SnapGen: Taming High-Resolution Text-to-Image Models for Mobile Devices with Efficient Architectures and Training

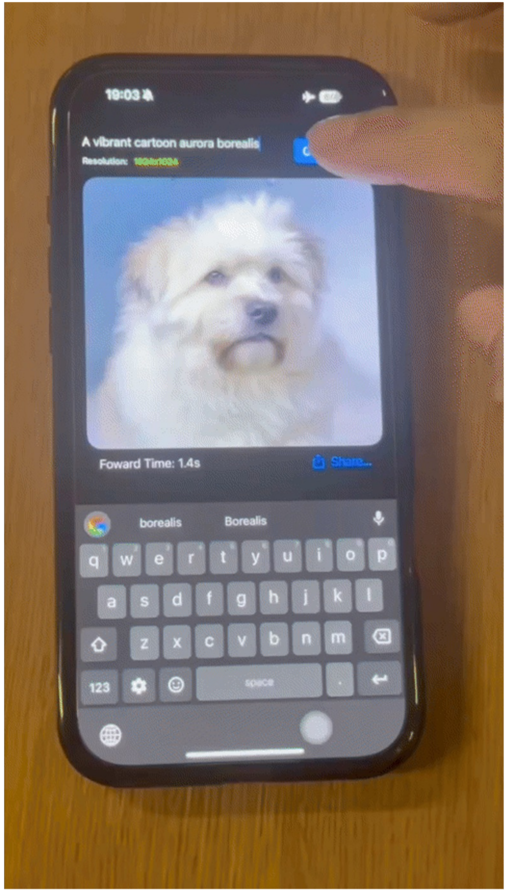
CVPR 2025 Highlight

Dongting Hu*, Jierun Chen*, Xijie Huang*, Huseyin Coskun, Arpit Sahni, Aarush Gupta, Anujraaj Goyal, Dishani Lahiri, Rajesh Singh, Yerlan Idelbayev, Junli Cao, Yanyu Li, Kwang-Ting Cheng, S.-H. Gary Chan, Mingming Gong, Sergey Tulyakov, Anil Kag, Yanwu Xu, Jian Ren

Snap Inc.

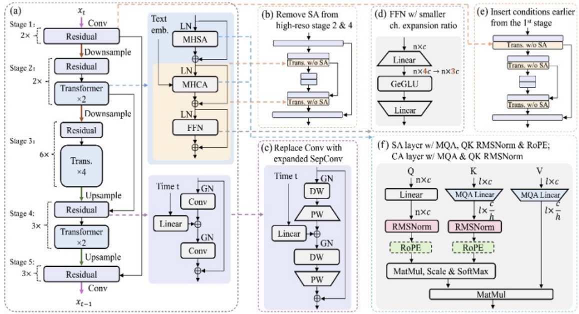
SnapGen is the first image generation model (379M) that:

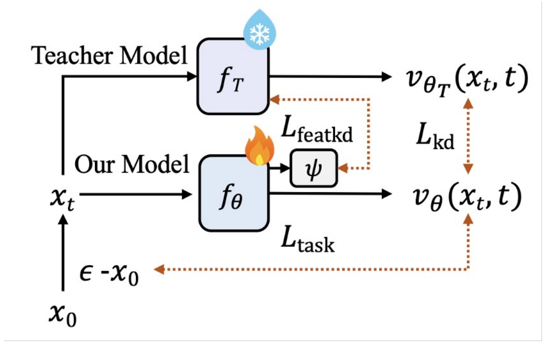
- synthesizes **high-resolution** (1024x1024) images
- runs on **mobile** devices in 1.4s
- achieves **high quality** (0.66 on GenEval).

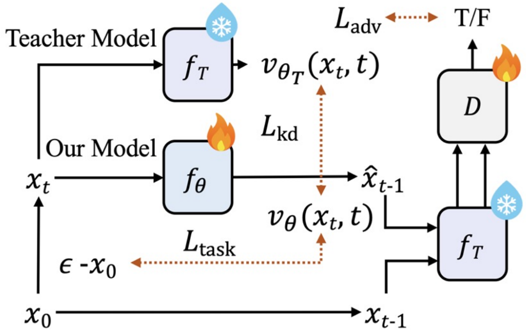


In this work we propose:

- Efficient Network Architectures
- Efficient Training Techniques
- Advanced Step Distillation

