Autoguided Online Data Curation for Diffusion Model Training

Valeria Pais
University of Glasgow
11 Chapel Ln, Glasgow G11 6EW, UK
v.pais-malacalza.1@research.gla.ac.uk

Luis Oala
Dotphoton
Nordstrasse 3, 6300 Zug, Switzerland
luis.oala@dotphoton.com

Daniele Faccio
University of Glasgow
11 Chapel Ln, Glasgow G11 6EW, UK
daniele.faccio@glasgow.ac.uk

Marco Aversa
Dotphoton
Nordstrasse 3, 6300 Zug, Switzerland
marco.aversa@outlook.com

Abstract

The costs of generative model compute rekindled promises and hopes for efficient data curation. In this work, we investigate whether recently developed autoguidance and online data selection methods can improve the time and sample efficiency of training generative diffusion models. We integrate joint example selection (JEST) and autoguidance into a unified code base for fast ablation and benchmarking. We evaluate combinations of data curation on a controlled 2-D synthetic data generation task as well as (3×64^2)-D image generation. Our comparisons are made at equal wall-clock time and equal number of samples, explicitly accounting for the overhead of selection. Across experiments, autoguidance consistently improves sample quality and diversity. Early AJEST —applying selection only at the beginning of training— can match or modestly exceed autoguidance alone in data efficiency on both tasks. However, its time overhead and added complexity make autoguidance or uniform random data selection preferable in most situations. These findings suggest that while targeted online selection can yield efficiency gains in early training, robust sample quality improvements are primarily driven by autoguidance. We discuss limitations and scope, and outline when data selection may be beneficial.

1. Introduction

Scaling trends have made modern generative modeling increasingly expensive, shifting attention from model- and hardware-centric efficiency towards data efficiency (e.g., [3, 5, 16, 24, 31]). Evidence suggests that targeted curation can improve performance by removing redundancy and noise [10]. Diffusion models, however, aim to approximate full data distributions, raising the question of whether data selection can help without harming diversity; recent work indicates pruning can help in some regimes [4].

We study online data selection for diffusion models through Autoguided JEST (AJEST), which integrates joint example selection (JEST) with autoguidance. We evaluate on a controlled 2-D task and Tiny ImageNet at $3 \times 64 \times 64$. Comparisons are made at equal wall-clock time and equal number of samples, explicitly accounting for selection overhead. Our focus is practicality: does targeted early selection provide efficiency gains, and how does it interact with autoguidance? **Our contributions and findings** are as follows:

1. A unified code base to run combinations of autoguidance and JEST for online data curation of diffusion models[1]
2. An evaluation harness for time and data efficiency of data curation during diffusion model training.
3. Early AJEST can match or modestly exceed autoguidance in data efficiency on both a 2-D and (3×64^2)-D; however, its time overhead and added complexity makes it less attractive for most applications.
4. A transparent discussion of limitations and where data curation hits limits.

Related work. Data selection and active learning span diversity-based approaches (e.g., k-center and submodular selections [1, 29]) and gradient/importance-based methods [2, 22, 28, 30, 37], as well as instance difficulty and loss-based strategies [6, 9, 11, 15, 27, 34, 35]. However, existing literature also suggests that random data selection can outperform these approaches in many settings [25].

JEST [8] formalizes online joint selection using a learner–reference pair and has been shown capable of replacing knowledge distillation in contrastive setups [36]. For diffusion models, classifier-free guidance [13] improves fidelity at the cost of diversity, whereas autoguidance [18]

[1] https://github.com/0xInfty/Autoguided

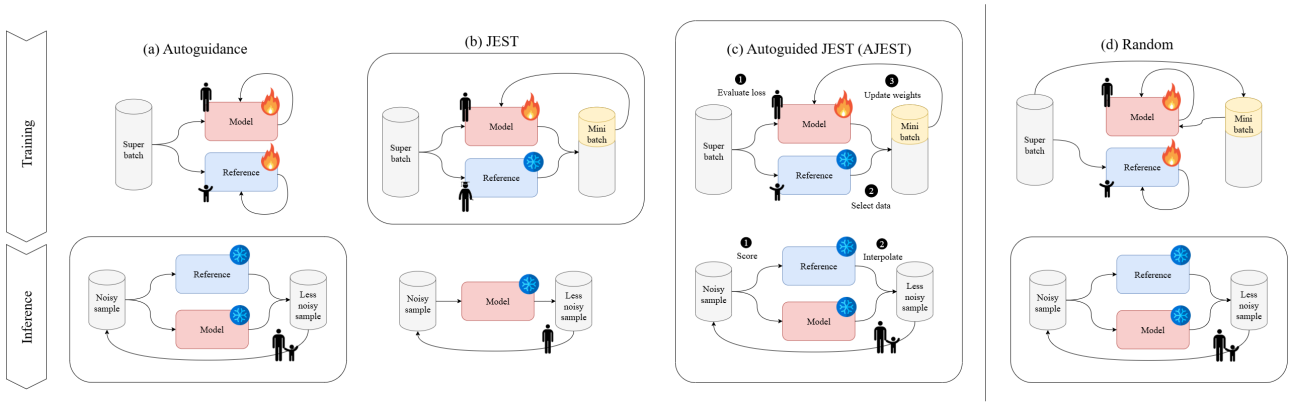

Figure 1. Visual summary. (a) JEST: a large pre-trained foundational reference model is used to select data to train the main model, which can be smaller in size; inference is done using only the main model. (b) Autoguidance: a smaller reference model is trained for less iterations; both reference and main model score samples during the denoising process. (c) Autoguided JEST (AJEST): combination of JEST's training strategy and autoguidance's small less-trained reference and collaborative denoising process. (d) Random: JEST is replaced with uniform random data selection to act as a benchmarking baseline with independent training for the main model.

leverages a weaker guide to boost quality while better preserving coverage. Our work examines how JEST-style online selection interacts with autoguidance for diffusion models under equal-time and equal-backprop comparisons.

2. Methods

We briefly review the core ingredients underlying the experiments, but we include details in Appendix A.

Diffusion models. Diffusion models learn a family of scores over noise levels and iteratively denoise samples from Gaussian noise towards the data distribution [17]. Sampling follows a probability-flow ODE discretized over a noise (time) schedule; the model provides score evaluations that guide each denoising step. We adopt EDM/EDM2 conventions for preconditioning and training [19].

Given the cost and scale of diffusion training, online data curation can appear appealing to reduce redundancy while preserving performance. Here we evaluate three regimes: an online learner–reference strategy (JEST) that adapts to training dynamics in combination with autoguidance (AJEST), a standalone autoguidance baseline, and a cheap random subset selection baseline (see Fig. 1).

JEST. Joint Example Selection (JEST) [8] pairs a learner with a reference model to score examples in a super-batch and draws a mini-batch via softmax sampling over learnability scores. With super-batch size B and filtering ratio f, the update mini-batch has size $b = (1-f)B$. The canonical learnability contrasts learner and reference losses and prioritizes examples that are easy to learn for the reference but not for the learner. JEST can act as implicit knowledge distillation in contrastive setups [36]. Exact scoring, chunked sampling logits, and our stability normalization are given in Appendix A.1.

Autoguidance. Classifier-free guidance (CFG) [13] improves fidelity at a diversity cost by mixing conditional/unconditional scores. Autoguidance [18] instead leverages a weaker guide model (smaller or less-trained) to produce a corrective signal that boosts quality while better preserving coverage; we follow their formulation and report both unguided and autoguided sampling. Exact guiding signal and collaborative sampling details are in Appendix A.1.

AJEST. We integrate JEST with autoguidance for diffusion models. A smaller guide (weaker version of the learner) serves as the JEST reference. Because the learner quickly surpasses the guide, we apply selection primarily at the beginning of training (Early AJEST) and then continue without selection; we also report a full-selection variant for completeness. At inference, we evaluate both unguided sampling and autoguided collaborative sampling.

Random data selection. As a cheap control, we uniformly sample a mini-batch of size b from each super-batch of size B, matching the filtering ratio and schedule used by selective methods. This removes learner–reference scoring entirely, yields negligible compute and memory overhead, and provides a strong time-efficiency baseline. The training loop and optimization settings are identical to those of AJEST and autoguidance; only the selection rule differs. In contrast with lowering the batch size, random data selection ensures some examples in the training dataset might never be seen by the learner model.

3. Experiments

We compare methods under two budgets: (i) equal time budget, explicitly including selection overhead; and (ii) equal data budget. This allows us to evaluate both time- and data-efficiency.

2-D Synthetic Data. We first tested our methods on Karras et al.'s 2D tree task [18]. The goal is to generate (x, y) points with most of its probability density lying within a tree-shaped manifold (Figure 2). This mimics low local di-

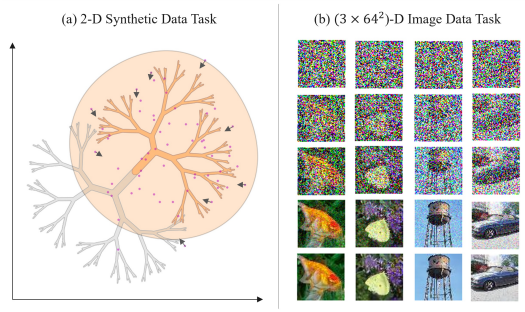

Figure 2. Illustration of the data generation tasks. (a) Normal 2D (x, y) samples (purple points) with noise level σ (orange circle) are pushed towards the ground truth tree to obtain the final samples (gray points). (b) High-dimensional Normal samples are denoised to obtain natural images matching Tiny ImageNet's distribution.

mensionality and hierarchical detail emergence in natural images [18]. We train simple diffusion models to sample from one of two classes (upper half of the tree).

We use Karras et al.'s toy model architecture [18]. We trained a small reference model for 512 iterations and a larger main learner model for 4096 iterations, optionally applying data selection with mini-batch size $b = 812$, super-batch size $B = 8192$ and filtering ratio $f = 0.8$. The loss function is evaluated on the whole batch of 8192 data points, but backpropagation is only executed for 20% of those points. Details on the model and distribution are provided in Appendix A.3. Evaluation uses loss- and coverage-based metrics (MSE, L2, mandala, and a simple classification accuracy); precise definitions are given in Appendix C.

