.11×)	44.19	29.07
Ours	6.84T	**20.2%**	**28.71**	**720** (1.22×)	**43.35**	28.99

Table 2: Compared with other existing search methods OMS-DPM [17] and DDSM [11], we focus on these aspects: the dimensionality of the search space, the underlying architecture, and the ratio by which the search space is effectively reduced. Additionally, we assess the computational efficiency of our method by measuring the search cost for a single model in terms of GPU hours required. UNet in OMS-DPM and DDSM refers to SD1.4 [30] and ADM [6] respectively.

Methods	Search dimension	Cost (GPU hours)	Reduction	Training
OMS-DPM [17]	{Models, Timestep}	≈ 1000	50% (T=24)	×
DDSM [43]	Width	≈ 1000	40% (T=50)	✓
T-Stitch [24]	Models	0	41% (T=100)	×
Ours	{Depth, Position, type}	≈ 1/10 (T=4/40)	**59% (T=40), 20% (T=4)**	×

3.1 Comparation with SOTA methods

Comparison under LCM Settings. By refining the search space, we can enhance search efficiency by a large margin. As shown in Table 1, TGATE's impact on LCM results in a modest acceleration of only 11%. However, we achieve a 20% reduction in MACs with improved Clip Score and FID metrics. One reason is that compressing LCM models is inherently more difficult and can lead to a substantial drop in generative performance. TGATE's significant advantage lies in its elimination of classifier-free guidance, simplifying the process by condensing two batches into one. In contrast, LCM is designed without incorporating classifier-free guidance.

Comparison with other Search Methods. In Table 2, we compare our method with OMS-DPM [17] and DDSM [43] in terms of search dimensions, cost, and computational efficiency. OMS-DPM has high search costs due to the need for a comprehensive training dataset for its predictor model. DDSM requires training a supernet and performing FID-based searches, which demands sampling thousands of images. While T-Stitch lowers search costs, it has GPU memory limitations and struggles with smaller timesteps like $T = 4$. In contrast, our one-to-many framework uses a gradient-free search, achieving optimal settings in under one GPU hour at $T = 4$, and remains under 10 hours at $T = 40$, offering a 100-fold cost reduction compared to OMS-DPM.

Table 3: In our comparison with T-Stitch [24], we follow the same settings using the 5K ImageNet dataset, we set the timestep to $T = 100$ and employed the DDIM [37] scheduler. The observed difference in latency is attributed to our use of the DiT in Diffusers [40], which incorporates FlashAttention [4] by default, resulting in significantly faster inference speed.

Model	MACs (Tera)↓	Latency (s)↓	FID↓
DiT-XL [25]	11.45	10.7	9.08
DiT-S [25]	0.55 (↓ 95%)	0.95 (11.3×)	29.46(+20.38)
T-Stitch [24]	6.0 ((↓ 47%)	5.15 (2.1×)	9.65(+0.57)
DiT-XL-Flash [25]	11.45	**4.2**	**8.94**
Ours	5.0 (↓ **56**%)	2.15 (1.95×)	**9.04** (+0.1)

In Table 3, we contrast our approach with T-Stitch [24], which merges two distinct models operating at different time steps. Unlike T-Stitch, our methodology can generate multiple models from a single base model, each with varying computational demands. Furthermore, T-Stitch utilizes a small, manually selected model ratio, substituting the larger model during the initial phase (proximal to

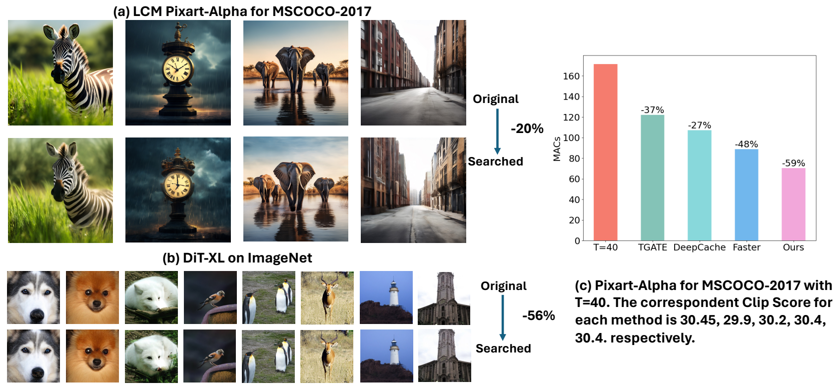


Figure 2: Visualization outcomes (a) (b) for LCM Pixart-Alpha on the MSCOCO-2017 dataset and DiT-XL on ImageNet are as follows: for MSCOCO-2017 generated images, the majority of the structural elements are preserved, albeit with some regions appearing slightly blurry. Comparison with other training-free methods is also included in (c). -20% means 20% MACs reduction.

