DESIGNING PARAMETER AND COMPUTE EFFICIENT DIFFUSION TRANSFORMERS USING DISTILLATION

Vignesh Sundaresha
University of Illinois Urbana Champaign
vs49@illinois.edu

ABSTRACT

Diffusion Transformers (DiTs) with billions of model parameters form the backbone of popular image and video generation models like DALL.E, Stable-Diffusion and SORA. Though these models are necessary in many low-latency applications like Augmented/Virtual Reality, they cannot be deployed on resource-constrained Edge devices (like Apple Vision Pro or Meta Ray-Ban glasses) due to their huge computational complexity. To overcome this, we turn to knowledge distillation and perform a thorough design-space exploration to achieve the best DiT for a given parameter size. In particular, we provide principles for how to choose design knobs such as depth, width, attention heads and distillation setup for a DiT. During the process, a three-way trade-off emerges between model performance, size and speed that is crucial for Edge implementation of diffusion. We also propose two distillation approaches - Teaching Assistant (TA) method and Multi-In-One (MI1) method - to perform feature distillation in the DiT context. Unlike existing solutions, we demonstrate and benchmark the efficacy of our approaches on practical Edge devices such as NVIDIA Jetson Orin Nano.

1 INTRODUCTION

Diffusion Transformers (DiTs) (Peebles & Xie, 2023) have become the de facto method (Dhariwal & Nichol, 2021) for generating images and videos due to their high fidelity (Ho et al., 2020), generalizability (Nichol & Dhariwal, 2021), ease of training (Ho et al., 2020) and scalability (Peebles & Xie, 2023). DiTs form the backbone of various practically deployed image and video generation models like DALL.E (Betker et al., 2023), StableDiffusion (Esser et al., 2024) and Sora (OpenAI, 2024). Due to the large parameter size and computational complexity of these models, one has to employ Cloud services to run them remotely. The significant latencies associated with such data transmissions from Cloud to the Edge cannot be afforded for high-frame-rate applications like Augmented/Virtual Reality (AR/VR) which need to be implemented on resource-constrained Edge devices (Apple Inc.; Meta Platforms, Inc.).

The main challenge in directly implementing neural network inferences on Edge devices comes from the limited memory and energy capacity of the Edge hardware. To address this, we need to *design parameter- and compute-efficient DiTs*. Edge devices which typically hold on-chip memories in the range of few megabytes, thus requiring model sizes to be in the order of a million parameters as compared to existing practical models (Peebles & Xie, 2023; Crowson et al., 2024) which have billions of parameters. Prior works (explained in detail in Appendix A) which focus on efficient DiTs optimize only specific layers (Pu et al., 2024) or look at only precision (Wu et al., 2024) or do not push the parameter-limit required to achieve the desired performance (Geng et al., 2024). The focus of our work is *not* about providing SOTA DiT models through novel algorithmic methods. Instead, *our goal is to provide the best DiT model - in terms of performance and speed - at a given parameter size using principled design choices.*

Contributions:

1. We provide principles for designing an efficient SOTA (at the given model size) DiT model (DiT-Nano) by employing distillation. Through the process, we highlight a key trade-off that emerges between model performance (FID), size (#parameters) and speed (latency). In particular, we show the practical impact of our designed models on real-life Edge devices such as NVIDIA Jetson Orin Nano. This is our key contribution.
2. We propose two algorithms - Teaching Assistant (TA) method and the Multi-In-One (MI1) method, and explore the efficacy of these methods.

2 DESIGNING EFFICIENT DITS

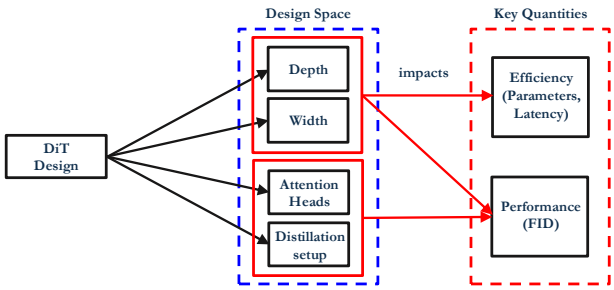


Figure 1: Design space exploration of diffusion distillation.

Design-Space Exploration: Among the several design knobs for distilling DiT models, we pick the following most relevant ones - depth, width, number of attention heads of the DiT model, and the setup (loss function and teacher models) for distillation. The former two knobs impact both the efficiency and performance, while the latter two only impact performance. We do not consider timesteps as a design knob despite its importance since extensive studies have been performed on that already (Peebles & Xie, 2023; Yin et al., 2024a) (we do 1-step diffusion only). Below we propose two methods which explore new Distillation Setups for DiT.