(3×64^2)-D Image Data. We then scale up the dimensionality to Tiny ImageNet [20] using the public split (HuggingFace [38]). This dataset contains 64×64 RGB images and its training set has 200 classes following WordNet synsets as in ImageNet.

We trained an XS EDM2 model [19] as our main model for 22000 iterations. We defined a smaller XXS EDM2 model to use as the reference, and we trained it for only 5160 iterations. We used super-batch size $B = 2048$ and mini batch-size $b = 384$, effectively applying the same filtering ratio $f = 0.8$ as in the 2D toy task. Further details can be found in Appendix A.4. Evaluation uses FID [12], FD-DINOv2 [26, 32], and top-1/top-5 accuracy of samples against a pretrained Tiny ImageNet Swin-L classifier [14]; details are given in Appendix C.

4. Results

We report data-efficiency (metrics versus number of back-propagated examples) and time-efficiency (metrics versus wall-clock time, including selection overhead) as defined in Section 3. More results can be found in Appendix B.

2-D Synthetic Data. In the time-limited scenario, the 2D tree task results indicate that Early AJEST can be as time-efficient as random data selection and the baseline trained with no data selection. Despite this, Random may still be preferable due to its low algorithmic complexity and memory requirements. Results also show that Full AJEST is too time-consuming compared to other methods.

Table 1. Evaluation metrics with guidance on the 2D task. Results are averaged over 5 runs with different random seeds. Standard deviation is used to indicate uncertainty. Yellow-filled cells contain the best scores. Undistinguishable results are marked in bold.

Comparison	Method	Average L2 Distance		Mandala score	Classification score
		Full Tree	External Branches		
Same time budget	Baseline	0.246 ± 0.297	0.613 ± 0.210	**0.73 ± 0.11**	**0.94 ± 0.01**
	Early AJEST	0.110 ± 0.019	**0.516 ± 0.019**	**0.75 ± 0.08**	**0.93 ± 0.02**
	Full AJEST	0.118 ± 0.025	**0.514 ± 0.020**	0.53 ± 0.12	0.85 ± 0.04
	Random	**0.101 ± 0.006**	**0.508 ± 0.009**	**0.71 ± 0.08**	**0.93 ± 0.01**
Same data budget	Baseline	0.156 ± 0.053	0.543 ± 0.036	0.53 ± 0.15	0.83 ± 0.09
	Early AJEST	0.120 ± 0.029	**0.515 ± 0.030**	0.58 ± 0.14	0.86 ± 0.06
	Full AJEST	**0.102 ± 0.006**	**0.507 ± 0.009**	**0.69 ± 0.10**	**0.92 ± 0.02**
	Random	**0.101 ± 0.005**	**0.508 ± 0.005**	**0.72 ± 0.07**	**0.93 ± 0.01**

In the data-limited scenario, full AJEST performs on a similar level to random data selection, but its added complexity and time overhead makes it less preferable. Early AJEST sacrifices data-efficiency in favour of time-efficiency, but it still presents advantages over the pure autoguidance approach: it reaches slightly better results, while using less examples for backpropagation. This begets the question whether Early AJEST has the potential to unlock better performance on tasks with higher complexity and dimensionality, such as image generation.

Additional results and experimentation with other AJEST variants can be found in Appendix B.1.

(3×64^2)-D Image Data. Our Tiny ImageNet experiment shows further evidence that the time overhead of Early AJEST is negligible: it reaches results almost as good as autoguidance on this fix time budget. AJEST proves to be inefficient from a time-based perspective for images too.

Same as in the 2D task, Early AJEST achieved similar results to autoguidance on the fix data budget images scenario. It led to slightly better results on FID, FD-DINOv2 and Top-5 accuracy (Table 2), especially at very low data budgets (Figure 3). However, random data selection presented better perceptual metrics compared to both Early AJEST and Baseline. The computational overhead of Early AJEST makes random selection preferable.

Additional results including more generated images and other training and validation curves with all evaluated metrics can be found in Appendix B.2.

5. Discussion and limitations

Our study examines online data curation for diffusion models when combined with autoguidance. Two consistent observations emerge across tasks.

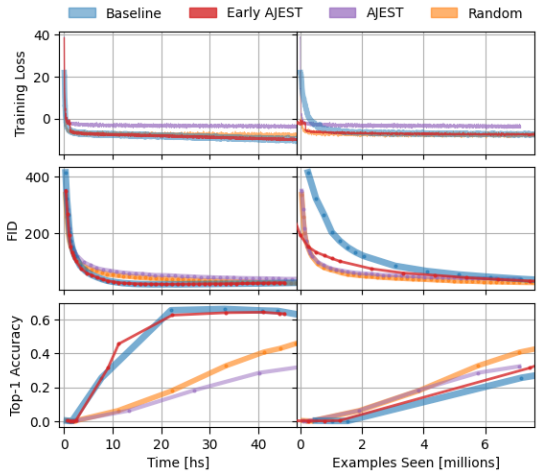

Figure 3. Training loss and validation metrics on Tiny ImageNet for EMA=0.10 and guidance weight α=2.2. Curves are shown versus number of backpropagated examples (data-efficiency) and wall-clock time (time-efficiency); error bars omitted for clarity.

Table 2. Evaluation metrics on Tiny ImageNet, averaging guided single-run results from EMA=0.05 and EMA=0.10 with α=1.7 and α=2.2 respectively. Yellow-filled cells contain the best scores. Close results are highlighted in bold.

Comparison	Method	Perceptual metrics		Classification-based metrics	
		FID	FD-DINOv2	Top-1	Top-5
Same time budget	Baseline	30.6	**624**	**59.8**	**80.6**
	Early AJEST	**31.0**	**628**	**58.8**	**79.8**
	AJEST	41.0	949	31.8	54.6
	Random	**27.5**	699	45.4	68.8
Same data budget	Baseline	40.4	934	**61.8**	**79.6**
	Early AJEST	**34.7**	849	**60.7**	**80.5**
	AJEST	41.0	949	31.8	54.6
	Random	**29.2**	**737**	40.9	65.0

First, autoguidance is a strong and reliable lever for improving quality while preserving diversity. It sets a robust baseline in our experiments that is difficult to surpass with data selection alone. Early AJEST—using the guide as the JEST reference only in the beginning of training—can yield modest data-efficiency gains and reach time-to-accuracy performance close to autoguidance on both the 2D and images task. However, the dominant improvements in coverage and fidelity are driven by autoguidance.

Second, accounting for overhead matters. Full AJEST introduces significant runtime overhead and is consistently unattractive in any time-limited scenario. Early AJEST keeps the overhead negligible in our setup and can therefore be competitive when time is the primary constraint, but its advantages shrink under a fixed-data budget where autoguidance already performs strongly. Random subsetting is frequently competitive because of its near-zero complexity and memory footprint.

The following limitations should be considered when

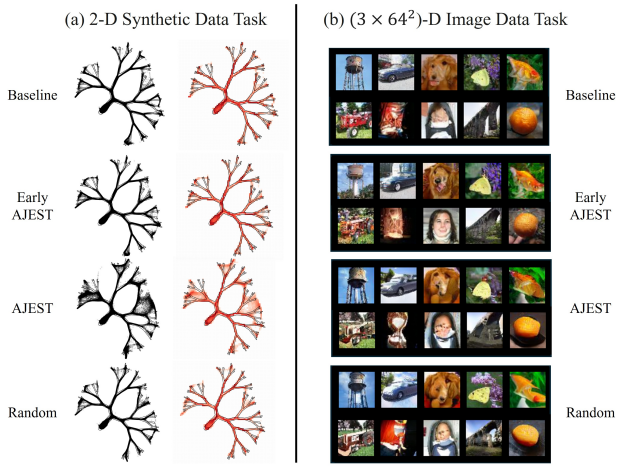

Figure 4. Qualitative results for fixed time budget on the (a) synthetic 2-D and (b) image generation tasks. For best viewing and fixed data budget results, please see full page figures in Appendix B.

intepreting the results. (i) Scope: we evaluate on a controlled 2-D task and Tiny ImageNet at 64×64; larger resolutions, datasets, and conditionings may alter the trade-offs. (ii) Metrics: while we report FD-DINOv2 and top-1/top-5 as more rigorous alternatives to FID, definitive perceptual assessment remains challenging. (iii) Design choices: we use only one reference model, one JEST parameterization and a simple early/always selection schedule; alternative parameters, chunking, and triggers could change outcomes. (iv) Data selection: we deploy AJEST, Early AJEST and a random uniform selection baseline; other methods, as those that use small proxy models, may offer improved synergy with autoguidance and diffusion models. (v) Hyperparameters: EMA and guidance weight were not exhaustively tuned; broader sweeps may shift the relative ranking at the margins. (vi) Engineering: results reflect a single hardware/software stack; absolute runtimes and overheads may vary.

Taken together, these results suggest a practical recipe: rely on autoguidance as the main driver of sample quality and diversity, and layer Early AJEST when training-time is scarce or when modest gains in data-efficiency are valuable relative to its small additional complexity. More aggressive selection schedules or heavier scoring generally do not justify their overhead in the setting explored here.

6. Conclusions

We evaluated Autoguided JEST (AJEST) for diffusion models on a 2-D toy task and Tiny ImageNet at 3×64^2 under equal time and data budgets. Autoguidance consistently improved quality and diversity and serves as a strong baseline. Early AJEST provided small but measurable data-efficiency gains with a minimal time overhead. In practice, we recommend autoguidance as

the default, with Early AJEST added when training-time is constrained or when minor data-efficiency gains justify the added machinery. Future work includes exploring richer selection scores and schedules, larger and higher-resolution datasets, and broader metric suites.

