

APPENDIX

A EDIFY IMAGE

The noising process is described as a weighted interpolation between the clean image $\mathbf{x}_1$ and the standard Gaussian noise $\mathbf{x}_0^{(0)} \sim \mathcal{N}(0, I^{(0)})$ (note this is not a residual) corresponding to the largest scale, where we get a noisy state $\mathbf{x}_t^{(k)}$ for scale $k = 0, 1, 2$ via:

$$\mathbf{x}_t^{(k)} = (1 - t)\text{Down}(\mathbf{x}_0^{(0)}, 2^k) + \mu^{(k)}(\mathbf{x}_1, t), \quad (10)$$

where we use a factor 2^k for better compatibility of sampling across multiple scales, and we provide more discussions on this in the supplementary.

Based on the noisy target $\mathbf{x}_t^{(k)}$, we construct the following ground-true velocity:

$$\begin{aligned} \mathbf{u}_t^{(k)}(\mathbf{x}_t^{(k)}|\mathbf{x}_1) &\triangleq \frac{d\mathbf{x}_t^{(k)}}{dt} \\ &= -\text{Down}(\mathbf{x}_0^{(0)}, 2^k) + \frac{d\mu^{(k)}(\mathbf{x}_1, t)}{dt}, \end{aligned} \quad (11)$$

where the last term is crucial because the data mean $\mu^{(k)}(\mathbf{x}_1, t)$ changes through timestep t . Based on this velocity, the model is trained through the following flow-matching target (Lipman et al., 2023):

$$\mathbb{E}_{t \in [T_{k+1}, 1], q(\mathbf{x}_1), p_t(\mathbf{x}_t^{(k)}|\mathbf{x}_1)} \left\| \mathbf{v}_t^{(k)}(\mathbf{x}_t^{(k)}) - \mathbf{u}_t^{(k)}(\mathbf{x}_t^{(k)}|\mathbf{x}_1) \right\|^2, \quad (12)$$

where t is sampled from $[T_{k+1}, 1]$ (we define $T_3 \triangleq 0$, and $T_0 \triangleq 1$), $p_t(\mathbf{x}_t^{(k)}|\mathbf{x}_1)$ is the underlying conditional probability path towards $\mathbf{x}_1$ determined by $\mathbf{u}_t^{(k)}(\mathbf{x}_t^{(k)}|\mathbf{x}_1)$, and $\mathbf{v}_t^{(k)}(\mathbf{x}_t^{(k)})$ is the multi-scale machine learning model to model the flow.

B PYRAMIDAL FLOW

Algorithm 3 Pyramidal Flow Training - 2

- 1: **Input:** Dataset of images D , number of stages K , number of epochs M , list of K start and end times s_k, e_k
 - 2: **Output:** Trained flow model $\mathbf{v}(\mathbf{x}, t)$.
 - 3: Initialize the model weights randomly
 - 4: **for** epoch= 1 to M **do**
 - 5: **for** clean data $\mathbf{x}_1$ in D **do**
 - 6: Determine the stage k for $\mathbf{x}_1$
 - 7: Sample a timestep $t \sim U(s_k, e_k)$
 - 8: Sample the start point $\mathbf{x}_0 \sim \mathcal{N}(0, I)$
 - 9: Down-sample noise $\mathbf{x}_0^{(k)} = \text{Down}(\mathbf{x}_0, 2^k)$
 - 10: Compute start points as $\mathbf{x}_s^{(k)} = s_k \cdot \text{Up}(\text{Down}(\mathbf{x}_1, 2^{k+1})) + (1 - s_k) \cdot \mathbf{x}_0^{(k)}$
 - 11: Compute endpoints as $\mathbf{x}_e^{(k)} = e_k \cdot \text{Down}(\mathbf{x}_1, 2^k) + (1 - e_k) \cdot \mathbf{x}_0^{(k)}$
 - 12: Compute the midpoints as $\mathbf{x}_t^{(k)} = \frac{e_k - t}{e_k - s_k} \cdot \mathbf{x}_s^{(k)} + \frac{t - s_k}{e_k - s_k} \cdot \mathbf{x}_e^{(k)}$
 - 13: Define the target as $\mathbf{u}_t(\mathbf{x}_t^{(k)}|\mathbf{x}_1) = \frac{\mathbf{x}_e^{(k)} - \mathbf{x}_s^{(k)}}{e_k - s_k}$
 - 14: Calculate loss as $\text{MSE}(\mathbf{u}_t(\mathbf{x}_t^{(k)}|\mathbf{x}_1), \mathbf{v}(\mathbf{x}_t^{(k)}))$
 - 15: Run back-propagation, update parameters of $\mathbf{v}$
 - 16: **end for**
 - 17: **end for**
 - 18: **Return:** $\mathbf{v}$
-