Teaching Assistant (TA) Method: This approach is inspired by the original TA paper (Mirzadeh et al., 2020) for distilling convolutional networks. We explore the possibility of combined feature distillation (see Fig. 2) using the teacher and TA with LPIPS loss (Zhang et al., 2018).

Multi-In-One (MI1) Method: This approach performs multiple diffusion timesteps in a single step by mapping the diffusion samples to specific layers of the student. The noise-image pair of the teacher model which does multi-step diffusion is used to calculate the intermediate noisy images by employing the forward diffusion probability flow ODE (shown in Appendix B.3).

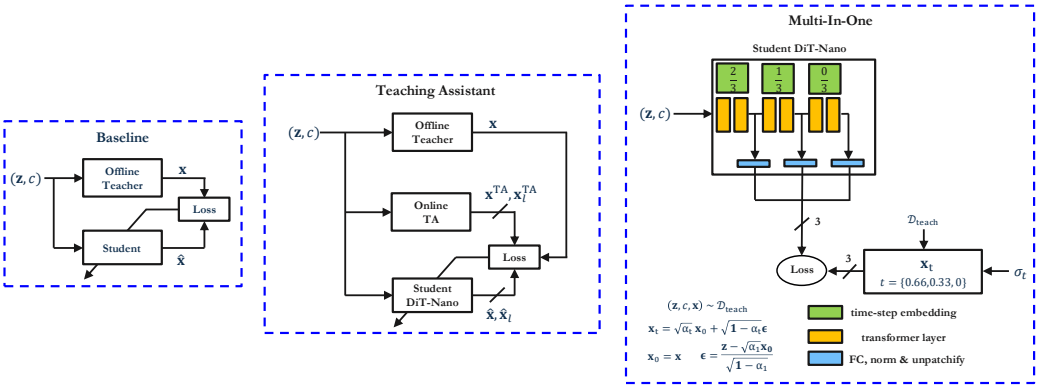


Figure 2: (Left) Baseline approach which performs regular knowledge distillation using offline teacher (Geng et al., 2024). (Center) Teaching Assistant (TA) which performs layer-wise distillation using an online teaching assistant and an offline teacher. (Right) Multi-In-One (MI1) approach which maps multiple diffusion steps into a single step. This is done by mapping different diffusion timesteps to different layers of a DiT. The training targets are obtained using forward diffusion of the probability flow ODE. See Appendix B for implementation details of TA and MI1.

3 EXPERIMENTS

In this section we mainly describe the principles for designing efficient DiTs and the experiments to corroborate them. The figure captions are provided with extensive detail and hence the text will only include points not mentioned in the captions.

3.1 SETUP

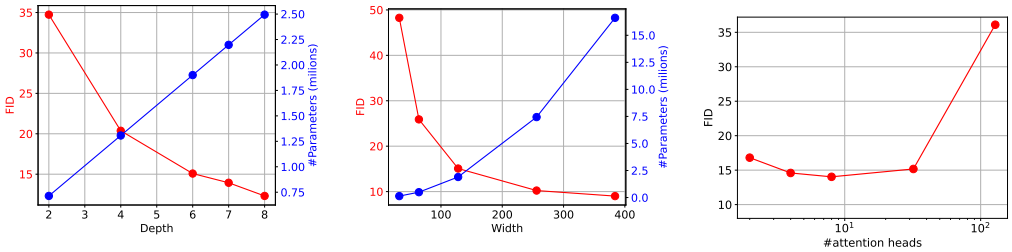
The comprehensive details of the training setup is provided in the Appendix C. The results below are shown for CIFAR-10 on DiTs distilled using EDM (Karras et al., 2022) as teacher. We do not show results for other larger models or datasets since such a design-space exploration is computationally infeasible. Thus we provide guidelines from our study for such bigger and more practical implementations. We use FID (Heusel et al., 2017) as the metric to evaluate the generated image performance

while using model size and latency (instead of FLOPs) as the metrics for efficiency. Even though our results can be extended to other scenarios, we consider here mainly the scenario of offline distillation (noise-image pairs of teacher are generated before training) due to compute resources.

3.2 Design Space Exploration and Principles

In this section, we look at the impact of various design knobs (mentioned in Section 2) on the performance and efficiency of the DiTs. Before sweeping the depth and width knobs, we first identify the best distillation setup based on existing methods and loss functions in Table 1. The huge impact of using LPIPS loss (Zhang et al., 2018) is obvious for both GET (Geng et al., 2024) and DMD (Yin et al., 2024b). The simpler training setup of GET provides better results compared to DMD.