References

[1] Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. Contextual diversity for active learning. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI*, page 137–153, Berlin, Heidelberg, 2020. Springer-Verlag. 1

[2] Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations*, 2020. 1

[3] Bernd Bischl, Giuseppe Casalicchio, Taniya Das, Matthias Feurer, Sebastian Fischer, Pieter Gijsbers, Subhaditya Mukherjee, Andreas C. Müller, László Németh, Luis Oala, Lennart Purucker, Sahithya Ravi, Jan N. van Rijn, Prabhant Singh, Joaquin Vanschoren, Jos van der Velde, and Marcel Wever. Openml: Insights from 10 years and more than a thousand papers. *Patterns*, 6(7):101317, 2025. 1

[4] Rania Briq, Jiangtao Wang, and Stefan Kesselheim. Data pruning in generative diffusion models, 2025. 1

[5] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2829, 2023. 1

[6] Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2020. 1

[7] 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, 2009. 11

[8] Talfan Evans, Nikhil Parthasarathy, Hamza Merzic, and Olivier J Henaff. Data curation via joint example selection further accelerates multimodal learning. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. 1, 2, 5

[9] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Advances in Neural Information Processing Systems*, pages 2881–2891. Curran Associates, Inc., 2020. 1

[10] Sachin Goyal, Pratyush Maini, Zachary C. Lipton, Aditi Raghunathan, and J. Zico Kolter. Scaling laws for data filtering– data curation cannot be compute agnostic. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22702–22711, 2024. 1

[11] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. 1

[12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 3, 10

[13] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. 1, 2

[14] Ethan Huynh. Vision transformers in 2022: An update on tiny imagenet, 2022. 3, 10, 11

[15] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018. 1

[16] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. 1

[17] Tero Karras, Miika Aittala, Samuli Laine, and Timo Aila. Elucidating the design space of diffusion-based generative models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2022. Curran Associates Inc. 2, 1

[18] Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. In *Advances in Neural Information Processing Systems*, pages 52996–53021. Curran Associates, Inc., 2024. 1, 2, 3, 5, 10

[19] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24174–24184, 2024. 2, 3, 10

[20] Ya Le and Xuan S. Yang. Tiny imagenet classification with convolutional neural networks. Course Project Report Winter Quarter 2015, Stanford University, CS231n: Convolutional Neural Networks for Visual Recognition, 2015. Unpublished student project, available online. 3, 2, 11

[21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021. 10

[22] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6950–6960. PMLR, 2020. 1

[23] mnmoustafa and Mohammed Ali. Tiny imagenet (mnmoustafa version) on kaggle. https://kaggle.com/competitions/tiny-imagenet, 2017. Accessed: 2025-08-24. 11

[24] Luis Oala, Manil Maskey, Lilith Bat-Leah, Alicia Parrish, Nezihe Merve Gürel, Tzu-Sheng Kuo, Yang Liu,

Rotem Dror, Danilo Brajovic, Xiaozhe Yao, Max Bartolo, William A Gaviria Rojas, Ryan Hileman, Rainier Aliment, Michael W. Mahoney, Meg Risdal, Matthew Lease, Wojciech Samek, Debojyoti Dutta, Curtis G Northcutt, Cody Coleman, Braden Hancock, Bernard Koch, Girmaw Abebe Tadesse, Bojan Karlaš, Ahmed Alaa, Adji Bousso Dieng, Natasha Noy, Vijay Janapa Reddi, James Zou, Praveen Paritosh, Mihaela van der Schaar, Kurt Bollacker, Lora Aroyo, Ce Zhang, Joaquin Vanschoren, Isabelle Guyon, and Peter Mattson. DMLR: Data-centric machine learning research - past, present and future. *Journal of Data-centric Machine Learning Research*, 2024. 1

[25] Patrik Okanovic, Roger Waleffe, Vasilis Mageirakos, Konstantinos Nikolakakis, Amin Karbasi, Dionysios Kalogerias, Nezihe Merve Gürel, and Theodoros Rekatsinas. Repeated random sampling for minimizing the time-to-accuracy of learning. In *The Twelfth International Conference on Learning Representations*, 2024. 1, 3

[26] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. Featured Certification. 3, 10, 11

[27] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. In *Advances in Neural Information Processing Systems*, pages 20596–20607. Curran Associates, Inc., 2021. 1

[28] Omead Pooladzandi, David Davini, and Baharan Mirzasoleiman. Adaptive second order coresets for data-efficient machine learning. In *Proceedings of the 39th International Conference on Machine Learning*, pages 17848–17869. PMLR, 2022. 1

[29] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018. 1

[30] Seungjae Shin, Heesun Bae, Donghyeok Shin, Weonyoung Joo, and Il-Chul Moon. Loss-curvature matching for dataset selection and condensation. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pages 8606–8628. PMLR, 2023. 1

[31] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. In *Advances in Neural Information Processing Systems*, pages 19523–19536. Curran Associates, Inc., 2022. 1

[32] George Stein, Jesse Cresswell, Rasa Hosseinzadeh, Yi Sui, Brendan Ross, Valentin Villecroze, Zhaoyan Liu, Anthony L Caterini, Eric Taylor, and Gabriel Loaiza-Ganem. Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models. In *Advances in Neural Information Processing Systems*, pages 3732–3784. Curran Associates, Inc., 2023. 3, 10

[33] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016. 10

[34] Haoru Tan, Sitong Wu, Fei Du, Yukang Chen, Zhibin Wang, Fan Wang, and Xiaojuan Qi. Data pruning via moving-one-sample-out. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2023. Curran Associates Inc. 1

[35] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019. 1

[36] Vishaal Udandarao, Nikhil Parthasarathy, Muhammad Ferjad Naeem, Talfan Evans, Samuel Albanie, Federico Tombari, Yongqin Xian, Alessio Tonioni, and Olivier J. Hénaff. Active data curation effectively distills large-scale multimodal models, 2025. Accepted for the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2025 to be hosted in June 2025. 1, 2

[37] Xiaobo Xia, Jiale Liu, Jun Yu, Xu Shen, Bo Han, and Tongliang Liu. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *The Eleventh International Conference on Learning Representations*, 2023. 1

[38] zh plus. Tiny imagenet (zh-plus version) on hugging face. https://huggingface.co/datasets/zh-plus/tiny-imagenet, 2025. Accessed: 2025-08-19. 3, 2, 11

Autoguided Online Data Curation for Diffusion Model Training

Supplementary Material

A. Implementation details

A.1. JEST data selection

Joint example selection (JEST) samples training examples based on a learnability score [8]. Assuming that datapoints with indices $i \in 1, ..., B$ are fed into the learner model and samples with indices $j \in 1, ..., B$ are fed into the reference model, the learnability score s_{ij} is defined as...

$$s_{ij}^{learn} = L_{ij}^{L} - L_{ij}^{R} \tag{1}$$

Where L^L is the loss evaluated by the learner model and L^R is the loss evaluated by the reference model.

The joint batch selection is done in an iterative process with N steps by selecting n chunks of size $\frac{b}{N}$ (Figure 5). The first chunk C_0 is populated sampling from a uniform probability distribution over all B datapoints. For each successive chunk C_n, sampling is done on a conditional probability distribution over the B datapoints given all $n\frac{b}{N}$ previously selected datapoints that define a $\mathcal{K} = C_1 \cup ... \cup C_n$ set. This conditional probability distribution is modeled as a softmax distribution that takes certain logits as input. The logits $\mathbf{z} = (z_1, ..., z_B)$ are calculated adding four vector terms of length B.

1. The $(s_{11}, ..., s_{BB})$ diagonal scores that feed the same datapoint to both the learner and the reference models.
2. The sums of scores $(\sum_{k\in K} s_{k1}, ..., \sum_{k\in K} s_{kB})$ that result from only considering the selected datapoints fed into the learner model.
3. The sums of scores $(\sum_{k\in K} s_{1k}, ..., \sum_{k\in K} s_{Bk})$ that result from only considering the selected datapoints fed into the reference model.
4. A penalizing term whose elements are -10^8 for all selected datapoints in K and 0 for unselected datapoints.

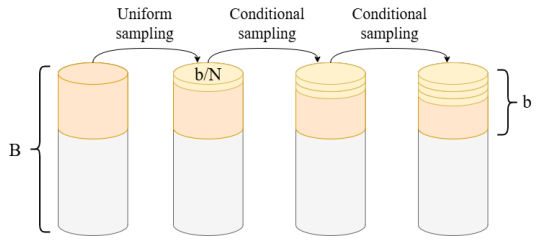

Figure 5. Iterative sampling process for JEST data selection.

Our code implementation follows quite closely the one published by Evans et al [8]. However, we observed that a softmax distribution applied directly to these $\mathbf{z}$ logits is highly unstable. For positive increasing scores, the exponential of the logits is unbounded and quickly causes a numerical overflow. We therefore applied a normalization factor to the logits before using them as input to the softmax function. We noticed that at any given JEST iteration with index n, the logits are calculated as the sum of $n.\frac{b}{N}$ terms. To bound the logits during the full iterative process, we simply divided them by that number, resulting in the following final expression:

$$z_l = \frac{N}{nb}\left(s_{ll} + \sum_{k\in\mathcal{K}} s_{kl} + \sum_{k\in\mathcal{K}} s_{lk} - \alpha_l\right) \tag{2}$$

A.2. Autoguidance and hyperparameters

By Karras et al.'s terminology [18], diffusion models are used to evolve any noisy sample $\mathbf{x}_{\text{initial}} \sim p(\mathbf{x}; \sigma_{\text{max}})$ from a Gaussian distribution $\mathcal{N}(\mathbf{x}; \sigma^2\mathbf{I})$ with maximum noise level σ_{max} to a sample matching the ground truth distribution $\mathbf{x}_{\text{final}} \sim p(\mathbf{x}; 0)$ with zero noise. The following probability flow ordinary differential equation needs to be solved:

$$d\mathbf{x}_\sigma = -\sigma\ S(\mathbf{x}_\sigma; \sigma)\ d\sigma \tag{3}$$

$$S(\mathbf{x}_\sigma; \sigma) = \nabla_{\mathbf{x}_\sigma} \log p(\mathbf{x}_\sigma; \sigma) \tag{4}$$

The diffusion model is trained to predict the score for every $\sigma \in [0, \sigma_{\text{max}}]$ and every example $\mathbf{x}_\sigma \sim p(\mathbf{x}_\sigma; \sigma)$. A $\sigma = t$ noise schedule set by a time step t can be used to obtain the final sample using an iterative approach [17]. Autoguidance improves the performance of diffusion models replacing the score from Eq. (3) with an interpolation of scores defined as in Eq. (5):

$$S = \alpha S^{\text{main}} + (1 - \alpha) S^{\text{ref}} \tag{5}$$

The S^{main} score is evaluated by the main model and the S^{ref} score is evaluated by an auxiliary reference model. The two models share the same architecture and conditioning, but the reference model is assumed to be a weaker version of the main model: a smaller model that is trained for less iterations. This applies a corrective force that pushes samples on each step of the denoising process towards regions on which the reference and the main model disagree on their predictions. This guidance is beneficial because it acts on the assumption that both the reference and the main model will make similar mistakes.