noise), which results in nearly a 47% reduction in MACs at the cost of a 0.57 decrease in (FID) score. In contrast, our approach can transform a single model into various specialized models, achieving significantly improved outcomes with a 56% reduction in MACs and a negligible impact on the FID score.

Comparison with Caching Methods. As depicted in Figure 2, we implement Faster [16] and DeepCache [22] within our framework, as they originally do not support transformer architectures. We have also searched for the optimal cache blocks but with fixed depth patterns for these methods. Our framework's ability to integrate these three distinct methods capitalizes on their strengths, resulting in enhanced performance over manually designed patterns. Furthermore, as visualized in Figure 2, even with a large acceleration ratio, our approach maintains most of the structural integrity compared to the original uncompressed model, achieving a better quality-computational trade-off.

4 Related Work

Model Scheduling Methods. DeepCache [22], and FasterDiffusion [16] use caching to avoid redundant computations. Recent work [47] skips cross-attention in later stages, though many methods are untested on fewer-timestep models and not applicable to diffusion transformers. OMS-DPM [17], first propose a method where one of six models with varying computational demands can be randomly selected to execute a single timestep and even lead to better performance. T-Stitch is more straightforward which replaces large models with small models with a hyperparameter. Thus it can perform a grid search over the replace ratio. Instead of selecting from a pool of models, DDSM [43], advances the strategy by training a single, adaptable neural network and then determining the most suitable network width for each timestep. However, DDSM can impose significant additional training requirements, if one attempts to train on more extensive foundational models, such as SDXL [26] or Pixart-Alpha [2]. There is a risk of ending up with less optimally trained sub-networks due to the complexity and scale of the task. More related works can be found in Appendix.

5 Conclusions

In this paper, we introduce a novel one-to-many framework capable of accommodating Mixture-of-Depths across various timesteps. Our findings demonstrate superior performance in comparison to other training-free methods and offer insightful contributions to the field of efficient diffusion transformers.

Acknowledgements. Mingzhu Shen is funded by Imperial President's PhD Scholarships. Part of the work is supported by National Natural Science Foundation of China (No. 62071127, and 62101137), National Key Research and Development Program of China (No. 2022ZD0160101), Shanghai Natural Science Foundation (No. 23ZR1402900), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103). Part of the computations in this research were performed using the CFFF platform of Fudan University.

References

[1] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment, 2020.

[2] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-sigma: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. *arXiv preprint arXiv:2403.04692*, 2024.

[3] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, et al. Pixart-alpha: Fast training of diffusion transformer for photorealistic text-to-image synthesis. *arXiv preprint arXiv:2310.00426*, 2023.

[4] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[7] Nikolaus Hansen and Andreas Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of IEEE international conference on evolutionary computation*, pages 312–317. IEEE, 1996.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021.

[10] 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*, 30, 2017.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[12] Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic lora composition. *arXiv preprint arXiv:2307.13269*, 2023.

[13] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

[14] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. Bk-sdm: A lightweight, fast, and cheap version of stable diffusion. *arXiv preprint arXiv:2305.15798*, 1(2):3, 2023.

[15] Akio Kodaira, Chenfeng Xu, Toshiki Hazama, Takanori Yoshimoto, Kohei Ohno, Shogo Mitsuhori, Soichi Sugano, Hanying Cho, Zhijian Liu, and Kurt Keutzer. Streamdiffusion: A pipeline-level solution for real-time interactive generation. *arXiv preprint arXiv:2312.12491*, 2023.

[16] Senmao Li, Taihang Hu, Fahad Shahbaz Khan, Linxuan Li, Shiqi Yang, Yaxing Wang, Ming-Ming Cheng, and Jian Yang. Faster diffusion: Rethinking the role of unet encoder in diffusion models. *arXiv preprint arXiv:2312.09608*, 2023.

[17] Enshu Liu, Xuefei Ning, Zinan Lin, Huazhong Yang, and Yu Wang. Oms-dpm: Optimizing the model schedule for diffusion probabilistic models. In *International Conference on Machine Learning*, pages 21915–21936. PMLR, 2023.