Fig. 3a & 3b indicate how the depth d (no. of layers) and width w (embedding dimension) affect the FID and no. of parameters. The key takeaway is that increasing only one quantity without changing the other results in diminishing returns. When it comes to no. of attention heads h (which can only take factors of w as values), we find there is a sweet spot in the middle as shown in Fig. 3c.



(a) (b) (c)

Figure 3: (a) shows the impact of depth on FID and no. of parameters. It can be observed that the no. of parameters increases linearly with the increase in depth. (b) shows the impact of width on FID and no. of parameters. We see that the no. of parameters increases quadratically with increase in width. Both (a) & (b) show diminishing returns if increased independently and the performance (FID) does not scale in the same proportion as the increase in parameters, especially for width. (c) impact of no. of attention heads on FID. When the depth (d) or width (w) or attention heads (h) is not changing in any of these plots, it is assumed to be $d = 6$, $w = 128$ and $h = 4$ respectively.

Table 1: Compares FIDs for different distillation setups and loss functions for DiT. Effect of classifier-free guidance has been excluded for simplicity. The GET setup with LPIPS loss is the most effective.

Setup	Loss	FID
GET (Geng et al., 2024)	L1	45.60
GET (Geng et al., 2024)	L2	42.73
GET (Geng et al., 2024)	LPIPS	**18.30**
DMD (Yin et al., 2024b)	LPIPS	20.92

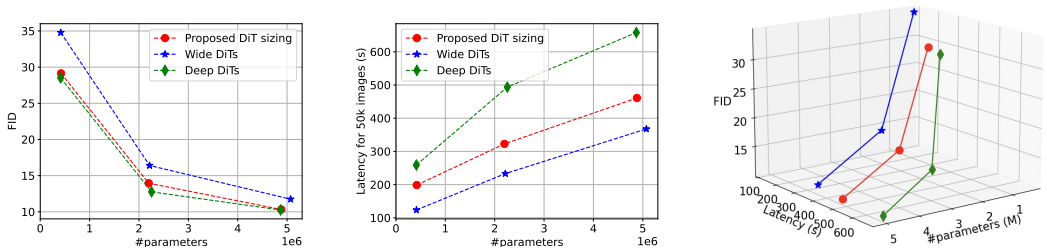
Table 2: Design-space exploration when the parameter size is 0.42M. The latency (L) for generating 50k images is measured on NVIDIA Jetson Nano.

Name	d	w	h	L (s)	FID
wider	1	128	8	97.5	57.07
wide	2	96	8	123.6	34.78
lower heads	5	64	4	162.6	29.66
proposed	**5**	**64**	**8**	198.7	29.14
deep	9	48	6	259.0	28.52
deeper	21	32	4	411.7	32.81

Design Principles: From the experiments in Fig. 3 and Table 1 we provide the below guidelines:

1. Use LPIPS loss when performing distillation for diffusion tasks.
2. Choose depth $d \approx \lfloor \log_2 w \rfloor$ with respect to the width w subject to satisfying the model parameter constraint.
3. Choose number of attention heads $h = \min \text{median}(\{\text{factors}(w)\})$.

The simulations that validate the design rules only explored widths which were of the form $\{2^n, 3.2^n\}$ since these utilize hardware most effectively. Above, the median for an even set of numbers is taken to be both the middle numbers. Fig. 4 and Table 2 show the benefits of our design principles (our proposed sizing is in bold) which achieve close to optimal FID for a much lower latency. One interesting aspect in Table 2 is that when we reduce the attention heads from our proposed suggestion



(a) (b) (c)

Figure 4: (a) Impact of number of parameters on FID when comparing deep DiTs, wide DiTs and the proposed sizing for DiTs. (b) Impact of parameters on the latency (on NVIDIA Jetson Nano). Wide DiTs have an observably worse performance while the deep DiTs have significantly higher latency due to serial processing. Table 2 shows the FID worsens going any deeper/wider. Our proposed sizing provides almost optimal FID at a much smaller latency. (c) The 3-dimensional trade-off highlighting the #params (memory) v/s FID (image quality) v/s latency (image frame rate) amidst optimal DiTs.

of 8 to 4, the latency reduces. Since this could related to the algorithm-microarchitecture mapping in NVIDIA Jetson Orin Nano, we *do not* include this factor in our design principles.