The guidance weight α is a key hyperparameter in our experiments. Autoguidance influences the results for any $\alpha > 1$. If set to $\alpha = 1$, the main scores are retrieved; if

set to $\alpha = 0$, the reference scores are retrieved instead. Another key hyperparameter is the EMA length: the duration over which an Exponential Moving Average (EMA) on the models weights is calculated during training.

Whenever available, we use α and EMA values directly extracted from Karras et al.'s publication [18]. For the 2D tree task, we use their default value $\alpha = 3$. On ImageNet-64 with EDM-XS, Karras et al report the best FID metric for $\alpha = 1.7$ and EMA=0.045 and the best FD-DINOv2 metric for $\alpha = 2.2$ and EMA=0.105. We train with default EMA=0.100 and EMA=0.050 values, so we adopt $\alpha = 2.2$ for the first one and $\alpha = 1.7$ for the second one.

A.3. 2D tree data generation task

The ground truth distribution for this task is modeled with a mixture of multi-variate 2D Gaussians (see Figure 6). Each brach is created out of 8 Gaussians uniformly-distributed along its central line segment. The exact same parameters from Karras et al [18] were used. The upper half of the tree is assigned to class A -the class we aim to sample from- and the lower half of the tree is assigned to class B. The tree is designed to have depth level 7, meaning a branch is split in two 7 times. To gain further resolution on our metrics and data analysis, we define as "external branches" all those branches with depth level 5 or more.

As described in [18], we built 2D diffusion models following a simple multi-layer perceptron architecture. The neural network's input consisted of a set of (x, y) coordinates and the σ noise level that correlates with the time step t of the denoising process. Four hidden layers with the same number of neurons followed the input layer. The scalar output was trained to represent the logarithm of the unnormalized probability density. Differentiating this value, the model could also return the score function. Training was done via exact score-matching. The loss function was defined as the mean squared error between the score and the ground truth score scaled by a σ^2 factor (see Equation (7) in C).

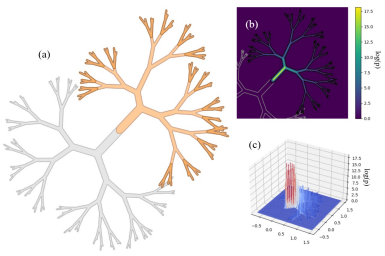

Figure 6. Visualization of the 2D tree ground-truth distribution. (a) A contour of the 2D tree distribution, showing which areas are considered class A and its external branches. (b) Color map of the logarithm of the ground-truth probability distribution function $\log p$. (c) A mixture of Gaussians is used to model the tree distribution and its peaks are visible in the 3D plot of $\log p$.

Following Karras et al. [18], we first trained a small reference model for only 512 iterations with hidden dimension 32. We then independently trained a larger main model with hidden dimension 64 for 4096 iterations. We trained on a single NVIDIA A5000 GPU. In this setup, a no-selection baseline could be trained for 4096 iterations in approximately 23 minutes; Full AJEST increased this to $\approx$ 36 minutes, while Early AJEST added only $\approx$ 38 seconds over baseline.

We store models every 128 training iterations and we evaluate data and time efficiency at the latest point possible during training. For the equal wall-time scenario, this happens at the last epoch of Baseline, the fastest method; the closest matching checkpoints are collected from all other methods, resulting on an average training time of (16.6 ± 0.5) min. For the scenario with equal number of backpropagated examples, this happens at the last epoch of AJEST and Random, the methods with the highest number of filtered examples; closest matching checkpoints result on an average of (3.3 ± 0.2) million examples.

A.4. EDM2 for Tiny ImageNet

As Karras et al. [18], we used the EDM2 diffusion model architecture [19] for image generation. EDM2 diffusion models are based on a UNet network with 4 blocks on its decoder and encoder. The XS EDM2 model has 128 channels on its first UNet block, whereas the XXS EDM2 model desgined by us has 64 channels instead. We trained these models to solve a natural images generation task on Tiny ImageNet [20] as found in HuggingFace [38]. The training dataset contains 500 64×64 RGB images for each of its 200 classes, adding up to a total of 1 million training examples. The categories correspond to synsets of the WordNet hierarchy, same as they do in the widely-used ImageNet dataset.

We know that Karras et al. [18] trained the XS model with batch size 2048 until it had seen 2147.5 Mimg from ImageNet, a dataset containing 1,281,167 training examples. We therefore estimated we would need to train the XS model for 167.6 Mimg in Tiny ImageNet to achieve convergence, assuming linear scaling on the training dataset size out of simplicity. Karras et al. [18] used a smaller model trained for 1/16th training iterations as their reference, so we decided we could use an XXS EDM2 model trained for 5160 iterations instead of the 81845 iterations required to go through 167.6 Mimg at batch size 2048.

We executed parallel-data distributed training on 2 NVIDIA A5000 GPUs. However, our dual GPU system could only achieve a speed of 8.2 sec/epoch or 1.1 hs/Mimg, forcing the convergence of the XS model to require 7.7 days. As we had seen that the best results in the 2D task could only be achieved by early data selection, we

decided to stop training the XS main model at 48 hs and 22000 iterations. However, we kept the reference model trained for 5160 iterations. In consequence, our reference model was trained for a 1/4th of the main model's total iterations, 4 times more what Karras et al do.

We converted the 8-bits Tiny ImageNet images into $[-1, 1]$ float images, and applied no other normalization besides the input preconditioning from the EDM2 model. We used Adam optimizer and we adopted Karras et al's EDM2 learning rate schedule [19]. However, to train with data selection we follow Okanovik et al's recommendation [25] on how to adapt the learning rate schedule to decay as fast as it would when training on the entire dataset (see Figure 1 from [19]).

We stored models every 500 training iterations. We followed the same methodology as in the 2D tree task to select a fix data and time budget for our analysis. For the equal wall-time scenario, this corresponded to (47.73 ± 0.07) hs; for the scenario with equal number of backpropagated examples, (7.2 ± 0.2) Mimg.

B. Additional results

B.1. 2D toy example with all metrics evaluated

Results for the complete set of metrics evaluated both with guidance and no guidance at equal wall time and equal number of backpropagated examples are presented on Table 3. We also include visual representations of the distribution of generated points in Figures 7 and 8 for the fixed time and fixed data budget scenarios, both with guidance and without it.

Unguided models reach better performance on the average L2 metric in external branches, but this is in clear conflict with the perceptual quality assessment that can be done on the generated points distributions. As Karras et al. state, guided models generate points closer to the high-density regions of the distribution, avoiding regions between branches where the ground-truth probability density is low. This is what originally led us to develop the classification and mandala scores (see Appendix C), a set of metrics that better reflects autoguidance advantages and rewards the desired behaviour of diffusion models in the 2D toy task.

B.2. EDM2 for Tiny ImageNet

We present images generated on the fix-budget and fix-time scenarios in Figure 9. In the same-time comparison, it is remarkable for Early AJEST to lead to the first human face and the first goldfish with eyes. The autoguidance baseline, however, leads to the first tractor and the closest image to a golden retriever's face.

We also present images generated using different EMA and guidance weight in Figure 10. These hyperparameters are shown to have a large influence on the image quality, as first reported by Karras et al. [18]. Guidance improves sharpness and contents considerably for EMA=0.10, but not as much for EMA=0.05. This might be due to 2.2 being closer to the ideal EMA for 0.10 than 1.7 is for 0.05.

Tables with separate results for EMA=0.05 and EMA=0.10 -both with guidance and without it- can be seen in Tables 5 and 5. Results averaged over those two EMA, but with both guided and unguided metrics, are presented in Table 6.

Our Tiny ImageNet results show new evidence that autoguidance is especially beneficial to improve coverage and precision over the ground-truth distribution: guidance improves all Swin-L top-1 and top-5 results, for as much as 10% in the best-case scenario.

In the time-limited scenario, there is a marked difference between FID and all other metrics. Random reaches the best FID score, and Early AJEST outperforms autoguidance. However, all other metrics show Baseline as the best, followed by Early AJEST, Random and finally AJEST. This may be added to the list of alarming evidence on how FID might not be an adequate metric to measure generated image quality [32]. We strongly believe that generative vision AI would benefit from better metric definitions.

We include training and validation curves for different EMA and guidance combinations in Figures 11 to 14. The trigger for Early AJEST can be deduced from the training curves, as its training loss falls from AJEST's to Baseline's. Data selection methods lead to consistently better results at low data budgets, especially Full AJEST and random data selection. Early AJEST only shows limited-data advantages over Baseline on FID and FD-DINOv2, but it follows quite closely its Top-1 and Top-5 performance under limited-time settings.

Evidence suggests that data selection might help mitigate overfitting on FID and FD-DINOv2 for unguided models: this is especially noticeable on the FID plot for EMA=0.05 and $\alpha = 1.7$ (Figure 14). However, this effect might also be explained by this being a very early stage in training for AJEST and Random; the same overfitting phenomenum might be present at later training times.