[18] Jialin Liu, Antoine Moreau, Mike Preuss, Jeremy Rapin, Baptiste Roziere, Fabien Teytaud, and Olivier Teytaud. Versatile black-box optimization. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pages 620–628, 2020.

[19] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[20] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

[21] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

[22] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. *arXiv preprint arXiv:2312.00858*, 2023.

[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] Zizheng Pan, Bohan Zhuang, De-An Huang, Weili Nie, Zhiding Yu, Chaowei Xiao, Jianfei Cai, and Anima Anandkumar. T-stitch: Accelerating sampling in pre-trained diffusion models with trajectory stitching. *arXiv preprint arXiv:2402.14167*, 2024.

[25] William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023.

[26] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[27] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces, 2020.

[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*, 1(2):3, 2022.

[29] David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*, 2024.

[30] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 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*, pages 234–241. Springer, 2015.

[32] 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*, 35:36479–36494, 2022.

[33] Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast high-resolution image synthesis with latent adversarial diffusion distillation, 2024.

[34] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation, 2023.

[35] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1972–1981, 2023.

[36] Andy Shih, Suneel Belkhale, Stefano Ermon, Dorsa Sadigh, and Nima Anari. Parallel sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

[37] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022.

[38] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models, 2023.

[39] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

[40] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022.

[41] Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, et al. Cache me if you can: Accelerating diffusion models through block caching. *arXiv preprint arXiv:2312.03209*, 2023.

[42] xiaoju ye. calflops: a flops and params calculate tool for neural networks in pytorch framework, 2023.

[43] Shuai Yang, Yukang Chen, Luozhou Wang, Shu Liu, and Yingcong Chen. Denoising diffusion step-aware models. *arXiv preprint arXiv:2310.03337*, 2023.

[44] Xingyi Yang, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Diffusion probabilistic model made slim. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22552–22562, 2023.

[45] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3):5, 2022.

[46] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks, 2018.

[47] Wentian Zhang, Haozhe Liu, Jinheng Xie, Francesco Faccio, Mike Zheng Shou, and Jürgen Schmidhuber. Cross-attention makes inference cumbersome in text-to-image diffusion models. *arXiv preprint arXiv:2404.02747*, 2024.

[48] Lin Zhao, Tianchen Zhao, Zinan Lin, Xuefei Ning, Guohao Dai, Huazhong Yang, and Yu Wang. Flasheval: Towards fast and accurate evaluation of text-to-image diffusion generative models. *arXiv preprint arXiv:2403.16379*, 2024.

[49] Yang Zhao, Yanwu Xu, Zhisheng Xiao, and Tingbo Hou. Mobilediffusion: Subsecond text-to-image generation on mobile devices. *arXiv preprint arXiv:2311.16567*, 2023.

(a) Original (b) [S]: B0 (c) [S]: B21 (d) [S]: B18-21 (e) [C]: B0 (f) [C]: B18-21

Figure 3: The generated results involve selective skipping and caching for various blocks with timesteps=20. Here, [S], [C], and B denote Skipping, Caching, and Block Numbers respectively. For skipping, the chosen block is omitted across all time steps. In caching, the first timestep uses full computations while all the rest timesteps use caching.

6 Appendix

6.1 Preliminary

When viewed within the temporal domain, diffusion models can be conceptualized as exceedingly deep transformers. Each timestep in the process adds to the depth, similar to stacking multiple layers in a deep neural network in different stages [8]. This iterative, step-by-step denoising process allows diffusion models to create detailed and complex generative outputs.

Original. Therefore, for every block in the diffusion transformer in the denoising process, given a timestep t, for a residual block with the function f with learnable parameters w_i at layer i, the output y_i^t is calculated by the input x_i^t:

$$y_i^t = x_i^t + f(x_i^t) \tag{2}$$

Skipping. One straightforward way to reduce computation for a block is skipping:

$$y_i^t \approx x_i^t, \tag{3}$$

As shown in Figure 3, the sensitivity varies when skipping different blocks. Skipping Block 0 results in a completely noisy generated image (Figure 3b). Conversely, skipping Block 21 has a negligible impact on the final results, still yielding reasonable images (Figure 3c). However, achieving further computational reduction by skipping more blocks (18-21) while maintaining satisfactory results remains a challenge (Figure 3d).