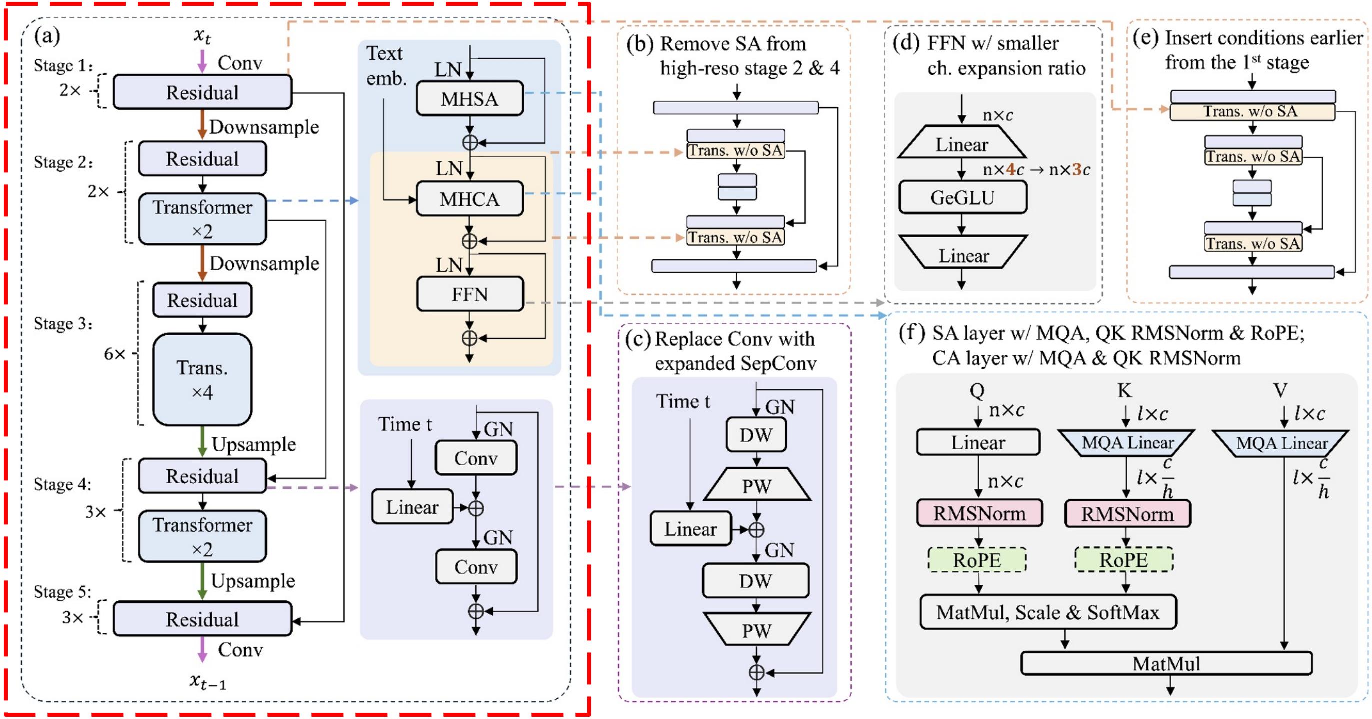






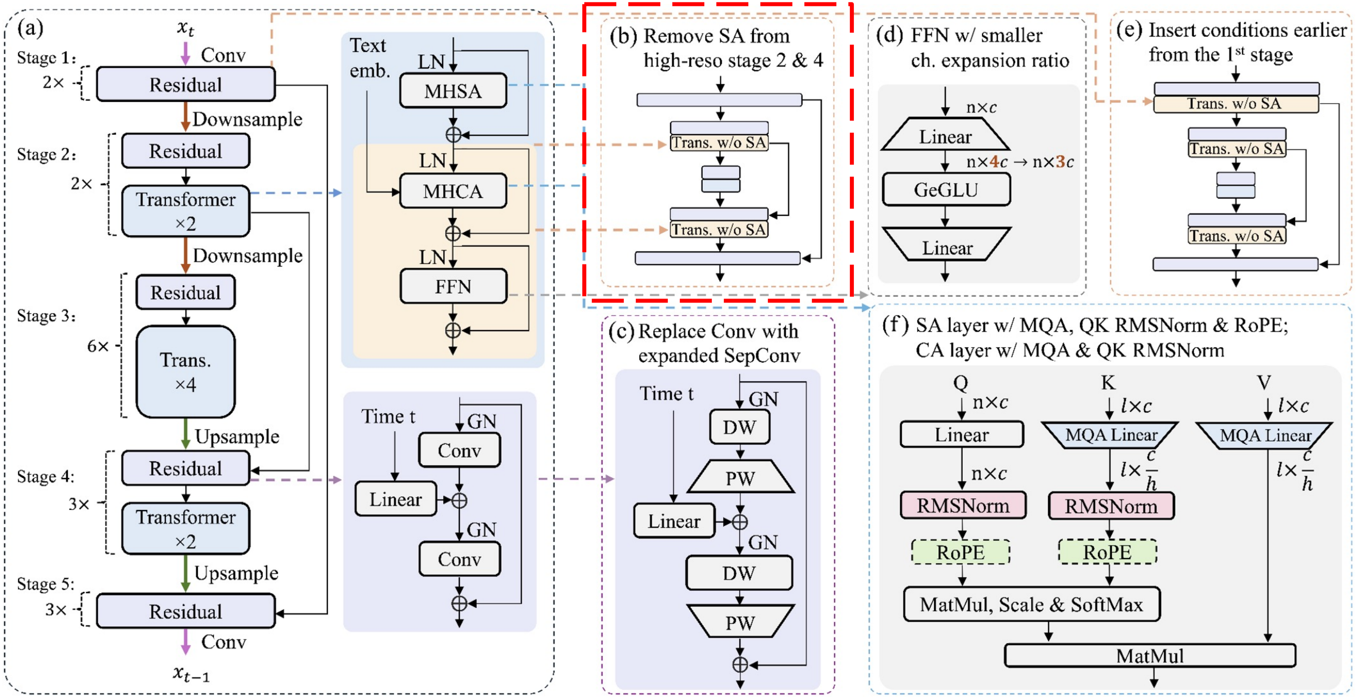
Efficient Network Architectures (Denoiser)

(a) Based on SDXL's **UNet**, we design an **efficient** architecture that maintains **high-quality** generation.



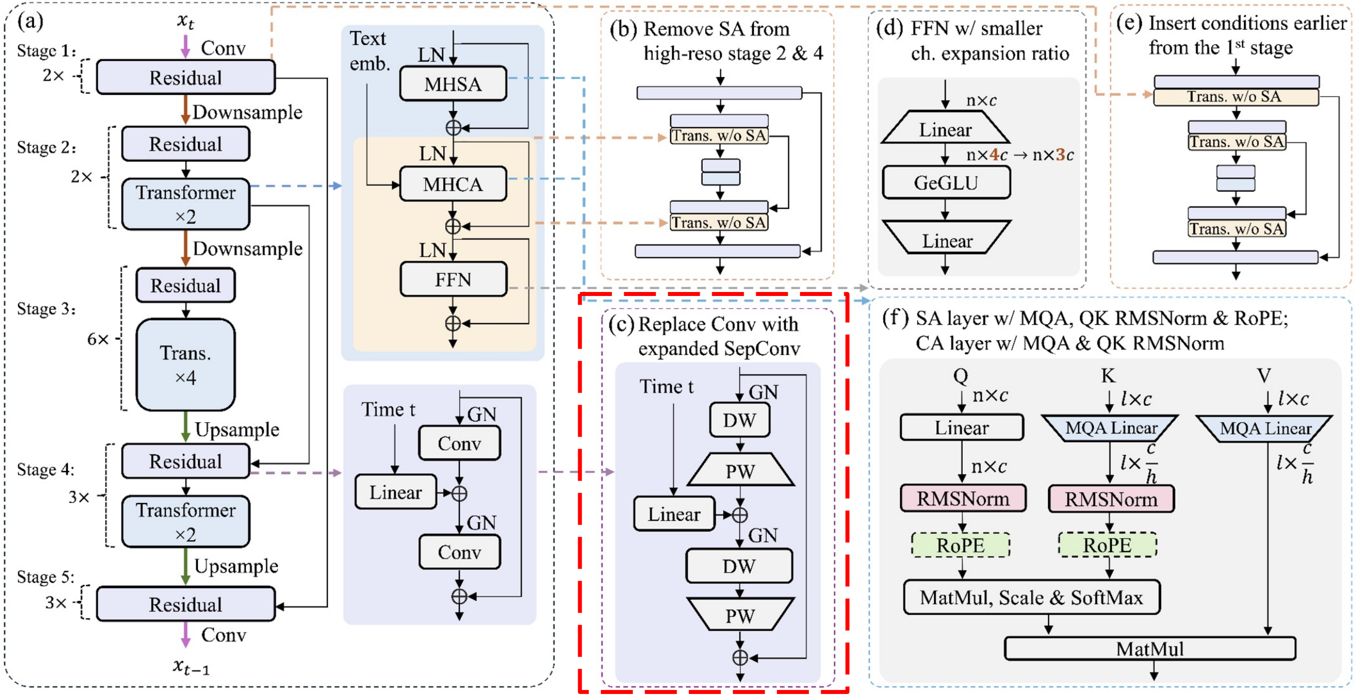
Efficient Network Architectures (Denoiser)

(b) We remove **self attention** layer from **high-resolution** stages.