B.3. 2D toy example and other AJEST strategies

We explored the possibility of using both JEST's learnability score and an inverted learnability score:

$$s_{ij}^{inv} = L_{ij}^{R} - L_{ij}^{L} = -s_{ij}^{learn} \tag{6}$$

Table 3. All evaluation metrics applied to the 2D tree task. Results are averaged over 5 runs with different random seeds. Standard deviation is used to indicate uncertainty. Yellow-filled cells indicate the best scores for each metric: the lowest for the loss and L2 distance, and the highest for classification-based scores. Undistinguishable results according to the uncertainty are marked in bold.

Comparison	Method	Average Loss		Average L2 Distance				Mandala score		Classification score	
		Full Tree	External Branches	Full Tree		External Branches		Unguided	Guided	Unguided	Guided
				Unguided	Guided	Unguided	Guided				
Same time budget	Baseline	**0.011 ± 0.003**	**0.227 ± 0.007**	**0.019 ± 0.005**	0.246 ± 0.297	**0.482 ± 0.003**	0.613 ± 0.210	**0.51 ± 0.07**	**0.73 ± 0.11**	**0.88 ± 0.02**	**0.94 ± 0.01**
	Early AJEST	**0.011 ± 0.003**	**0.227 ± 0.008**	**0.018 ± 0.004**	0.110 ± 0.019	**0.483 ± 0.003**	**0.516 ± 0.019**	**0.50 ± 0.08**	**0.75 ± 0.08**	**0.88 ± 0.02**	**0.93 ± 0.02**
	Full AJEST	0.024 ± 0.006	0.253 ± 0.009	0.028 ± 0.008	0.118 ± 0.025	**0.482 ± 0.004**	**0.514 ± 0.020**	0.40 ± 0.07	0.53 ± 0.12	0.80 ± 0.03	0.85 ± 0.04
	Random	**0.013 ± 0.003**	**0.232 ± 0.005**	**0.019 ± 0.003**	**0.101 ± 0.006**	**0.482 ± 0.002**	**0.508 ± 0.009**	**0.47 ± 0.06**	**0.71 ± 0.08**	**0.87 ± 0.02**	**0.93 ± 0.01**
Same data budget	Baseline	0.026 ± 0.008	0.255 ± 0.011	0.034 ± 0.010	0.156 ± 0.053	0.487 ± 0.003	0.543 ± 0.036	0.40 ± 0.07	0.53 ± 0.15	0.80 ± 0.05	0.83 ± 0.09
	Early AJEST	0.022 ± 0.007	0.249 ± 0.012	0.028 ± 0.006	0.120 ± 0.029	**0.481 ± 0.005**	**0.515 ± 0.030**	**0.42 ± 0.07**	0.58 ± 0.14	0.81 ± 0.05	0.86 ± 0.06
	Full AJEST	**0.013 ± 0.003**	**0.234 ± 0.008**	**0.018 ± 0.004**	**0.102 ± 0.006**	**0.481 ± 0.002**	**0.507 ± 0.009**	**0.46 ± 0.06**	**0.69 ± 0.10**	**0.86 ± 0.03**	**0.92 ± 0.02**
	Random	**0.012 ± 0.002**	**0.231 ± 0.005**	**0.019 ± 0.003**	**0.101 ± 0.005**	**0.482 ± 0.001**	**0.508 ± 0.005**	**0.47 ± 0.06**	**0.72 ± 0.07**	**0.87 ± 0.02**	**0.93 ± 0.01**

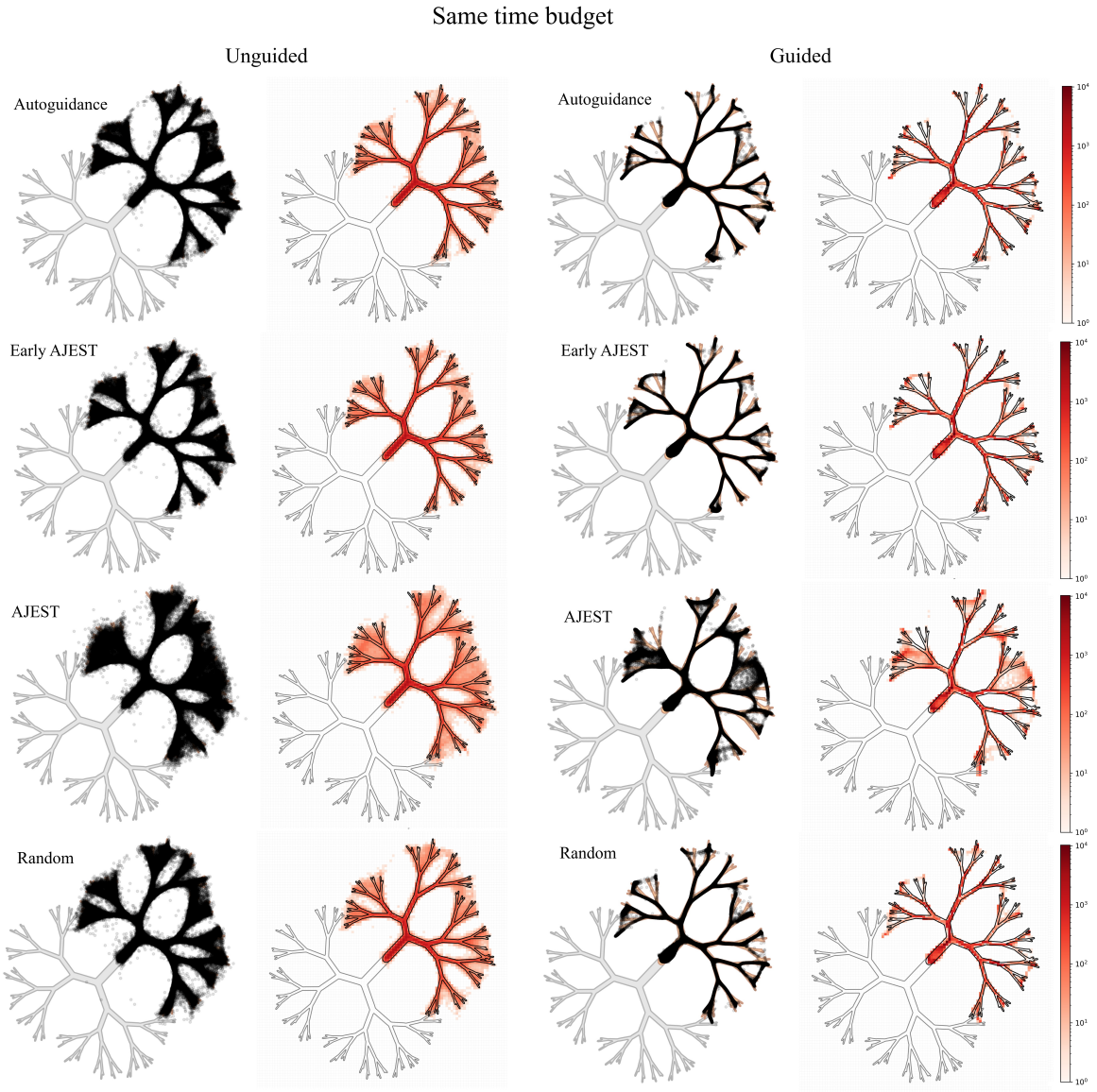

Figure 7. Distribution of generated points on the 2D tree task for the fix time budget of (16.6 ± 0.5) min.