Caching. Building upon the progressive denoising noise process, recent advancements [16, 22] introduce a training-free acceleration technique utilizing cached feature maps from the preceding timestep to bypass calculations. Despite employing slightly outdated feature maps, this method yields comparable results with those of the original model. It can be defined as follows:

$$y_i^t \approx x_i^t + f(x_i^{t+1}), \tag{4}$$

As depicted in Figure 3, the straightforward application of cached feature maps significantly improves generation outcomes, transforming noisy images into meaningful ones (see Figure 3e–Figure 3f).

In the original DeepCache [22] and FasterDiffusion [16] models, they retain the output in their caching approach. In contrast, our method caches the incremental change or delta represented as $f(x_i^{t+1})$. This distinction arises from the underlying architectures and their respective motivations. Specifically, in the UNet architecture, the encoder's output is concatenated with the output from the middle stage and then passed to the final decoder stage. This design leverages the UNet's long-range shortcut connections, which are not present in transformer models. Furthermore, our approach is motivated by a desire for a more granular level of control in the search process. Rather than opting to bypass the computation for an entire branch, we aim to make more precise decisions on whether to skip the computation for each block. It can benefit more for models with fewer timesteps.

Unified Block Definition. From a block perspective, we can unify these methods with two hyperparameters α_i^t, and β_i^t to decide whether to skip, cache, or fully compute one block.

$$y_i^t \approx x_i^t + \alpha_i^t \times f(x_i^{t+1}) + \beta_i^t \times f(x_i^t) \tag{5}$$

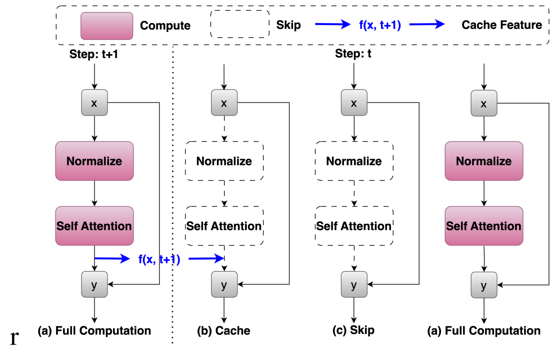
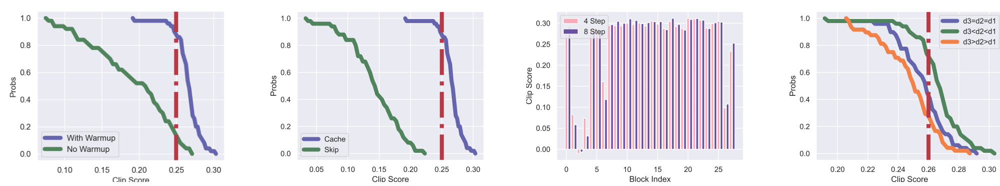


Figure 4: For every block, Self Attention, Cross Attention, and Feed Forward in the transformer architecture, the choice is 3 for Skipping, Caching, and Full Computation.



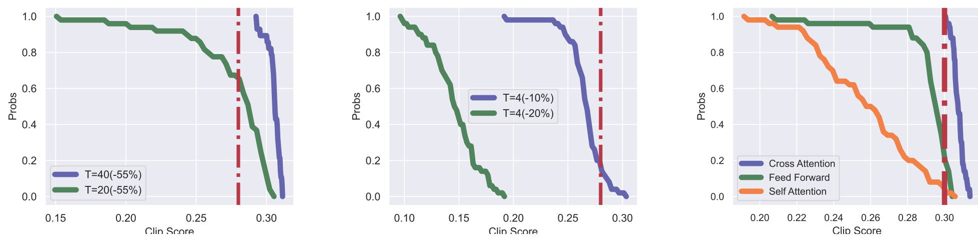
(a) First Step (b) Skip or Cache (c) Block Sensitivity (d) Depth Allocation

Figure 5: Clip Score is computed with FlashEval [48] dataset. Probs represent the percentage of subnets that surpass the corresponding Clip Score. For every search space, we random sample 50 subnets under the same computation budget. (a) We compare skip blocks in the first timestep or with full computation in the first timestep. (b) We compare skip or cache all the blocks. (c) A single block with the same block index is cached across all time steps, except the initial time step. (d) We first set the depth allocation in each timestep and sample different blocks.