3.3 Impact of TA & MI1 methods and SOTA comparison

Table 3 suggests that feature distillation using the TA is not helpful. Only the distillation directly with the TA (architecture details in Appendix B.2) provides marginal benefits over the baseline obtained from our design principles. MI1 also performs worse compared to baseline as shown in Table 4. While these results are poor, we include them in support of the recent movement for publishing negative results (Blaas et al., 2025; Tafreshi et al., 2023; Guo et al., 2025). A crucial aspect to note is that the constraint on the intermediate layers is not the reason for worse performance since (2, 4, 6) performs better than (3, 6). Lastly, we compare our baseline approach with the only SOTA diffusion-based transformer model (Geng et al., 2024) that does model parameter distillation. We beat them on all metrics - model size, FID and latency (more on SOTA comparison in Appendix A.4).

Table 3: Impact of feature distillation using the TA method. The penultimate transformer layers of the TA and student are matched as indicated by (Muralidharan et al., 2024). Distilling with only TA (Mirzadeh et al., 2020) provides the best solution.

Teacher	TA	TA features	FID
✓			14.03
	✓		**13.99**
	✓	✓	14.36
✓	✓	✓	14.28

Table 4: Impact of using MI1.

Mapped Layers	timesteps	FID
Baseline	-	**14.03**
(3, 6)	(0.5, 0)	14.49
(2, 4, 6)	(0.66, 0.33, 0)	14.32

Table 5: SOTA comparison. Params (P) & Latency (L) for generating 50k images on A100 GPUs.

Method	P	FID	L (s)
GET(Geng et al., 2024)	8.6M	12.93	6.46
GET (Our principles)	5.3M	**10.21**	7.01
DiT-Nano	**5M**	10.32	**3.66**

4 Conclusion

We perform a thorough design space exploration of DiT distillation and provide design principles to obtain SOTA DiTs for a given model size. When these DiTs are implemented on NVIDIA Jetson Orin Nano, we identify a key trade-off between model performance-size-speed which can direct future researchers on practical areas to innovate. We hope the above guidelines serve as a reference when distilling DiTs for larger and more practically-relevant tasks. Through this paper, we also want to emphasize the practice of creating strong and obvious baselines (which was already SOTA in our case) before comparing the novel methods with prior works. Even though the TA method beats our baseline marginally, we conclude that the student model designed from our principles is a better option compared to the TA and MI1 methods due to cheaper training cost. Future directions can include justifying the above guidelines analytically, or expanding the design-space to knobs like MLP ratio and diffusion timesteps, or having custom attention heads for each layer (Michel et al., 2019), especially since there is an impact on latency with changing attention head size (see Table 2).

REFERENCES

Apple Inc. Apple vision pro. https://www.apple.com/apple-vision-pro/. Accessed: 2024-10-10.

David Berthelot, Arnaud Autef, Jierui Lin, Dian Ang Yap, Shuangfei Zhai, Siyuan Hu, Daniel Zheng, Walter Talbott, and Eric Gu. Tract: Denoising diffusion models with transitive closure time-distillation. *arXiv preprint arXiv:2303.04248*, 2023.

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

Arno Blaas, Priya Ronald DCosta, Fan Feng, Andreas Kriegler, Zhaoying Pan, Tobias Uelwer, Jennifer Williams, Yubin Xie, and Rui Yang. I can't believe it's not better: Challenges in applied deep learning. In *ICLR 2025 Workshop Proposals*, 2025.

Timothy J Boerner, Stephen Deems, Thomas R Furlani, Shelley L Knuth, and John Towns. Access: Advancing innovation: Nsf's advanced cyberinfrastructure coordination ecosystem: Services & support. In *Practice and Experience in Advanced Research Computing 2023: Computing for the Common Good*, pp. 173–176. 2023.

Lei Chen, Yuan Meng, Chen Tang, Xinzhu Ma, Jingyan Jiang, Xin Wang, Zhi Wang, and Wenwu Zhu. Q-dit: Accurate post-training quantization for diffusion transformers. *arXiv preprint arXiv:2406.17343*, 2024.

Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. In *Forty-first International Conference on Machine Learning*, 2024.

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

Tim Dockhorn, Robin Rombach, Andreas Blatmann, and Yaoliang Yu. Distilling the knowledge in diffusion models. In *CVPR Workshop Generative Modelsfor Computer Vision*, 2023.

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

Gongfan Fang, Kunjun Li, Xinyin Ma, and Xinchao Wang. Tinyfusion: Diffusion transformers learned shallow. *arXiv preprint arXiv:2412.01199*, 2024.