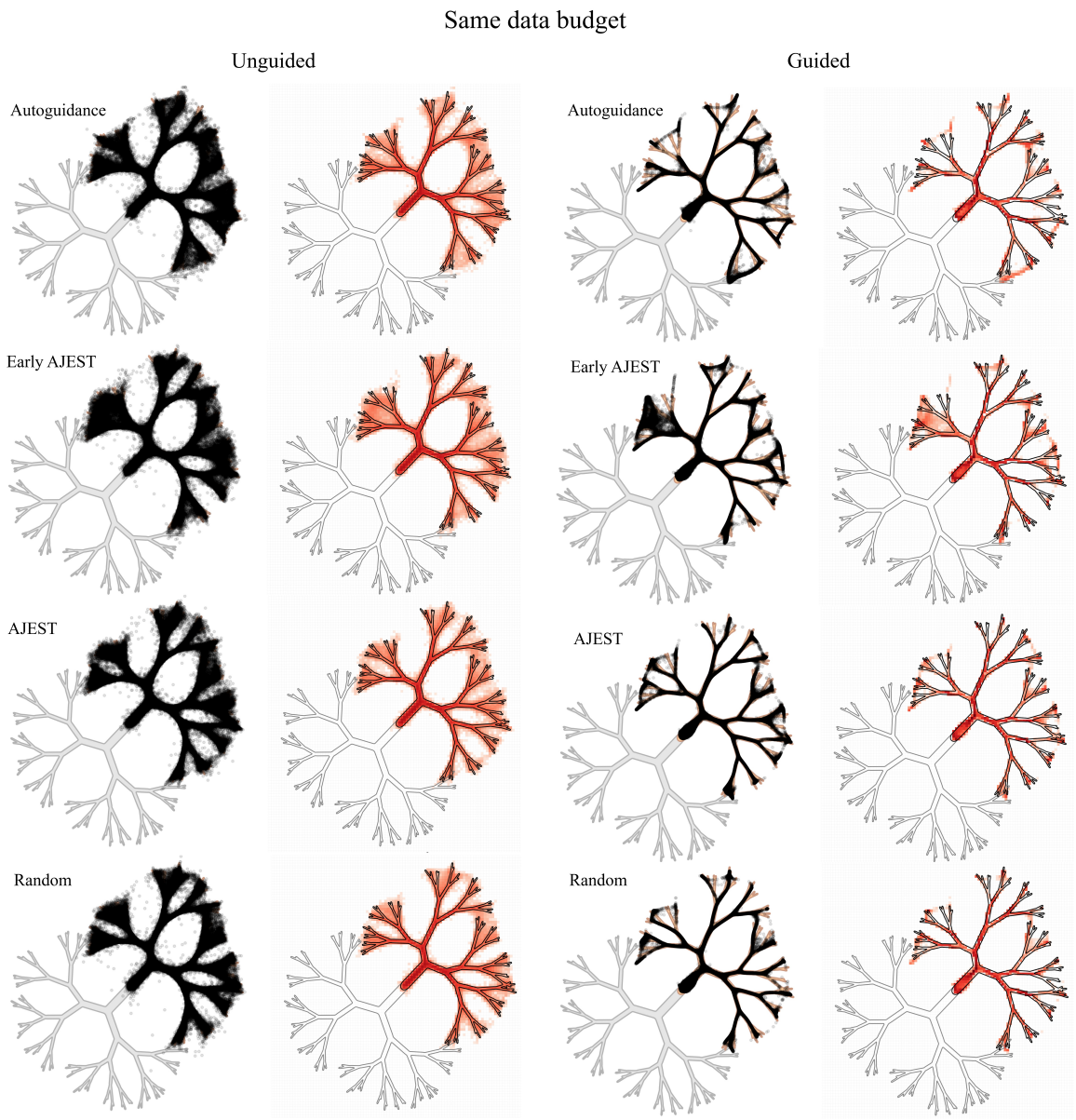

Figure 8. Distribution of generated points on the 2D tree task for the fix data budget of (3.3 ± 0.2) million training examples.

In JEST they use Eq. (1) to prioritize extremely difficult samples that are only easy to learn for a high-capacity reference [8]. This learnability score is therefore designed for situations in which a larger pre-trained model with good performance is available to use it as the smaller learner model's teacher. Karras et al's research was framed on a different situation, one in which we do not count with a different, better pre-trained model, but with a "bad version" of the learner model instead: a copy of the learner model with restricted capability [18]. We therefore hypothesized that it may be necessary to invert the sign of the learnability score, using Eq. (6) to compensate for the guide not being more capable than the learner. We refer to this approach as inverted JEST or iJEST.

In addition to Early AJEST, we also explored a Late AJEST strategy. As indicated in Figure 15, data selection would only be run late in training under the Late AJEST strategy. We assumed that Early AJEST and Late iAJEST may reach complementary results, especially when combined with the inverted and non-inverted learnability scores. If data selection with learnability scores required a large pre-trained model to work correctly, then we would expect to see it work better early in training (Early AJEST). In contrast, in case the inverted learnability scores would help alle-

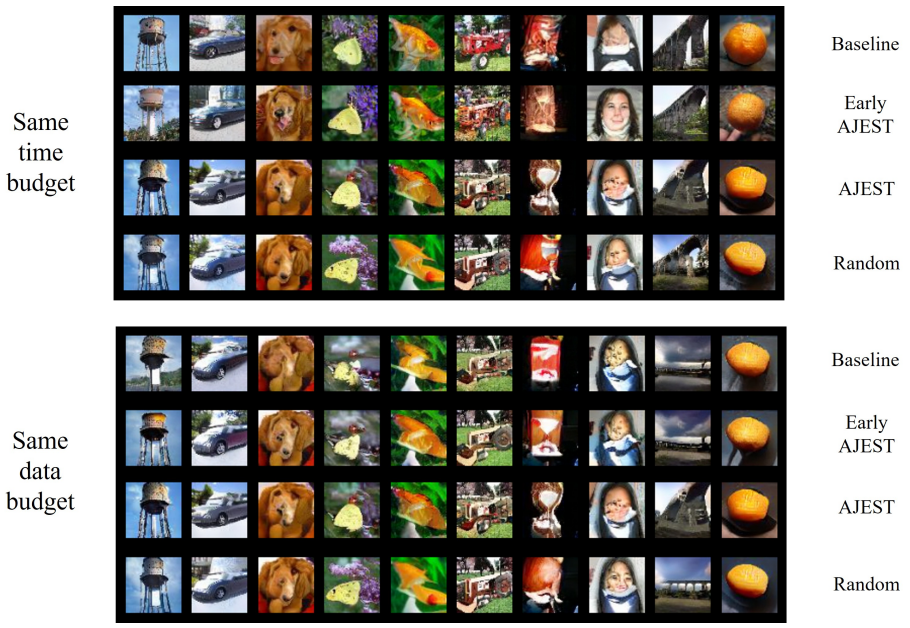

Figure 9. Generated images for 10 classes on the 48 hs time-limited and 7.1 Mimg data-limited scenarios: (1) water tower, (2) convertible, (3) golden retriever, (4) sulphur butterfly, (5) goldfish, (6) tractor, (7) hourglass, (8) neck brace, (9) viaduct, (10) orange.

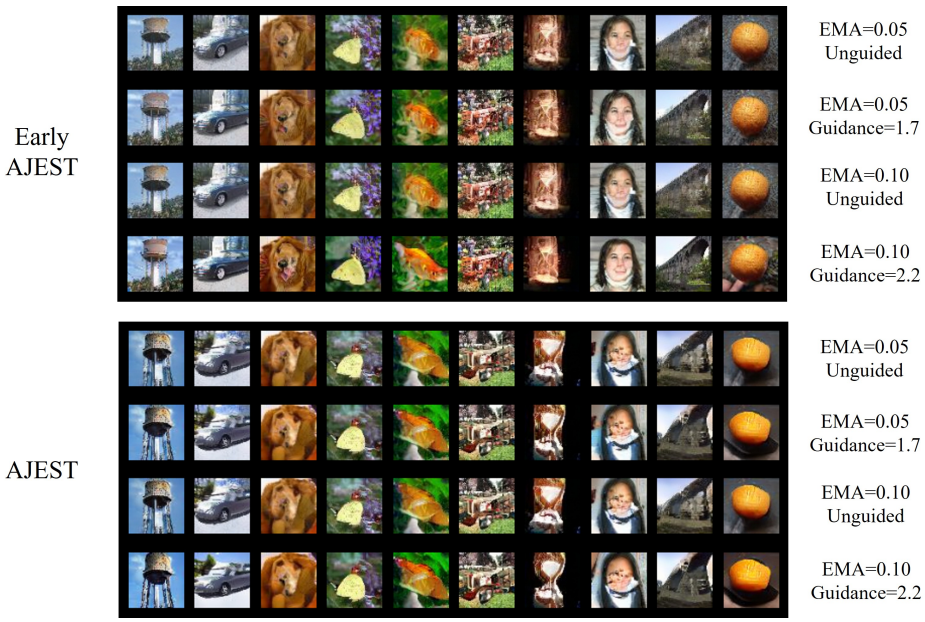

Figure 10. Generated images for 10 classes for AJEST under two different EMA values, both with guidance and without it: (1) water tower, (2) convertible, (3) golden retriever, (4) sulphur butterfly, (5) goldfish, (6) tractor, (7) hourglass, (8) neck brace, (9) viaduct, (10) orange.

viate this JEST requirement, then we would expect iAJEST to work better late in training (Late iAJEST).

Inverted JEST learnability scores applied late in training can lead to improvements superior to 5% both for the average L2 distance and the mandala score. In general, iJEST seems to improve the coverage over the generated samples distribution, meaning it can lead to generative models with higher diversity. Despite this, applying iJEST late in training for an extended period of time seems to over-estimate the autoguidance corrective force, resulting in lower average loss and L2 distance evaluated in the full tree. This seems to indicate that iJEST scoring enforces a trade-off between diversity and fidelity or precision. The same trade-

Table 4. Evaluation metrics on Tiny ImageNet with EMA=0.05. Yellow-filled cells indicate the best scores for each block of the table, considering both unguided results and guided results using guidance weight 1.7. Bold values indicate the best results for guided and unguided models separately.

Comparison	Method	Perceptual metrics				Classification-based metrics			
		FID		FD-DINOv2		Top-1		Top-5	
		Unguided	Guided	Unguided	Guided	Unguided	Guided	Unguided	Guided
Same time budget	Baseline	44.6	34.3	822	**677**	**47.9**	**56.8**	**69.8**	78.5
	Early AJEST	45.5	34.7	**813**	**677**	46.3	54.2	68.9	**76.8**
	AJEST	42.7	41.8	959	959	29.3	31.2	54.2	53.7
	Random	**33.8**	**28.6**	799	717	39.2	44.9	62.9	67.6
Same data budget	Baseline	41.5	40.8	927	927	**47.7**	**58.4**	**70.3**	77.0
	Early AJEST	38.8	36.1	878	857	45.3	57.1	69.6	**78.3**
	AJEST	42.7	41.8	959	959	29.3	31.2	54.2	53.7
	Random	**34.9**	**30.4**	**823**	**753**	35.9	41.0	60.5	64.6

Table 5. Evaluation metrics on Tiny ImageNet with EMA=0.10. Yellow-filled cells indicate the best scores for each block of the table, considering both unguided results and guided results using guidance weight 2.2. Bold values indicate the best results for guided and unguided models separately.

Comparison	Method	Perceptual metrics				Classification-based metrics			
		FID		FD-DINOv2		Top-1		Top-5	
		Unguided	Guided	Unguided	Guided	Unguided	Guided	Unguided	Guided
Same time budget	Baseline	40.4	26.9	774	**571**	**51.9**	62.8	**72.5**	82.7
	Early AJEST	40.9	27.2	**769**	578	48.2	**63.4**	72.2	**82.9**
	AJEST	41.5	40.3	950	940	29.4	32.5	55.9	55.5
	Random	**32.8**	**26.4**	790	681	40.7	45.9	63.3	69.9
Same data budget	Baseline	41.1	40.1	930	940	**50.4**	**65.3**	**72.7**	82.2
	Early AJEST	37.7	33.4	871	842	47.7	64.3	71.9	**82.7**
	AJEST	41.5	40.3	950	940	29.4	32.5	55.9	55.5
	Random	**34.0**	**28.1**	**815**	**720**	36.0	40.8	60.7	65.4

Table 6. Evaluation metrics on Tiny ImageNet, averaging results from EMA=0.05 and EMA=0.10. Bold yellow values indicate the best scores for each block of the table, considering both unguided results and guided results using guidance weight 1.7 for EMA=0.05 and 2.2 for EMA=0.10. Bold values indicate the best results for guided and unguided models separately.