With the definition in Equation (5), we have the following three scenarios. (1) When α_i^t and β_i^t are zero, it's equivalent to skipping a block, potentially leading to a discrepancy between the original and current results. (2) When only $\beta_i^t = 0$, this effectively provides caching: the cached feature map can serve as an approximation, offering a cost-free solution as it only requires caching the feature map from the previous generation timestep. However, for certain critical and sensitive blocks, relying on the current input is necessary to compensate for the large deviation effects. Therefore, (3) the block needs to fall back to full computation to improve the overall generation performance. In summary, by appropriately tuning the hyperparameters α_i^t, and β_i^t in each block, we can tailor the computation and generative quality to suit various scenarios. Thus we can only search for three options as shown in Figure 4 with **(1)** $\lambda_i^t = 0$ means $\alpha_i^t = 0$ and $\beta_i^t = 0$ for block skipping and **(2)** $\lambda_i^t = 1$ means $\alpha_i^t = 1$ and $\beta_i^t = 0$ for block caching. **(3)** $\lambda_i^t = 2$ means $\alpha_i^t = 1$ and $\beta_i^t = 0$ for full block computation.

6.2 Search Datasets and Metrics.

Choosing the appropriate datasets is essential for optimizing search efficiency. We select FlashEval [48], a small dataset with only 50 images, as it provides a quick and precise measure for image quality assessment. Given the gradient-free nature of our method, we have the flexibility to consider both distribution-level metrics like FID and per-sample metrics such as Clip Score, without being limited by differentiability requirements. However, FID's accuracy demands a substantial number of generated images, which can be time-consuming [17, 43]. Therefore, we prefer the Clip Score metric, which offers a per-sample evaluation and reflects text alignment. In terms of computational budgeting, we can precisely control the computation by strategically deciding which blocks to skip or cache. This approach enables the rapid identification of optimal model configurations in under a minute (4 steps), fitting well with our search framework. Moreover, for various downstream tasks, we can customize different small datasets to ensure efficient and precise feedback.



(a) Reduction (T=40, 20) (b) Reduction (T=4) (c) Block Type

Figure 6: Clip Score is computed with FlashEval [48] dataset. We random sample 50 subnets for each computation budget like 55%, 20%, and 10% for timestep 2, 20 and 40. We also evaluate impact of different block types.

6.3 Search Space Trimming

Applying a search algorithm naively would confront a prohibitively large search space. Consider the LCM-4 Step Pixart model as a demonstrative example. Given the block number is $28 \times 4 = 112$, the resultant search space is 3^{112}, which is impractically large for an efficient and effective search. We thus employ the following four search space trimming tricks to circumvent unnecessary explorations.

(1) All-active First Step. It is readily apparent that alterations to the initial timestep can be significantly magnified in subsequent timesteps, leading to substantial deviations. As shown in Figure 5a, we recommend leveraging the first timestep untouched with full computation to both reduce the search space and preserve the integrity of the generated image structure. The search space is reduced to 3^{84}.

(2) Skip or Cache. As shown in Figure 5b, we find that caching is generally better than skipping which makes sense as simply skipping can cause greater deviation from the full computation output and thus we can decrease the block choice from 3 to 2 and the space can be reduced to 2^{84}. We observe that caching generally outperforms skipping in low-timestep configurations (e.g., (T < 10)), but for higher timesteps, skipping continues to serve as an effective means for computational reduction when T is large, as shown in Figure 2. So we conditionally apply this search space trimming trick when the timestep is low.

(3) Remove Sensitive Blocks. As shown in Figure 5c, we only cache 1 block and find that some of them lead to catastrophic degradation (like 1, 2, 3) and thus these should be eliminated from the search space and always be fully computed. We can observe that this can reduce the searched block number from 28 to 23. The search space size is 2^{69}.

(4) Prioritizing Later Timeteps. As illustrated in Figure 5d, the depth allocation configuration with $d1 < d2 < d3$ results in nearly 70% of subnets having a Clip Score higher than 0.27. In comparison, when the depths are equally allocated, $d1 = d2 = d3$, the percentage drops to approximately 40%. Conversely, an inverse allocation, $d1 > d2 > d3$, yields less than 30%. This indicates that allocating more computational resources to later timesteps significantly increases the probability of obtaining a model with a higher Clip Score.