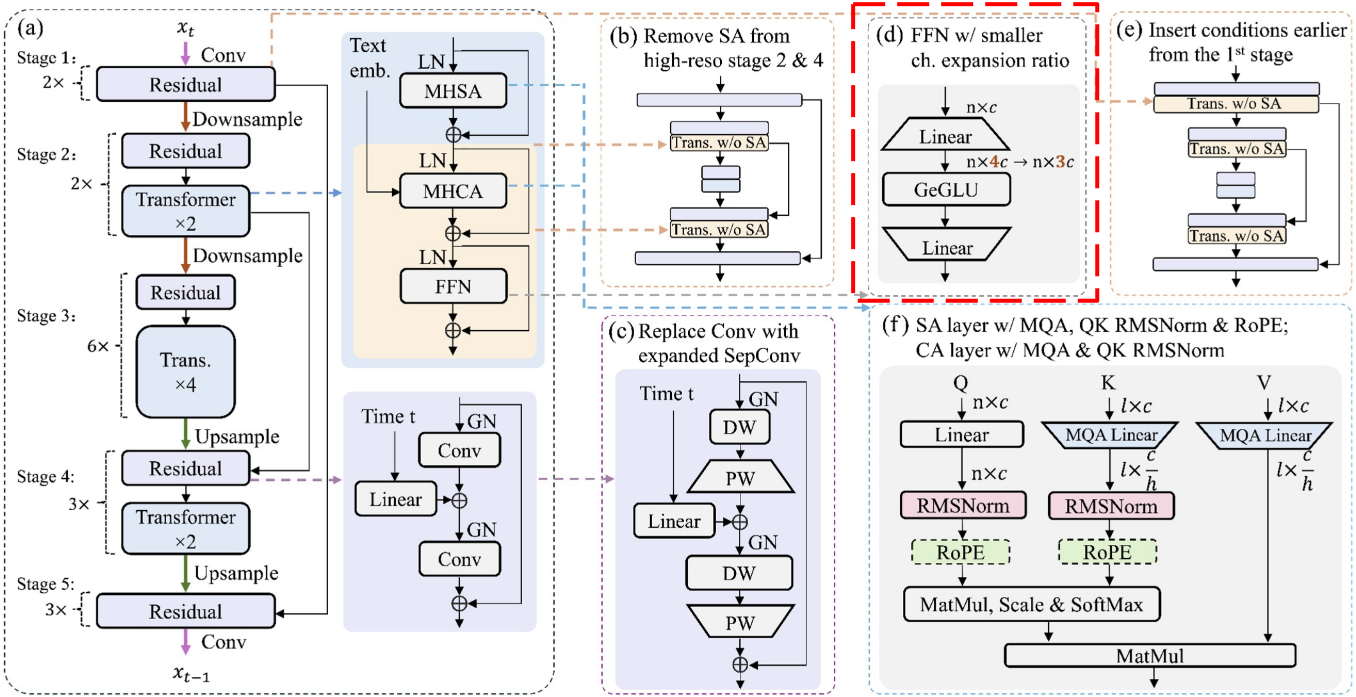
Efficient Network Architectures (Denoiser)

(c) We replace the conv in the **residual blocks** with **expanded separable convolutions**.



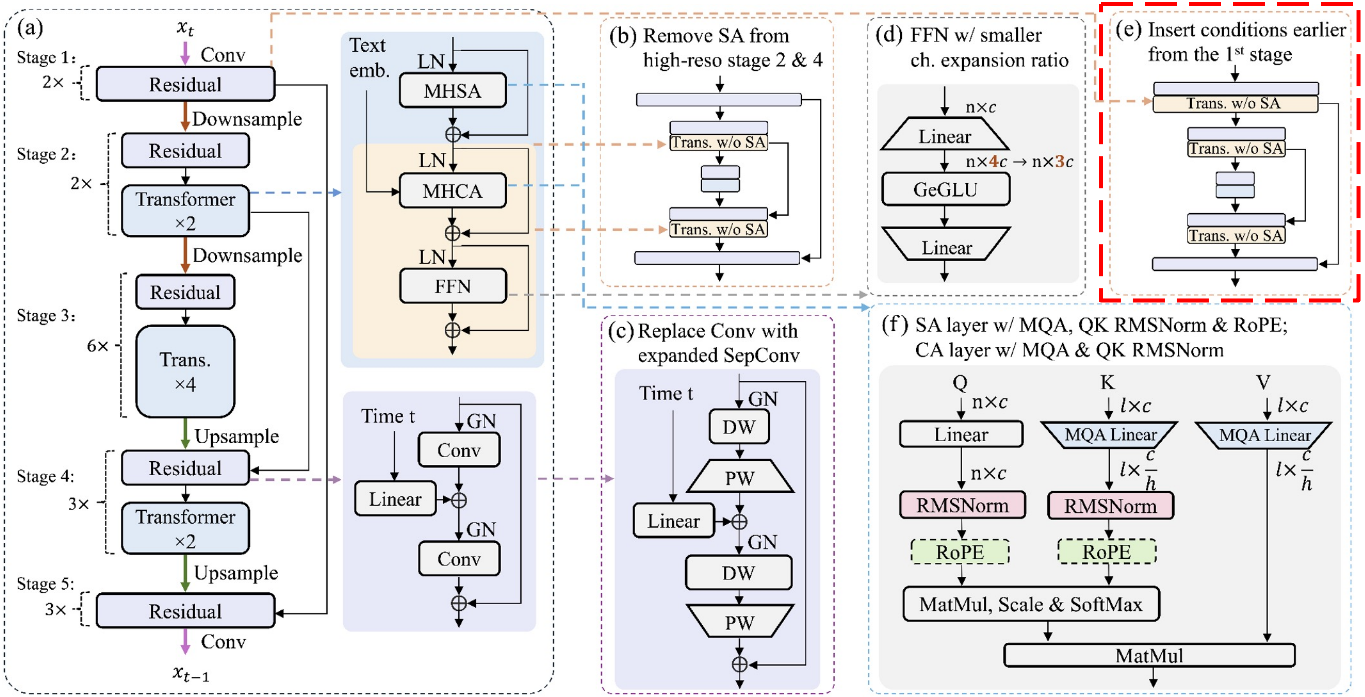
Efficient Network Architectures (Denoiser)

(d) We trim the **expansion ratio** in the transformer **feedforward** blocks.