Zhengyang Geng, Ashwini Pokle, and J Zico Kolter. One-step diffusion distillation via deep equilibrium models. *Advances in Neural Information Processing Systems*, 36, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

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.

Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17535–17545, 2023.

Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. *arXiv preprint arXiv:2401.08740*, 2024.

Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14297–14306, 2023.

Meta Platforms, Inc. Orion - reality labs by meta. `https://about.meta.com/realitylabs/orion/`. Accessed: 2024-10-10.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 5191–5198, 2020.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Bhuminand Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PMLR, 2021.

OpenAI. Video generation models as world simulators. Technical report, 2024. URL `https://openai.com/index/video-generation-models-as-world-simulators`. Accessed: October 10, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Yifan Pu, Zhuofan Xia, Jiayi Guo, Dongchen Han, Qixiu Li, Duo Li, Yuhui Yuan, Ji Li, Yizeng Han, Shiji Song, et al. Efficient diffusion transformer with step-wise dynamic attention mediators. *arXiv preprint arXiv:2408.05710*, 2024.

Yifan Pu, Zhuofan Xia, Jiayi Guo, Dongchen Han, Qixiu Li, Duo Li, Yuhui Yuan, Ji Li, Yizeng Han, Shiji Song, et al. Efficient diffusion transformer with step-wise dynamic attention mediators. In *European Conference on Computer Vision*, pp. 424–441. Springer, 2025.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

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

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Shabnam Tafreshi, Arjun Akula, João Sedoc, Aleksandr Drozd, Anna Rogers, and Anna Rumshisky. Proceedings of the fourth workshop on insights from negative results in nlp. In *Proceedings of the Fourth Workshop on Insights from Negative Results in NLP*, 2023.

Joshua Tian Jin Tee, Kang Zhang, Hee Suk Yoon, Dhananjaya Nagaraja Gowda, Chanwoo Kim, and Chang D Yoo. Physics informed distillation for diffusion models. *Transactions on Machine Learning Research*.

Zhendong Wang, Yifan Jiang, Huangjie Zheng, Peihao Wang, Pengcheng He, Zhangyang Wang, Weizhu Chen, Mingyuan Zhou, et al. Patch diffusion: Faster and more data-efficient training of diffusion models. *Advances in neural information processing systems*, 36, 2024.

Noam Wies, Yoav Levine, Daniel Jannai, and Amnon Shashua. Which transformer architecture fits my data? a vocabulary bottleneck in self-attention. In *International Conference on Machine Learning*, pp. 11170–11181. PMLR, 2021.

Junyi Wu, Haoxuan Wang, Yuzhang Shang, Mubarak Shah, and Yan Yan. Ptq4dit: Post-training quantization for diffusion transformers. *arXiv preprint arXiv:2405.16005*, 2024.

Zhuoyi Yang, Heyang Jiang, Wenyi Hong, Jiayan Teng, Wendi Zheng, Yuxiao Dong, Ming Ding, and Jie Tang. Inf-dit: Upsampling any-resolution image with memory-efficient diffusion transformer. In *European Conference on Computer Vision*, pp. 141–156. Springer, 2025.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. *arXiv preprint arXiv:2405.14867*, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6613–6623, 2024b.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

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

Jianbin Zheng, Minghui Hu, Zhongyi Fan, Chaoyue Wang, Changxing Ding, Dacheng Tao, and Tat-Jen Cham. Trajectory consistency distillation. *arXiv preprint arXiv:2402.19159*, 2024.

Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *Forty-first International Conference on Machine Learning*, 2024.

Acknowledgements: The author would like to thank Prof. Naresh Shanbhag for his extensive support and discussions during this project. The author would also like to thank Kaining Zhou, Soonha Hwang and Nathan Chiang for the incredible collaboration out of which this work was a part of. This work was supported by the Center for the Co-Design of Cognitive Systems (CoCoSys) funded by the Semiconductor Research Corporation (SRC) and the Defense Advanced Research Projects Agency (DARPA). This work used Delta GPUs at NCSA through allocation CIS240722 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program Boerner et al. (2023), which is supported by U.S. National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296. Lastly, thanks to my advisor Prof. Lav Varshney for supporting me for this conference.