All these four trimming techniques would have to be executed before the search occurs, and this provides a cost of like 3 hours and can be applied for different computation budgets in this timestep setting.

6.4 Architecture Analysis

In the following section, we try to give some insights on how to design diffusion transformers. By analyzing our search results, we can reveal insights into several design principles in diffusion transformers.

Is Caching Always Better Than Skipping? Although the complimentary cached feature map is cost-free, it should be carefully utilized to enhance results. As shown in Figure 2, Faster [16] is even better than DeepCache [22] in Clip Score with 20% less computation for 40 timesteps. One of the reasons is that the feature maps between adjacent timesteps are much closer. However, we argue that in LCM settings, caching is more likely to be better as shown in Figure 5b.

How Many Percentages can be Reduced for Different Timesteps. We randomly sampled many subnet settings at different percentages—10%, 20%, and 55%—under various timestep conditions, where T takes values of 2 20 and 40 respectively. As illustrated in Figure 6a, when $T = 40$, discarding more than 50% of the blocks remains quite robust compared to when $T = 20$. Moreover, at the lowest setting $T = 4$, even with a 20% block reduction, all subnets sampled result in greater performance drops below 0.2 Figure 6b. This demonstrates that the compression task becomes increasingly challenging with fewer timesteps. Existing compression techniques like caching and branching modify architecture at a coarser-grained level (such as dropping more than 20 blocks), whereas these low-timestep setups necessitate a finer-grained, block-level manipulation akin to our search approach.

Self-Attention is the most sensitive while Cross-Attention is the least. Inspired by TGATE [47], we've evaluated the sensitivity across various block types under a fixed computation budget aimed at a 10% compression rate. As shown in Figure 6c, cross-attention blocks emerged as the least sensitive, with the majority scoring above 0.30, whereas self-attention blocks were identified as the most sensitive, prone to significant performance degradation. Although our block definition simplifies the model by considering the three distinct blocks as a single entity—reducing the search space from 84 to 28 blocks—we believe this approach provides valuable insights that can inform the design of future diffusion transformer architectures.

7 Related Work

Training-aware Acceleration. Consistency Model [21, 38], introduces a consistency loss that significantly accelerates convergence, thereby reducing the number of timesteps required for stable performance. Additionally, ADD [33, 34], combines the strengths of Generative Adversarial Networks (GANs) and Diffusion models by employing an adversarial loss to effectively distill knowledge from a more complex, multi-timestep model into a more efficient, smaller timestep diffusion model. From a structural design perspective, the BK-SDM [14] MobileDiffusion [49] represents a series of Stable Diffusion models that enhance computational stability by strategically redistributing computational loads across different stages of the model. However, it is important to note that despite the potential for acceleration offered by these training-aware methods, most still necessitate substantial computational resources like thousands of GPU hours.

Post-training Acceleration. Post-training acceleration methods can be implemented without altering the original foundational models, while still making the denoising process more efficient. However, most of these methods involve some degree of loss. Notably, Flash Attention [4] is one of the few lossless acceleration methods seamlessly integrated into diffusion model computations. StreamDiffusion [15] proposes optimizing diffusion models at a pipeline level, incorporating techniques such as cache prompt embedding and utilizing hardware inference backends. DeepCache [22] and FasterDiffusion [16] advocate for cached output feature maps within UNet to bypass computation in certain stages. A more recent work [47] proposes to skip cross attention in the later fidelity-improving stage. However, most of these methods are not verified on fewer timestep diffusion models and also cannot be directly implemented in diffusion transformers.

7.1 Limitations

As we concentrate exclusively on training-free methodologies. Consequently, for diffusion transformers that operate with fewer timesteps, the reduction in computational ratio might not be as significant. Nonetheless, we are still able to achieve comparable generation outcomes, maintaining the majority of the structural integrity, albeit with some minor loss of detail clarity. As the number of timesteps increases, the search cost escalates; however, this results in a more pronounced reduction in computational requirements and achieves a better computation-quality trade-off.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claim in the abstract and method reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes] ,

   Justification: Please refer to Conclusions and Limitations section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: No theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: we have included the experimental details and the code will be released.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All of the models and data can be accessed through open GitHub repos or Huggingface models.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: we have included the experimental details and the code will be released.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Error bars are not reported.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have discussed computation resources in our search method.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research is conducted with NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Not applicable to societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original paper and dataset.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: This paper dose not introduce new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve study participants or any other mentioned above.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.