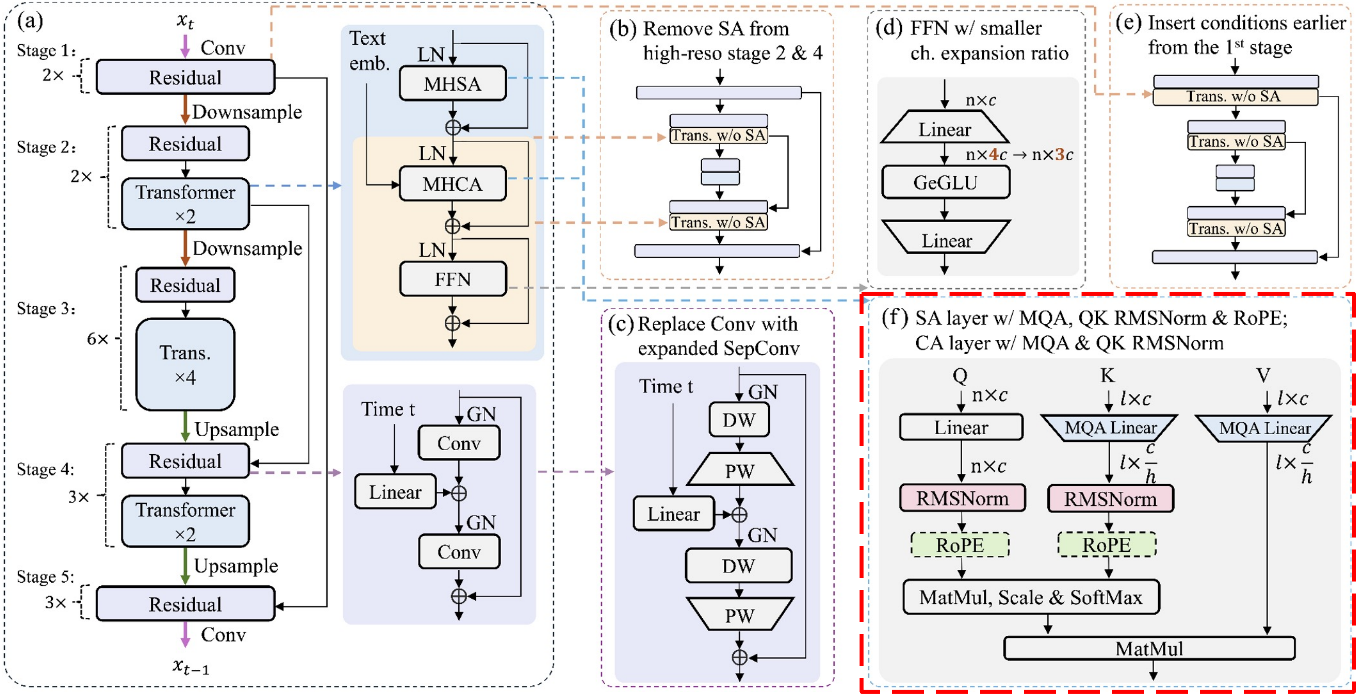
Efficient Network Architectures (Denoiser)

(e) We incorporate **cross attention** in the **first** stage.



Efficient Network Architectures (Denoiser)

(f) We replace MHSA with **MQA** and employ **QK RMSNorm** and **RoPE** Embeddings.

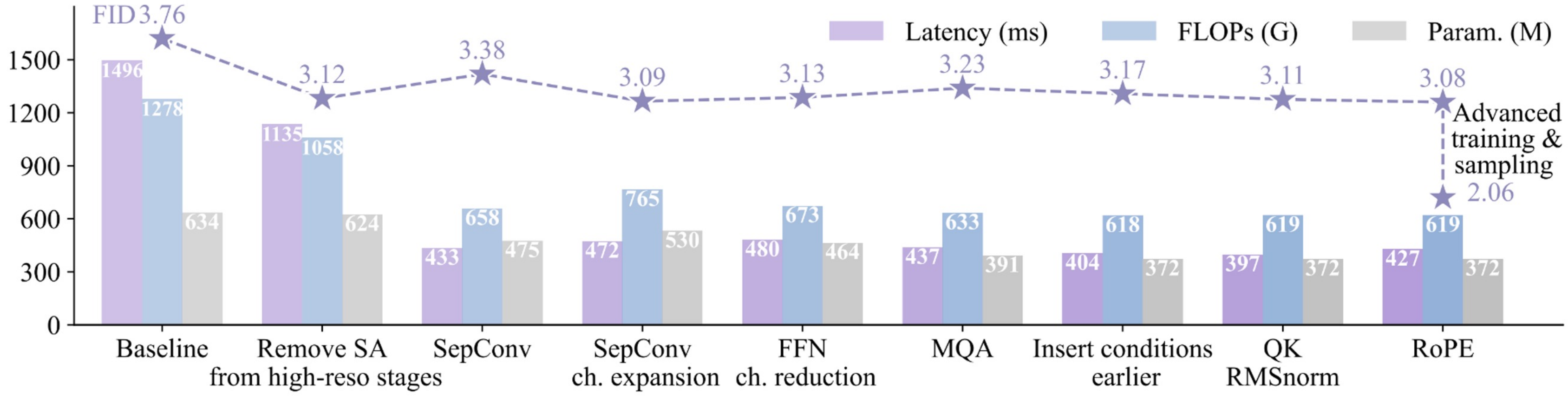


Efficient Network Architectures (Denoiser)

We obtain an efficient denoising backbone with these optimizations.