Algorithm 4 Pyramidal Flow Sampling - 2

```

1: Input: Trained flow model  $\mathbf{v}$ , number of stages  $K$ , list of  $K$  start times  $s_k$ , total number of
   timesteps  $N$ .
2: Output: Generated image.
3: Sample the starting point  $\hat{\mathbf{x}}_0^{K-1} \sim \mathcal{N}(0, I \cdot \frac{1}{4^{K-1}})$ 
4: Sample Gaussian noise at maximum resolution  $\mathbf{x}_0 \sim \mathcal{N}(0, I)$ 
5: for stage  $k = K - 1$  to 0 do
6:    $\hat{\mathbf{x}}_{s_{k-1}}^{(k-1)} = \text{ODEINT}(\mathbf{v}_t^{(k)}, \{s_k, s_{k-1}\}, \hat{\mathbf{x}}_{s_k}^{(k)})$ 
7:   Define the re-noising factor  $n_k = (1 - s_{k-1}) (\text{Down}(\hat{\mathbf{x}}_0, 2^{k-1}) - \text{Up}(\text{Down}(\hat{\mathbf{x}}_0, 2^k)))$ 
8:   Define the next starting point  $\hat{\mathbf{x}}^{(k-1)} = \text{Up}(\hat{\mathbf{x}}^{(k)}) + n_k$ 
9: end for
10: Return:  $\mathbf{x}_{K-1}$ 

```

Algorithm 5 EdifyImage Flow Matching Training

```

1: Input: Dataset of images  $D$ , number of epochs  $M$ , critical time points  $T_1$  and  $T_2$ , where
    $0 < T_2 < T_1 < 1$ 
2: Output: Trained flow model  $\mathbf{v}_t^{(0)}, \mathbf{v}_t^{(1)}, \mathbf{v}_t^{(2)}$ 
3: Initialize the model weights
4: for epoch = 1 to  $M$  do
5:   for  $\mathbf{x}_1$  in  $D$  do
6:     Sample the stage  $k$  from a uniform distribution over  $\{0, 1, 2\}$ 
7:     Sample  $t$  from a uniform distribution over  $[T_{k+1}, 1]$  where  $T_0 = 1$  and  $T_3 = 0$ 
8:     Sample the start point as random Gaussian  $\mathbf{x}_0^{(0)} \sim \mathcal{N}(0, I^{(0)})$ 
9:     Compute noisy image  $\mathbf{x}_t^{(k)} = (1 - t)\text{Down}(\mathbf{x}_0^{(0)}, 2^k) + \mu^{(k)}(\mathbf{x}_1, t)$ 
10:    Compute velocity target  $\mathbf{u}_t^{(k)}(\mathbf{x}_t^{(k)}|\mathbf{x}_1) = -\text{Down}(\mathbf{x}_0^{(0)}, 2^k) + \frac{d\mu^{(k)}(\mathbf{x}_1, t)}{dt}$ 
11:    Forward the model to get  $\mathbf{v}_t^{(k)}(\mathbf{x}_t^{(k)})$ 
12:    Calculate loss  $\text{MSE}(\mathbf{v}_t^{(k)}(\mathbf{x}_t^{(k)}), \mathbf{u}_t^{(k)}(\mathbf{x}_t^{(k)}|\mathbf{x}_1))$ 
13:    Run back-propagation, update network parameters
14:   end for
15: end for
16: Return: Trained flow model  $\mathbf{v}_t^{(2)}, \mathbf{v}_t^{(1)}, \mathbf{v}_t^{(0)}$ 