Comparison	Method	Perceptual metrics				Classification-based metrics			
		FID		FD-DINOv2		Top-1		Top-5	
		Unguided	Guided	Unguided	Guided	Unguided	Guided	Unguided	Guided
Same time budget	Baseline	42.5	30.6	798	**624**	**49.9**	**59.8**	**71.1**	**80.6**
	Early AJEST	43.2	**31.0**	**791**	**628**	47.3	**58.8**	70.6	79.8
	AJEST	42.1	41.0	955	949	29.3	31.8	55.0	54.6
	Random	**33.3**	**27.5**	795	699	39.9	45.4	63.1	68.8
Same data budget	Baseline	41.3	40.4	929	934	**49.0**	**61.8**	**71.5**	**79.6**
	Early AJEST	38.2	**34.7**	875	849	46.5	**60.7**	70.8	**80.5**
	AJEST	42.1	41.0	955	949	29.3	31.8	55.0	54.6
	Random	**34.4**	**29.2**	**819**	**737**	36.0	40.9	60.6	65.0

off is apparently not as strongly enforced by non-inverted JEST scores, allowing Early JEST to improve autoguidance metrics without any marked disadvantages.

The best results were obtained applying Early AJEST. The cost of applying JEST in this way was an increase of just 2.7% in the training time, because the data selection

strategy was only applied to 190 of the 4096 training iterations. Early JEST improvements were less marked whenever guidance interpolation was applied in the inference phase, indicating that JEST may be able to use the small less-capable reference model only during training, discarding it at the inference phase.

Despite the high filtering ratio of 80%, our Early JEST method does not greatly reduce the volume of data used for training. Applied for 194 iterations on a super-batch size B of 4096, Early JEST implies only 1% of the total number of data points are discarded. In consequence, it may seem that Early JEST is not particularly data-efficient. Its success does not contradict the notion that diffusion models cannot learn a distribution accurately enough if a significant number of examples are discarded. However, the success of Early JEST in the 2D tree task does seem to imply that there are better ways to use data for diffusion models to learn. A slightly more data-efficient algorithm may still be more data-efficient that one that samples data uniformly from the available distribution.

C. Metrics definition

To quantitatively evaluate the performance of the diffusion models and the impact of data selection strategies, we employ several complementary metrics to capture both fidelity to the ground truth and diversity of the generated samples.

C.1. Metrics for the 2D tree toy example

We used 163,840 generated points with the same random seeds to evaluate 2D tree metrics.

Average Loss. The primary training and evaluation loss is the mean squared error (MSE) between the model's predicted score function and the ground truth score, scaled by the noise level σ^2:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} \sigma_i^2 \left\| S(\mathbf{x}_i, \sigma_i) - s^{gt}(\mathbf{x}_i, \sigma_i) \right\|^2 \quad (7)$$

where N is the batch size, $\mathbf{x}_i$ is a data sample, σ_i is the noise level, S is the predicted score with or without guidance, and s^{gt} is the ground truth score.

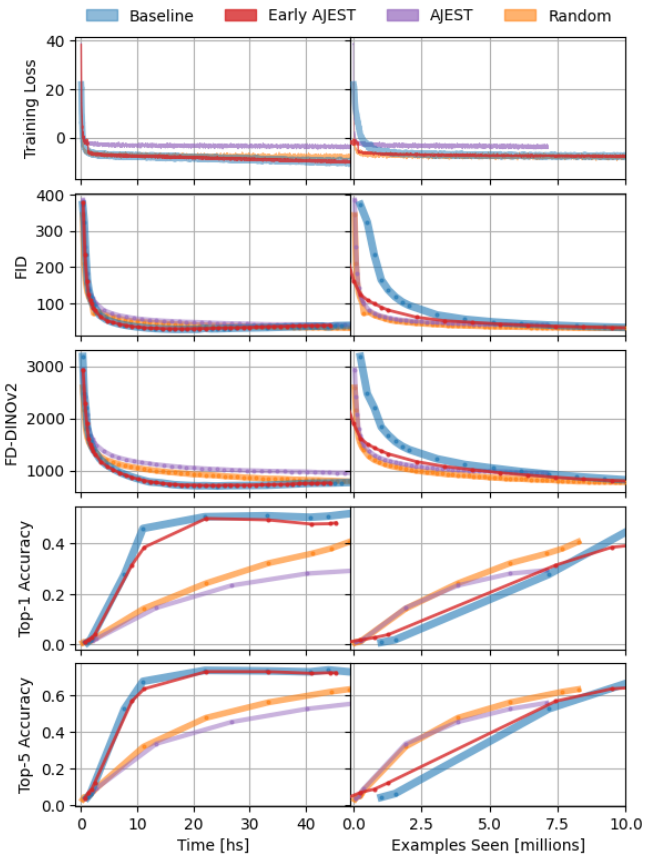

Figure 11. Training loss and validation metrics on Tiny ImageNet for EMA=0.10 with no guidance.

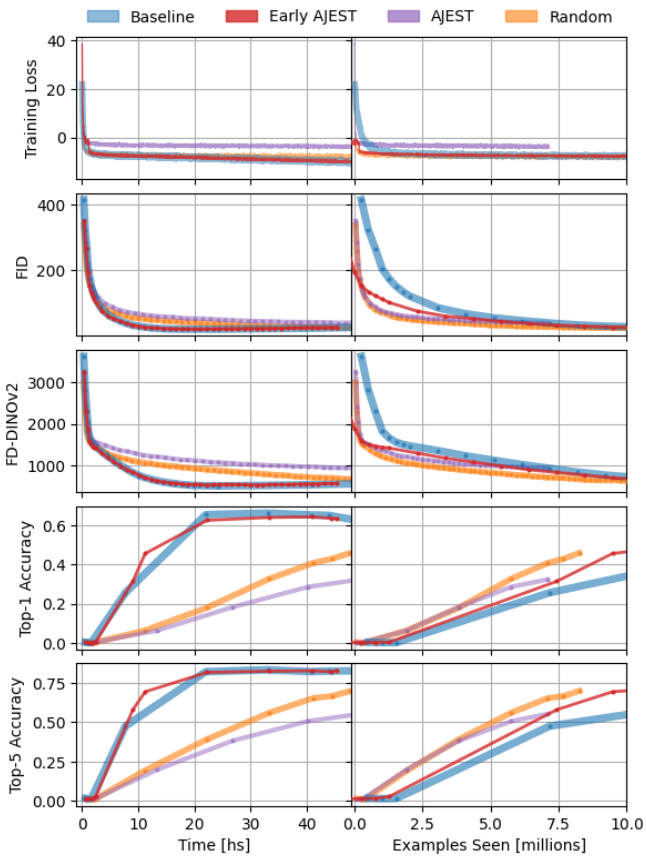

Figure 12. Training loss and validation metrics on Tiny ImageNet for EMA=0.10 with guidance weight $\alpha = 2.2$.

Table 7. Evaluation metrics for models trained for 4096 iterations on the 2D tree task. Results are averaged over 5 runs with different random seeds, with standard deviation indicating uncertainty. Bold yellow values represent the best scores for each metric across guided and unguided settings. Bold black values highlight scores statistically indistinguishable from the best (within uncertainty).

Comparison	Method	Average Loss (Unguided)		L2 Distance (All)		L2 Distance (External)		Mandala Score		Classification Score	
		All Branches	External Branches	Unguided	Guided	Unguided	Guided	Unguided	Guided	Unguided	Guided
Autoguidance	**Autoguidance**	**0.0111 ±0.003**	**0.223 ±0.007**	0.0180 ±0.004	0.207 ±0.220	0.4827 ±0.003	0.603 ±0.187	**0.63 ±0.05**	**0.82 ±0.06**	**0.89 ±0.02**	**0.94 ±0.01**
Early	AJEST	0.0108 ±0.003	**0.222 ±0.007**	0.0170 ±0.003	0.106 ±0.014	0.4830 ±0.004	0.516 ±0.019	**0.63 ±0.05**	**0.81 ±0.04**	**0.89 ±0.02**	**0.93 ±0.01**
	iAJEST	**0.0112 ±0.003**	**0.223 ±0.007**	**0.0162 ±0.002**	0.108 ±0.020	0.4823 ±0.004	0.521 ±0.031	**0.63 ±0.05**	**0.82 ±0.03**	**0.88 ±0.02**	**0.92 ±0.01**
Late	AJEST	0.0138 ±0.003	**0.229 ±0.007**	0.0176 ±0.003	**0.100 ±0.004**	0.4819 ±0.002	**0.510 ±0.009**	**0.57 ±0.05**	0.76 ±0.06	0.86 ±0.02	0.92 ±0.01
	iAJEST	0.0129 ±0.003	**0.228 ±0.007**	0.0196 ±0.004	0.114 ±0.024	0.4819 ±0.002	0.522 ±0.027	**0.62 ±0.06**	0.78 ±0.06	**0.86 ±0.02**	**0.93 ±0.01**
Full	AJEST	0.0138 ±0.003	0.231 ±0.010	0.0181 ±0.004	**0.101 ±0.004**	0.4814 ±0.002	**0.508 ±0.009**	**0.57 ±0.06**	0.74 ±0.06	**0.86 ±0.02**	0.92 ±0.01
	iAJEST	0.0131 ±0.004	**0.229 ±0.010**	0.0186 ±0.004	**0.101 ±0.004**	**0.4801 ±0.003**	**0.502 ±0.013**	**0.61 ±0.06**	0.78 ±0.06	**0.88 ±0.02**	0.92 ±0.01
Random	Random	0.0124 ±0.002	0.231 ±0.005	0.0192 ±0.003	**0.101 ±0.005**	0.4819 ±0.001	**0.508 ±0.005**	0.47 ±0.06	0.72 ±0.07	**0.87 ±0.02**	**0.93 ±0.01**

This loss is computed both over the entire data distribution and specifically on the external branches of the 2D tree, which correspond to regions of lower data density and higher generation difficulty. Lower loss values indicate better alignment with the true data distribution.