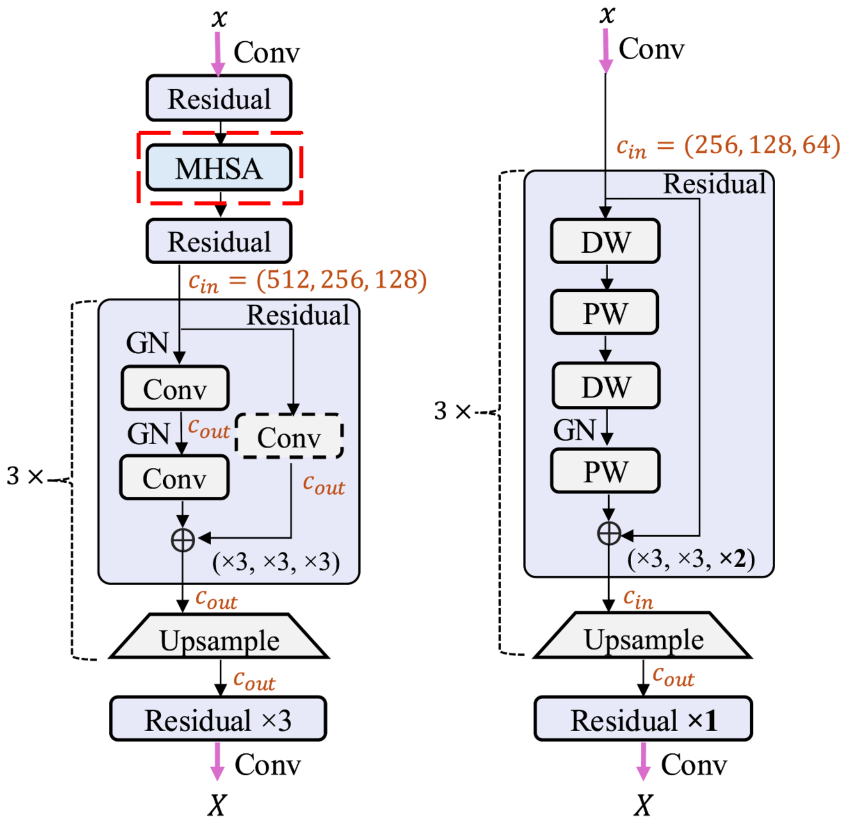
Table 1. **Class-conditional image generation on ImageNet** 256 × 256 with CFG. FLOPs are calculated for one forward pass.

Model	Param (M)	FLOPs (G)	FID↓
LDM-4 [61]	400	104	3.60
UViT-L [8]	287	77	3.40
UViT-H [8]	501	133	2.29
DiT-XL [55]	675	119	2.27
SiT-XL [52]	675	119	2.06
Ours	372	38	2.06



Efficient Network Architectures (Decoder)

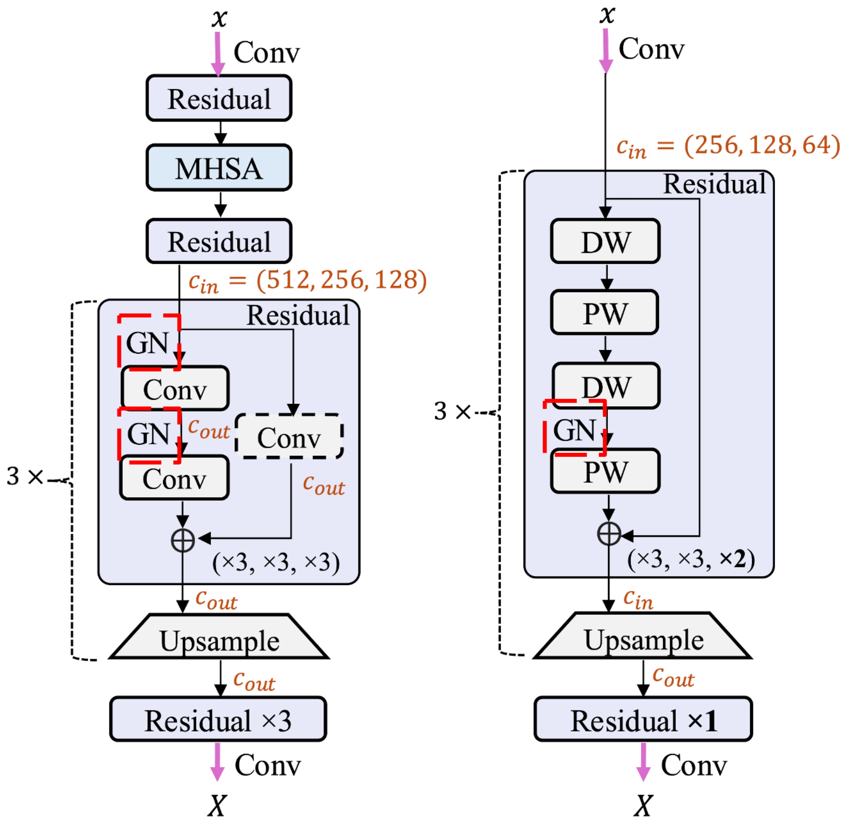
- Remove **attention** layers.



(a) SDXL/SD3 decoder (b) Our tiny decoder

Efficient Network Architectures (Decoder)

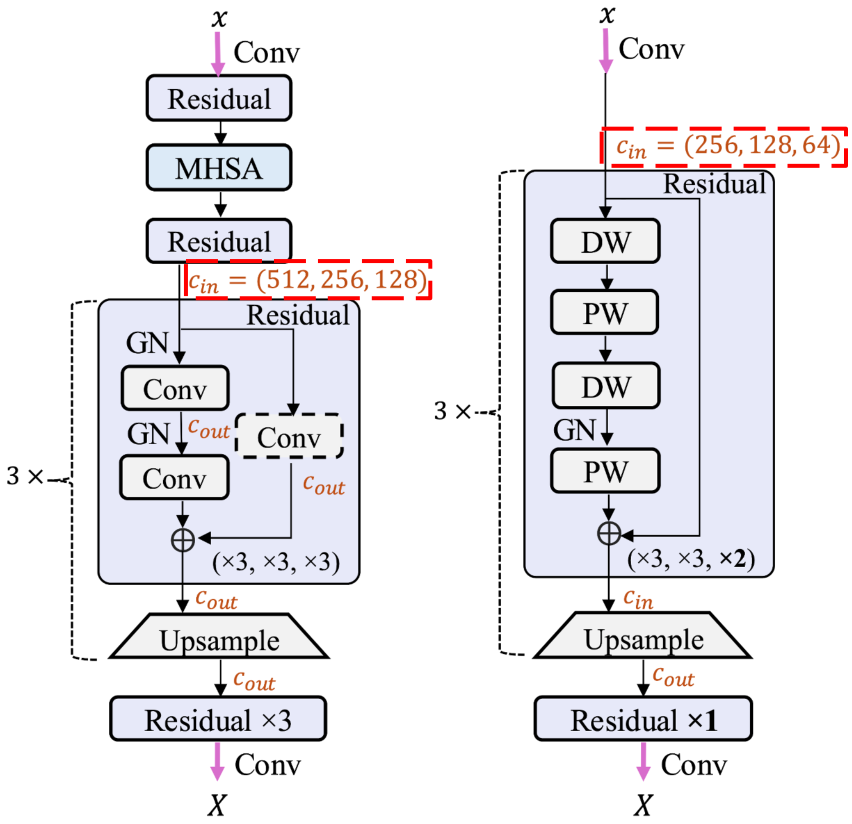
- Remove **attention** layers.
- Keep a minimal amount of **Group Norm** layer.