```

B.1 SAMPLING IN PYRAMID FLOW MATCHING

Sampling from a multiscale Pyramidal Flow model is sequential, starting from stage $K - 1$ and ending at stage 0, where K is the number of scales used in training. For each stage, two timesteps are defined: the start time s_k and the end time e_k . In this section, we assume the stages to have zero time

Algorithm 6 EdifyImage Flow Matching Sampling

```

1: Input: Trained flow model  $\mathbf{v}_t^{(0)}, \mathbf{v}_t^{(1)}, \mathbf{v}_t^{(2)}$ , standard Gaussian noise of the largest scale  $\mathbf{x}_0^{(0)}$ ,
   critical time points  $T_1$  and  $T_2$ 
2: Output: Largest resolution sample  $\hat{\mathbf{x}}_1^{(0)}$ 
3:  $\hat{\mathbf{x}}_{T_2}^{(2)} = \text{ODEINT}(\mathbf{v}_t^{(2)}, \{0, T_2\}, \text{Down}(\mathbf{x}_0^{(0)}, 4))$ 
4:  $\hat{\mathbf{x}}_{T_2}^{(1)} = \text{Up}(\hat{\mathbf{x}}_{T_2}^{(2)}) + (1 - T_2)(\text{Down}(\mathbf{x}_0^{(0)}, 2) - \text{Up}(\text{Down}(\mathbf{x}_0^{(0)}, 4)))$  # This is the re-noise step
5:  $\hat{\mathbf{x}}_{T_1}^{(1)} = \text{ODEINT}(\mathbf{v}_t^{(1)}, \{T_2, T_1\}, \hat{\mathbf{x}}_{T_2}^{(1)})$ 
6:  $\hat{\mathbf{x}}_{T_1}^{(0)} = \text{Up}(\hat{\mathbf{x}}_{T_1}^{(1)}) + (1 - T_1)(\mathbf{x}_0^{(0)} - \text{Up}(\text{Down}(\mathbf{x}_0^{(0)}, 2)))$ 
7:  $\hat{\mathbf{x}}_1^{(0)} = \text{ODEINT}(\mathbf{v}_t^{(0)}, \{T_1, 1\}, \hat{\mathbf{x}}_{T_1}^{(0)})$ 
8: Return:  $\hat{\mathbf{x}}_1^{(0)}$ 

```

Table 4: **Training details across datasets and resolutions.** We summarize the hyperparameters and training configurations used for our experiments on different benchmarks.

Training Details	CelebA-HQ 256	CelebA-HQ 512	CelebA-HQ 1024	ImageNet 256
Backbone	DiT-L/2	DiT-L/2	DiT-L/2	DiT-B/2 & DiT-XL/2
Largest latent dimensions	32×32	64×64	128×128	32×32
Batch size	256	256	128	256
Init LR	$2e-4$	$2e-4$	$2e-4$	$1e-4$
LR schedule	CosineAnnealingLR	CosineAnnealingLR	CosineAnnealingLR	N/A
Final LR	$1e-6$	$1e-6$	$1e-6$	N/A
Optimizer	AdamW	AdamW	AdamW	AdamW
Training epochs	500	1000	2000	Up to 1400
Time segments	0.5	0.33, 0.67	0.33, 0.67	0.5
Number of scales	2	3	3	2
VAE type	EQVAE	SDVAE	SDVAE	EQVAE
EMA decay rate	N/A	N/A	N/A	0.9999
Hardware	$8 \times H200$	$8 \times H200$	$8 \times H200$	$8 \times H200$

overlap, that is, $s_{k-1} = e_k$. Also, the stages together span the whole time interval, that is, $s_{K-1} = 0$ and $e_0 = 1$. The noise is sampled once in the largest dimension $n \sim \mathcal{N}(0, I)$. Within each stage the sampling process follows the standard flow path between

$$\mathbf{x}_s = s_{k-1} \cdot \text{Up}(\text{Down}(\mathbf{x}, 2^k)) + (1 - s_{k-1}) \cdot \text{Down}(n, 2^{k-1})$$