A Related Works

A.1 Diffusion Models and Architectures

Diffusion Models were originally introduced by (Sohl-Dickstein et al., 2015) and had a rapid rise later due to the works of (Song & Ermon, 2019; Ho et al., 2020; Song et al., 2020). (Karras et al., 2022) showed comprehensively how convolution-based U-Nets can be adapted to perform diffusion efficiently. More recently, with the advent of transformers, Diffusion Transformers (DiTs) (Peebles & Xie, 2023), Scalable Interpolant Transformers (SiTs) (Ma et al., 2024), Hourglass DiTs (HDiTs) (Crowson et al., 2024) and so on have been proposed to show the scalability of the transformer architecture for diffusion. Our work only looks at the distillation of DiTs (Peebles & Xie, 2023) even though the conclusions can be extended to the other transformer architectures mentioned above.

A.2 Efficient Diffusion Transformers

Even though many efficiency techniques (Wang et al., 2024; Li et al., 2023; Zhao et al., 2023) have been proposed for convolution-based diffusion models, we will focus here on the ones proposed for transformers. Quantization (Chen et al., 2024; Wu et al., 2024) and pruning (Fang et al., 2024) have been proposed for designing parameter-efficient DiTs. Efficient attention mechanisms (Pu et al., 2025; Yang et al., 2025) have also been proposed to perform efficient diffusion using transformers. However, all these methods are orthogonal to our approach of parameter-reduction using distillation. Our approach is the first to reduce both parameters and diffusion timesteps using distillation.

A.3 Timestep Distillation of Diffusion Models

To improve the slow generation process of diffusion models, extensive research has been done to reduce the timesteps through distillation (Luhman & Luhman, 2021; Salimans & Ho, 2022; Meng et al., 2023; Yin et al., 2024b;a; Zhou et al., 2024). Progressive distillation (Salimans & Ho, 2022) reduces the number of timesteps by two during each distillation stage, thus incurring a large training cost. (Meng et al., 2023) propose a classifier-free distillation method to generate images using 1-4 timesteps. (Yin et al., 2024b;a) use adversarial setups and distribution matching losses to perform one-step generation. Trajectory and consistency-based distillation (Berthelot et al., 2023; Song et al., 2023; Zheng et al., 2024) has also been considered to improve the speed of diffusion model generation. However, these methods are mainly proposed for U-Nets, do not consider parameter-reduction in models and hence typically initialize the student with the teacher at the beginning of training. Our problem looks at how to reduce both the parameters and timesteps for DiTs through distillation, which is considerably more challenging than just doing the latter. (Geng et al., 2024; Tee et al.) which do look at transformer distillation with a different student architecture mainly focus on the impact of equilibrium and physics-informed models, respectively, rather than pushing the parameters and compute down.

A.4 SOTA Comparison

The only fair comparison for our method is (Geng et al., 2024). Most works (Yin et al., 2024b;a; Berthelot et al., 2023) for diffusion distillation have focused solely on timestep distillation and the ones which do consider parameter distillation (Tee et al.; Dockhorn et al., 2023) are for U-Nets and do not focus on providing design principles like us. The methods which use U-Nets for timestep

distillation directly initialize their student model with the teacher model before starting training, a trick not available for methods like ours where there is architectural incompatibility in student-teacher networks. Hence we only compare with (Geng et al., 2024) which does parameter distillation for transformer-based diffusion. We also note that an expression similar to ours for depth was obtained in (Wies et al., 2021) independently and from a theoretical standpoint for different modality and non-distillation setup. This instills confidence in our design principles.

B Motivation and Implementation for Algorithms

B.1 GET setup

Motivation: A simple distillation setup that fulfills the essence of knowledge distillation for diffusion.

Implementation: Though this setup is same as (Geng et al., 2024), we explain it here and in Appendix C for completeness. The noise-image pairs are generated offline prior to training and employed as a dataset. This is necessary when training-compute budget is limited since it is more expensive to have an online teacher model generate samples on the fly. The teacher noise is fed as input to the student model and it is matched to the teacher image using a loss function.

$$\mathcal{L}_{\text{GET}} = \mathcal{L}_{\text{LPIPS}}(\mathbf{x}, \hat{\mathbf{x}}(\mathbf{z}, c)), \quad (\mathbf{z}, c, \mathbf{x}) \sim \mathcal{D}_{\text{teach}}$$

where $\mathbf{z}, c, \mathbf{x}$ represent the noise, class label and image drawn from the offline teacher dataset $\mathcal{D}_{\text{teach}}$, respectively. Here $\hat{\mathbf{x}}(\mathbf{z}, c)$ represents the output from the model which we will represent as $\hat{\mathbf{x}}$ moving forward.