(a) SDXL/SD3 decoder (b) Our tiny decoder

Efficient Network Architectures (Decoder)

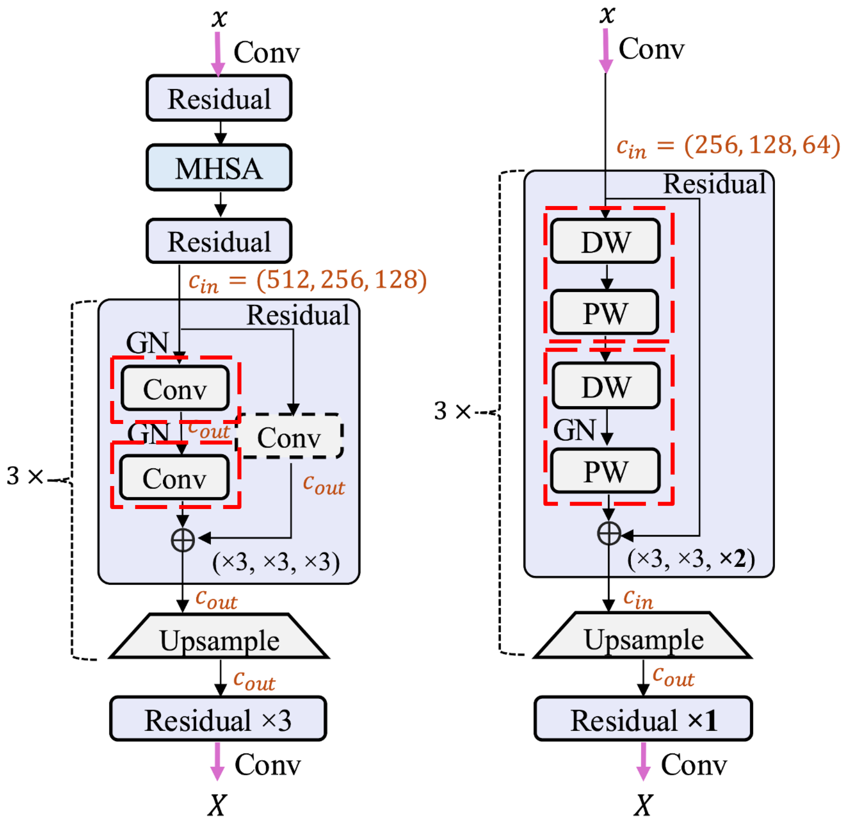
- Remove **attention** layers.
- Keep a minimal amount of **Group Norm** layer.
- Make the decoder **thinner.**



(a) SDXL/SD3 decoder (b) Our tiny decoder

Efficient Network Architectures (Decoder)

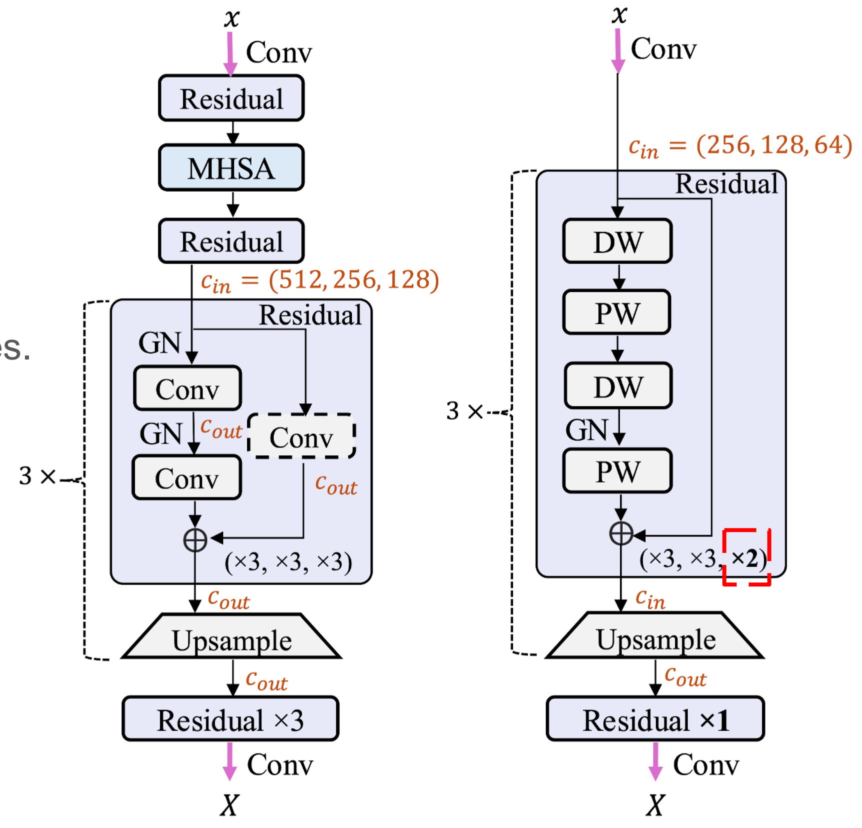
- Remove **attention** layers.
- Keep a minimal amount of **Group Norm** layer.
- Make the decoder **thinner.**
- Replace Conv with **SepConvs**.



(a) SDXL/SD3 decoder (b) Our tiny decoder

Efficient Network Architectures (Decoder)

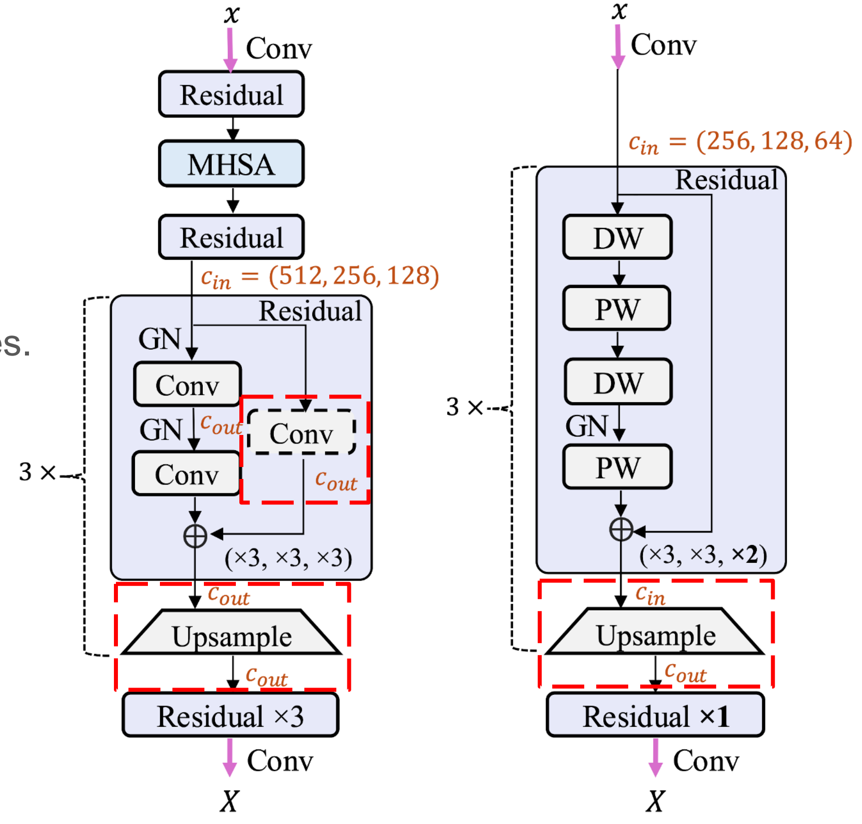
- Remove **attention** layers.
- Keep a minimal amount of **Group Norm** layer.
- Make the decoder **thinner.**
- Replace Conv with **SepConvs**.
- Use fewer **residual** blocks in **high-resolution** stages.