, and we have

$$\mathbf{x}_e = e_{k-1} \cdot \text{Down}(\mathbf{x}, 2^{k-1}) + (1 - e_{k-1}) \cdot \text{Down}(n, 2^{k-1}).$$

The jump between consecutive stages needs to be handled carefully to enforce the consistency of the flow. Consider the jump from stage k to stage $k-1$. The procedure starts with up-sampling the endpoint of stage k , yielding

$$\text{Up}(\mathbf{x}_e) = s_{k-1} \text{Up}(\text{Down}(x, 2^k)) + (1 - s_k) \text{Up}(\text{Down}(n, 2^k))$$

The distribution of this data differs from the distribution of the startpoint $\mathbf{x}_s^{k-1}$ of the $k-1$ stage.

$\text{Down}(x, 2^{k-1}) \sim \mathcal{N}(0, I \cdot \frac{1}{2^{k-1}})$, and similarly $\text{Down}(x, 2^k) \sim \mathcal{N}(0, I \cdot \frac{1}{2^k})$, however the up-sampled noise $\text{Up}(\text{Down}(n, 2^k)) \sim \mathcal{N}(0, \Sigma \cdot \frac{1}{2^k})$ where Σ is a block-diagonal matrix that has 2×2 blocks of ones on the diagonal and zeroes elsewhere. Then if we define a different block-diagonal matrix Σ' with blocks

$$\Sigma'_{\text{block}} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

and set $\alpha = \sqrt{\frac{3}{4^k}}$, and sample $m \sim \mathcal{N}(0, \Sigma')$, we have $\text{Var}(\text{Up}(\mathbf{x}_e^k) + (1 - s_{k-1})\alpha \cdot m) = \text{Var}(\mathbf{x}_s^{k-1})$. In practice this means that at the end of stage k we sample $m \sim \mathcal{N}(0, \Sigma')$ and set $\mathbf{x}_s^{k-1} = \text{Up}(\mathbf{x}_e^k) + (1 - s_{k-1})\alpha \cdot m$ as the starting point of stage $k-1$.

C MORE TRAINING DETAILS

We present the detailed training settings in Table 4, highlighting key implementation differences across datasets and resolutions. For CelebA-HQ, we utilize a DiT-L/2 backbone across all resolutions while adjusting the number of scales based on complexity—two scales for 256×256 and three scales for higher resolutions. For ImageNet, we experiment with both DiT-B/2 and DiT-XL/2 backbones to demonstrate scaling properties. Notably, we use EQVAE for our lower-resolution models due to its superior reconstruction quality, while higher-resolution CelebA-HQ models leverage SDVAE for better stability. All models were trained on $8 \times H200$ GPUs, with batch sizes adjusted according to memory constraints.

D LIMITATIONS

First, we didn't train our XL model on ImageNet for $7M$ steps due to a restriction of computational resources. This limitation may have prevented our model from reaching its full potential in comparison to diffusion models that are typically trained for significantly longer periods. Second, our approach still requires storing part of the model weights for each scale, which increases memory requirements during training, although our MoT design partially mitigates this issue. Third, while our method shows excellent performance on unconditional and conditional generation tasks, we have not yet extended it to conditional generation scenarios such as text-to-image synthesis or inpainting, which are important applications for generative models. Finally, our method inherits limitations from the VAE architecture used for latent space projection, potentially limiting the fidelity of fine details in generated images. Future work could address these limitations by exploring more efficient parameter-sharing techniques across scales, developing conditional variants of our approach, implementing memory-efficient sampling strategies, and investigating improved latent space representations.