B.2 TA method

Motivation: Since our approach looks at extremely tiny models (0.5M-5M model size) compared to the teacher model (EDM (Karras et al., 2022) with 62M parameters), the teaching assistant model helps bridge this disparity. Apart from that, to overcome the difficulty of feature distillation using a U-Net teacher, we use a DiT model for the TA to explore the benefits of feature distillation. We pick a reasonable choice for the TA architecture (DiT model with $(d, w, h) = (12, 384, 12)$) and leave the exploration of TA network architecture for future work.

Implementation: For the feature matching we take the transformer layer outputs from both TA and the student (expansion tensor is used for the student to match the TA width) and minimized using LPIPS loss. Based on (Muralidharan et al., 2024) we match the outputs from the penultimate transformer layer. The non-feature TA distillation approach is similar to the regular GET setup.

$$\mathcal{L}_{\text{GET}} = \lambda_0 \mathcal{L}_{\text{LPIPS}}(\mathbf{x}, \hat{\mathbf{x}}) + \lambda_1 \mathcal{L}_{\text{LPIPS}}(\mathbf{x}^{\text{TA}}, \hat{\mathbf{x}}) + \lambda_2 \sum_l \mathcal{L}_{\text{LPIPS}}(\mathbf{x}_l^{\text{TA}}, \hat{\mathbf{x}}_l), \quad (\mathbf{z}, c, \mathbf{x}) \sim \mathcal{D}_{\text{teach}}$$

Here the $\mathbf{x}^{\text{TA}}$, $\hat{\mathbf{x}}$, $\mathbf{x}_l^{\text{TA}}$ and $\hat{\mathbf{x}}_l$ represent the image output of the TA model, image output of the student model, intermediate layer l output of the TA model and intermediate layer l output of the student model, respectively. Depending on the configuration in Table 3, we make $\lambda_0 \neq 0$ (row 1, 4) or $\lambda_1 \neq 0$ (row 2, 3, 4) or $\lambda_2 \neq 0$ (row 3, 4) or all are non-zero (row 4).

B.3 MI1 method

Motivation: Even though many methods try to perform one-step diffusion distillation (Yin et al., 2024b;a), they do not explicitly map the intermediate outputs to each layer of the student. We do so since smaller models, unlike bigger models, have limited capacity to figure out how to effectively generate images. An explicit diffusion-based layer-wise guidance should ideally help teach them how to generate images using a "diffusion trajectory" instead of having to figure out how to do it.

Implementation: The goal of this method is to explicitly teach the small student model how to generate images by modeling the multi-step diffusion process in a single forward pass. We achieve this by mapping certain diffusion timestep samples to specific layers of the DiT. For example, if you consider a 1000 step diffusion process of the teacher(from a reverse diffusion perspective 0th step is

the noise sample and 1000th step is the image) to be performed in a single pass of a 8-layer student DiT, we map the 250th step to the 2nd layer, 500th to the 4th, 750th to the 6th and the 1000th to the last layer. For a practically feasible distillation, we look at the 18-step EDM teacher (Karras et al., 2022) to be mapped onto a single step of our DiT-Nano. We use the same noise-image pairs generated offline from the GET setup as our starting point. Instead of generating the intermediate timestep samples using the online EDM model, we use the forward probability flow ODE to generate them. We derive below the equations for forward diffusion of the probability flow ODE.

$$\mathbf{x}(t) = \sqrt{\alpha(t)}\,\mathbf{x}_0 \;+\; \sqrt{1-\alpha(t)}\,\epsilon, \quad \alpha(t) = \frac{1}{1+\sigma(t)^2}$$

$$\mathbf{x}(t) = \frac{1}{\sqrt{1+\sigma(t)^2}}\,\mathbf{x}_0 \;+\; \frac{\sigma(t)}{\sqrt{1+\sigma(t)^2}}\,\epsilon$$

$$\epsilon = \frac{\mathbf{z} \;-\; \sqrt{\alpha(1)}\,\mathbf{x}_0}{\sqrt{1-\alpha(1)}}$$

$$\sigma(t) = (\sigma_{\text{max}}{}^{\frac{1}{\rho}} + t(\sigma_{\text{min}}{}^{\frac{1}{\rho}} - \sigma_{\text{max}}{}^{\frac{1}{\rho}}))^{\rho}, \quad t = [0,1],\ \sigma_{\text{max}} = 80,\ \sigma_{\text{min}} = 0.02$$

Here the values of $\sigma(t)$ is taken directly from the implementation of EDM (Karras et al., 2022) while the other parts of the forward probability flow ODE has been derived. In the example shown in Fig. 2 and Table 4, we used $t = \{0.5, 0\}$ and $t = \{0.66, 0.33, 0\}$. In the former case, we map the timestep 0.5 to 3rd layer and 0 to last layer of a 6-layer DiT-Nano. In the latter case, we map 0.66 to 2nd layer, 0.33 to the 4th layer and 0 to the last layer of the same model. When we say mapping, we minimize the LPIPS loss between the two objects $(\mathbf{x}(t), \hat{\mathbf{x}}_l)$ being mapped.