(a) SDXL/SD3 decoder (b) Our tiny decoder

Efficient Network Architectures (Decoder)

- Remove **attention** layers.
- Keep a minimal amount of **Group Norm** layer.
- Make the decoder **thinner.**
- Replace Conv with **SepConvs**.
- Use fewer **residual** blocks in **high-resolution** stages.
- Remove the **Conv shortcut** in residual blocks.

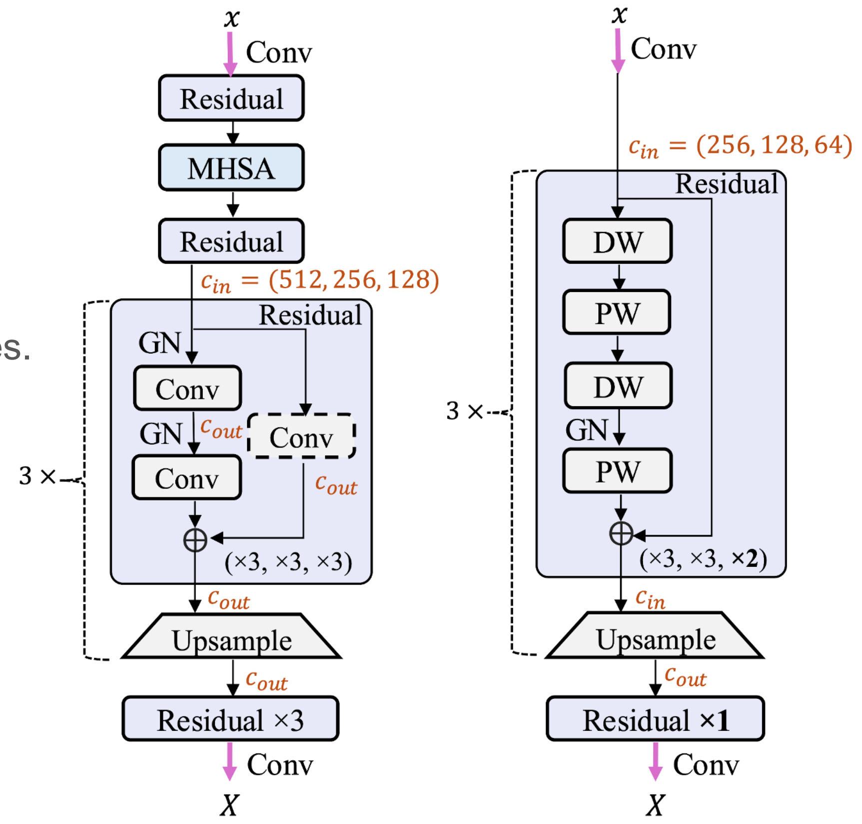


(a) SDXL/SD3 decoder (b) Our tiny decoder

Efficient Network Architectures (Decoder)

- Remove **attention** layers.
- Keep a minimal amount of **Group Norm** layer.
- Make the decoder **thinner.**
- Replace Conv with **SepConvs**.
- Use fewer **residual** blocks in **high-resolution** stages.
- Remove the **Conv shortcut** in residual blocks.

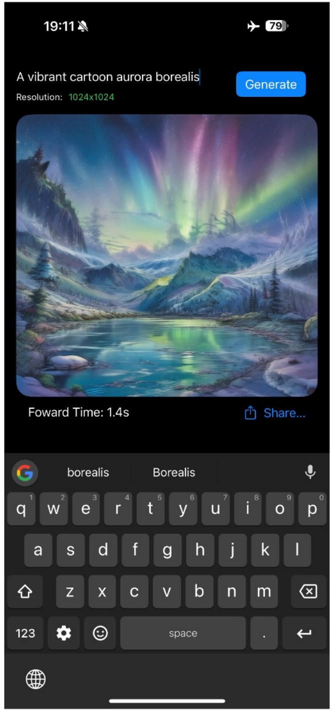
Decoder	Ch	PSNR	Param (M)	FLOPs (G)	Latency (ms) on ANE	Latency (ms) on GPU
SDXL [56]	4	24.89	49.49	4970	OOM	9469
SD3 [19]	16	27.92	49.55	4970	OOM	OOM
Ours	16	27.85	1.38	224	174	-



(a) SDXL/SD3 decoder (b) Our tiny decoder

Latency for 1024x1024 generation on iPhone 16 Pro-Max

Component	Param (M)	Latency on ANE
Tiny Decoder	1.38	119 ms
Denoiser UNet	378	274 ms
CLIP-L	123	4 ms
CLIP-G	302	23 ms
4-step Generation	-	1.4 s
8-step Generation	-	2.5 s

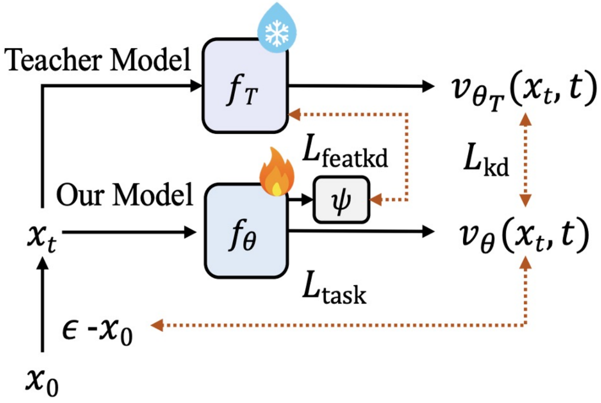


Multi-Level Knowledge Distillation

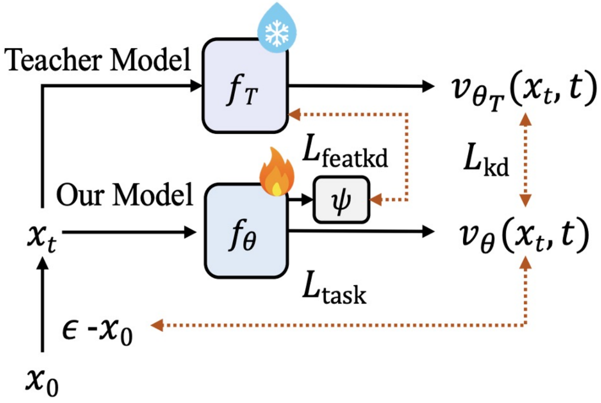
1. Teacher Model: SD3.5-Large (**heterogeneous** architecture)