L2 Distance. To assess the quality of generated samples, we compute the average Euclidean (L2) distance between fully denoised samples produced by the model, $\mathbf{x}_j$, and samples from the ground truth distribution, $\mathbf{x}_j^{\text{gt}}$ obtained with the same random seed. This metric is reported both for the full distribution and for the external branches, providing insight into the model's ability to capture both the core and the periphery of the data manifold. We also report a "guided" L2 distance, where the model's outputs are generated with guidance (e.g., autoguidance or classifier-free

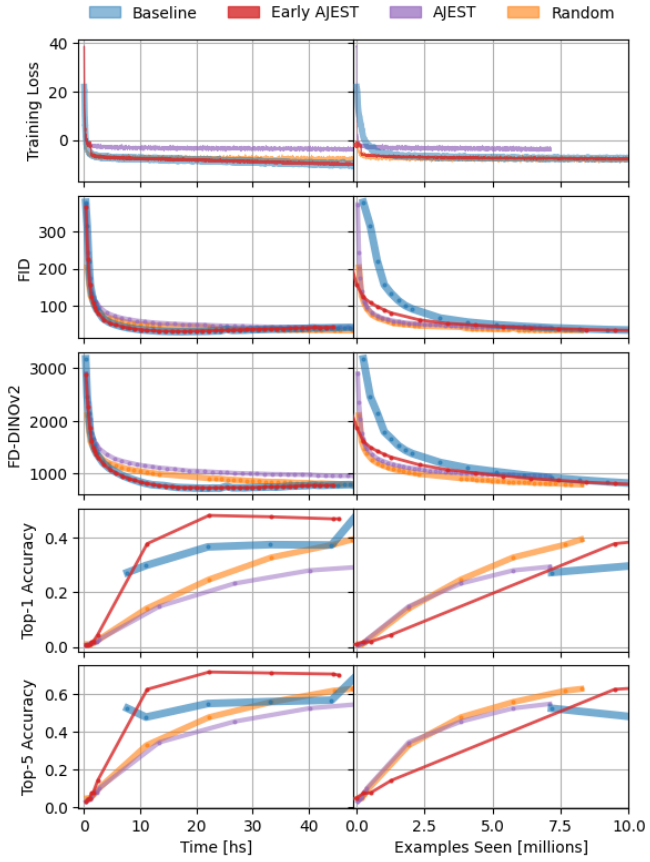

Figure 13. Training loss and validation metrics on Tiny ImageNet for EMA=0.05 with no guidance.

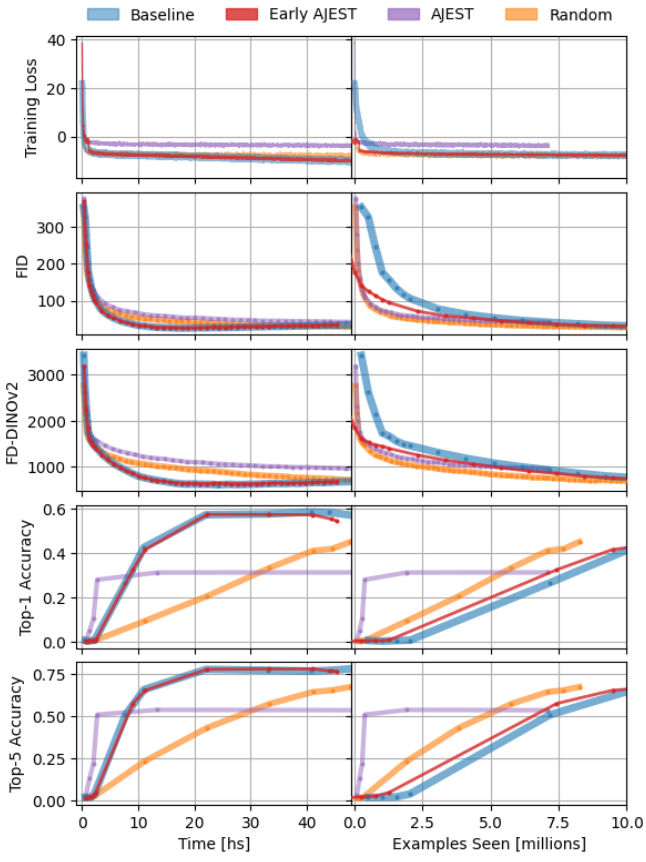

Figure 14. Training loss and validation metrics on Tiny ImageNet for EMA=0.05 and guidance weight $\alpha = 1.7$.

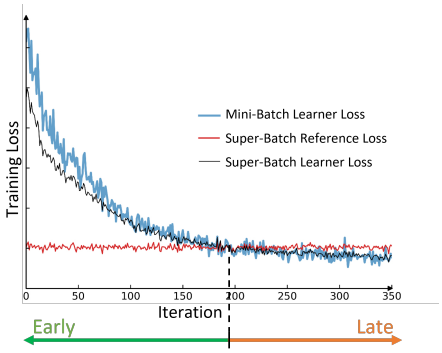

Figure 15. Illustration of how Early and Late data selection strategies are applied during training. Early in training, the reference is better than the learner, so the super-batch reference loss is smaller than the super-batch learner loss. Once the learner becomes better than the reference, the situation is inverted.

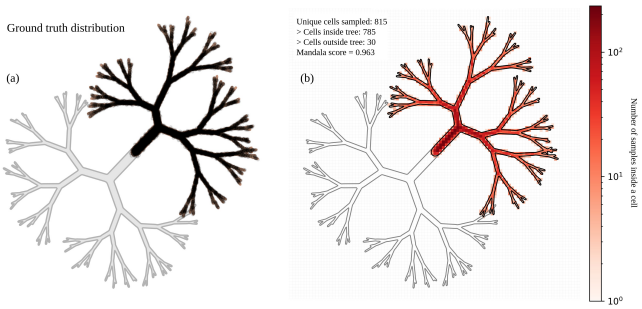

Figure 16. Single-run mandala score calculation for the 2D tree ground truth data distribution: (a) $2^{14} = 16384$ data points are generated as plotted on the left; (b) a section of the 2D space that completely covers the data distribution is discretized into a grid with $K = 100^2$ cells; the number of data points belonging to each cell is counted and plotted on the right.

guidance) during sampling. The L2 distance is given by:

$$\text{L2 Distance} = \frac{1}{M} \sum_{j=1}^{M} \left\| \mathbf{x}_j - \mathbf{x}_j^{\text{gt}} \right\| \tag{8}$$

where M is the number of generated samples, $\mathbf{x}_j$ is a generated (denoised) sample, and $\mathbf{x}_j^{\text{gt}}$ is the corresponding ground truth sample.

Mandala Score. To measure the diversity and coverage of the generated samples, we introduce the "mandala score". The 2D data space is discretized into a grid of K cells. Let $\mathcal{C}_{\text{gt}}$ denote the set of grid cells covered by the ground truth distribution, and let $\mathcal{C}_{\text{gen}}$ denote the set of 100×100 grid cells that contain at least one generated sample. The mandala score is then defined as

$$\text{Mandala Score} = \frac{|\mathcal{C}_{\text{gt}} \cap \mathcal{C}_{\text{gen}}|}{|\mathcal{C}_{\text{gt}}|} \tag{9}$$

where $|\cdot|$ denotes the cardinality of a set. A higher mandala score indicates better coverage and diversity, while a lower score suggests mode collapse or insufficient exploration of the data space.

Classification Metric. We also report a classification-based metric, specifically the binary accuracy. Let y_j be the true class label for sample j and $\hat{y}_j$ the predicted class label. The binary accuracy is defined as:

$$\text{Accuracy} = \frac{1}{M} \sum_{j=1}^{M} \mathbb{I}(\hat{y}_j = y_j) \tag{10}$$

where $\mathbb{I}(\cdot)$ is the indicator function and M is the number of samples. Higher accuracy indicates that the model generates samples that are both realistic and class-consistent.

C.2. Metrics for image generation in Tiny ImageNet

Quality evaluation. Following Karras et al. [18], we apply two commonly used metrics to assess the quality of the generated images. We calculate the Fréchet inception distance (FID) [12] and the Fréchet distance (FD) using DINOv2 [26] features, as done by Stein et al. [32]. We follow Karras et al.'s terminology and call this last metric FD-DINOv2.

We generate 2000 images equally distributed between classes to use 10 images per class, using always the same set of random seeds. We then use a pretrained InceptionV3 [33] and DINOv2 [26] to extract image features. We compare these features to those obtained with the same models applied to all images from Tiny ImageNet's training dataset.

Classification-based evaluation. To avoid any biases imposed by the use of a single family of metrics, we apply a pretrained classifier to the same 2000 generated images and calculate the Top-1 and Top-5 accuracy when predicting the ground-truth label out of the 200 classes available on Tiny ImageNet.

Huynh et al. [14] fine-tuned a Swin-L transformer on Tiny ImageNet, achieving a Top-1 accuracy of 91.35% on Tiny ImageNet classification. The Swin-L model had previously been pre-trained on ImageNet-21k [21].

D. Licences

- Autoguidance code [18]
  ↪ Creative Commons BY-NC-SA 4.0 license
- EDM2 models [19]
  ↪ Creative Commons BY-NC-SA 4.0 license
- InceptionV3 model [33]
  ↪ Apache 2.0 license

- DINOv2 model [26]
  ↪ Apache 2.0 license
- Tiny ImageNet Swin-L classifier [14]
  ↪ Apache 2.0 license
- Tiny ImageNet dataset [20, 23, 38]
  ↪ Custom non-commercial license from ImageNet [7]