$$\mathcal{L}_{\text{MI1}} = \sum_{l\in\Lambda, t\in\mathcal{T}} \mathcal{L}_{\text{LPIPS}}(\mathbf{x}(t), \hat{\mathbf{x}}_l)$$

where $\Lambda, \mathcal{T}$ are the sets of layers and timesteps that are being mapped, respectively. The preliminary results which we have reported in Table 4 have been poor and worse than the baseline generated from our guidelines. An easy argument or explanation for this could be that MI1 is forcing (constraining) the model to generate images through a diffusion trajectory, whereas it is possibly better for the model to figure the optimal way to generate the image from the input noise on its own. However, if one looks at $t = \{0.5, 0\}$ and $t = \{0.66, 0.33, 0\}$, we find the latter to have a lower FID even though it is more constraining, thus countering the argument provided above.

C SETUP

The training or distillation setup closely follows the GET setup (Geng et al., 2024) where as the model setup follows the DiT setup (Peebles & Xie, 2023).

Data: For training data we use the noise-image pairs generated from EDM (Karras et al., 2022) VP conditional model provided in the GitHub repository of (Geng et al., 2024). We only consider such an offline distillation method due to computational constraints of having an online teacher. All our results have been provided for only conditional generation similar to the DiT paper (Peebles & Xie, 2023). We acknowledge that our results and guidelines are on a small dataset and do not sweep extensively number of design points. However, we only provide our results on CIFAR-10 data due to computational constraints and hope our design principles will enable efficient design of DiTs for larger and more practical datasets and models.

Optimizer: We use AdamW optimizer with weight decay 0.01, a fixed learning rate of 0.0001, and a global batch size of 256. All experiments have been run for 100 epochs unless stated otherwise.

Models: Our backbone code for the model is adopted from the DiT paper (Peebles & Xie, 2023) with modifications to accommodate the TA and MI1 methods. We specifically look at DiTs since they are becoming the mainstream models for practical deployment over U-Nets. We also do not consider variants of DiTs (Ma et al., 2024; Crowson et al., 2024) since our results can be extended to those. Patch-size of 2 is used for all models.

Table 6: Design-space exploration for all parameter sizes. The latency for generating 50k images is measured on NVIDIA Jetson Orin.

#Params	D	W	H	Latency (s)	FID	IS
0.42M	1	128	8	97.5	57.07	6.38
	2	96	8	123.6	34.78	7.48
	5	64	4	162.6	29.66	7.76
	5	**64**	**8**	198.7	29.14	7.79
	9	48	6	259.0	28.52	7.78
	21	32	4	411.7	32.81	7.41
2.2M	3	192	12	233.2	16.40	8.78
	7	**128**	**8**	322.6	13.95	8.90
	13	96	8	493.7	12.76	8.92
5M	4	256	16	367.6	11.74	9.04
	16	128	8	657.9	10.25	9.15
	7	**192**	**12**	460.9	10.32	9.16
5M (200 epochs)	**7**	**192**	**12**	458.6	**8.90**	**9.31**

Hardware: Apart from Table 1 (which was implemented on NVIDIA Quadro RTX 6000), all the training experiments were run on $4\times$ A100 GPUs. While some of the inference has also been on $4\times$ A100 GPUs, the latency calculation for some models has been implemented on NVIDIA Jetson Orin Nano.

Inference: During inference we use an Exponential Moving Average (EMA) model trained with an EMA decay of 0.9999. We also employ classifier-free guidance during inference with a cfg-scale of 1.5. The latency was calculated for 50k images with a batch size 128. All our distillation methods have looked at only performing one-step diffusion with the student.

We will make the code public if accepted.

D EXTENSIVE RESULTS

Table 6 shows the comprehensive results from Fig. 4 in a tabular format. IS in Table 6 refers to Inception Score. All the results in the main paper and in Table 6 are run for 100 epochs, except the last row in Table 6 which is run for 200 epochs, and GET (Our principles) in Table 5 which was run for only 38 epochs! (due to compute constraints, GET training with LPIPS loss is very expensive). Increasing the epochs further can lead to further improvements in FID. For example, the 2.2M model with $D = 7, W = 128, H = 4$ achieved an FID of 10.91 due to training for 300 epochs.