E IMPACT STATEMENT

This paper presents Laplacian Multi-scale Flow Matching (LapFlow), a novel approach for generative image modeling. We believe this work has several potential positive impacts on the research community and society:

Our method improves computational efficiency in high-resolution image generation, potentially reducing the energy consumption and carbon footprint associated with training and deploying generative models. By requiring fewer function evaluations and reduced GFLOPs, LapFlow contributes to more sustainable AI development.

The technical innovations in multi-scale representations and unified transformer architectures may inspire new approaches to other generative tasks beyond images, such as audio, video, and 3D content creation. The proposed mixture-of-transformers architecture with parameter sharing could influence more efficient model designs across various domains.

However, we acknowledge that advances in generative image modeling also come with potential societal concerns. Like other generative models, this technology could be misused to create misleading or deceptive content. We encourage responsible development and deployment of systems based on our research, including content safeguards and transparent disclosure of AI-generated media.

Our work focuses on advancing the technical capabilities of generative models rather than specific applications. We emphasize that any practical implementation should consider ethical implications and be developed in accordance with responsible AI principles.

REFERENCES

- Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.
- Yuval Atzmon, Maciej Bala, Yogesh Balaji, Tiffany Cai, Yin Cui, Jiaojiao Fan, Yunhao Ge, Siddharth Gururani, Jacob Huffman, Ronald Isaac, Pooya Jannaty, Tero Karras, Grace Lam, J. P. Lewis, Aaron Licata, Yen-Chen Lin, Ming-Yu Liu, Qianli Ma, Arun Mallya, Ashlee Martino-Tarr, Doug Mendez, Seungjun Nah, Chris Pruett, Fitsum Reda, Jiaming Song, Ting-Chun Wang, Fangyin Wei, Xiaohui Zeng, Yu Zeng, and Qinsheng Zhang. Edify image: High-quality image generation with pixel space laplacian diffusion models, 2024. URL <https://arxiv.org/abs/2411.07126>.
- Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. Non-uniform diffusion models, 2022. URL <https://arxiv.org/abs/2207.09786>.
- Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 2018.
- Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space, 2023. URL <https://arxiv.org/abs/2307.08698>.
- 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. doi: 10.1109/CVPR.2009.5206848.
- Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=AAWuVzZaVt>.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- 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.
- William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.
- Google DeepMind. Veo 2. <https://deepmind.google/technologies/veo/veo-2/>, 2025. Accessed: 2025-05-09.
- 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 and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
- Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022.
- Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong MU, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=66NzcRQuOq>.

- Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=Hk99zCeAb>.
- Theodoros Kouzelis, Ioannis Kakogeorgiou, Spyros Gidaris, and Nikos Komodakis. Eq-vae: Equivariance regularized latent space for improved generative image modeling, 2025. URL <https://arxiv.org/abs/2502.09509>.
- Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 624–632, 2017.
- Weixin Liang, Lili Yu, Liang Luo, Srinivasan Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen-tau Yih, Luke Zettlemoyer, et al. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *arXiv preprint arXiv:2411.04996*, 2024.
- Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=PqvMRDCJT9t>.
- Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.
- Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. URL <https://openreview.net/forum?id=Skq89Scxx>.
- Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pages 23–40. Springer, 2024.
- William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.
- Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.
- Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.
- Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017.
- Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.
- Jiayan Teng, Wendi Zheng, Ming Ding, Wenyi Hong, Jianqiao Wangni, Zhuoyi Yang, and Jie Tang. Relay diffusion: Unifying diffusion process across resolutions for image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=qTlcbLSm4p>.
- Keyu Tian, Yi Jiang, Zehuan Yuan, BINGYUE PENG, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=gojL67CfS8>.
- Shiwei Zhang, Jiayu Wang, Yingya Zhang, Kang Zhao, Hangjie Yuan, Zhiwu Qin, Xiang Wang, Deli Zhao, and Jingren Zhou. I2vgen-xl: High-quality image-to-video synthesis via cascaded diffusion models. *arXiv preprint arXiv:2311.04145*, 2023.