Multi-Level Knowledge Distillation

1. Teacher Model: SD3.5-Large (**heterogeneous** architecture)
2. Multi-Level:
   a. Output Distillation: $\mathcal{L}_{\text{kd}} = \mathbb{E}\left[||v_{\theta_T}(x_t, t) - v_\theta(x_t, t)||_2^2 \right]$
   b. Feature Distillation: $\mathcal{L}_{\text{featkd}} = \mathbb{E}\left[\sum_{(l_T, l)} ||f_{\theta_T}^{l_T}(x_t, t) - \psi(f_\theta^l(x_t, t))||_2^2 \right]$

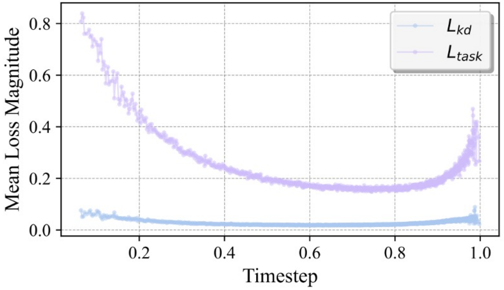


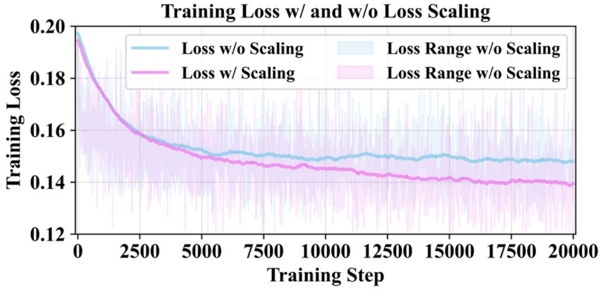
Multi-Level Knowledge Distillation



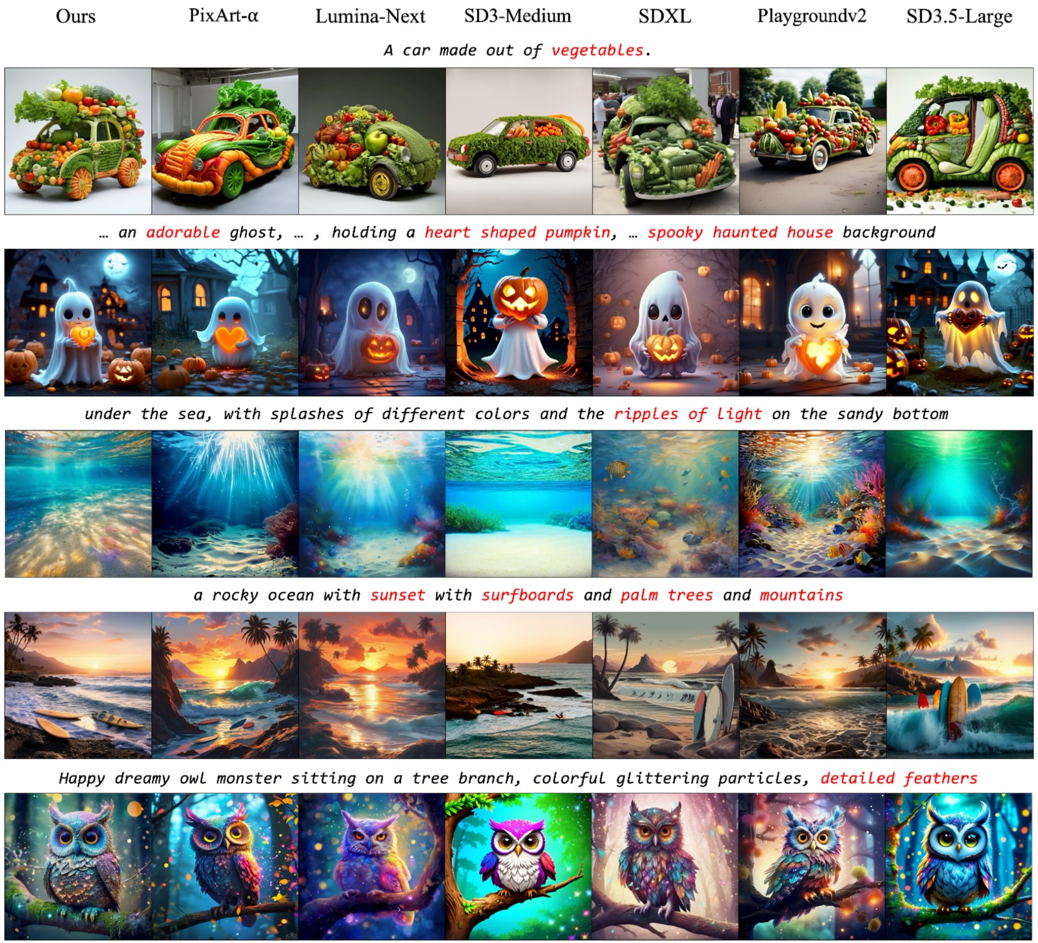
1. Teacher Model: SD3.5-Large (**heterogeneous** architecture)
2. Multi-Level:
   a. Output Distillation: $\mathcal{L}_{\text{kd}} = \mathbb{E}\left[||v_{\theta_T}(x_t, t) - v_\theta(x_t, t)||_2^2\right]$
   b. Feature Distillation: $\mathcal{L}_{\text{featkd}} = \mathbb{E}\left[\sum_{(l_T, l)} ||f_{\theta_T}^{l_T}(x_t, t) - \psi(f_\theta^l(x_t, t))||_2^2\right]$
3. Timestep-Aware Scaling: scale the loss **coefficient** w.r.t **prediction difficulty** in various **timesteps**:

$$\mathcal{S}(\mathcal{L}_{\text{task}}, \mathcal{L}_{\text{kd}}) = \mathbb{E}_t\left[\lambda(t) \cdot \mathcal{L}_{\text{task}}^t + (1 - \lambda(t)) \frac{|\mathcal{L}_{\text{task}}^t|}{|\mathcal{L}_{\text{kd}}^t|} \cdot \mathcal{L}_{\text{kd}}^t\right]$$



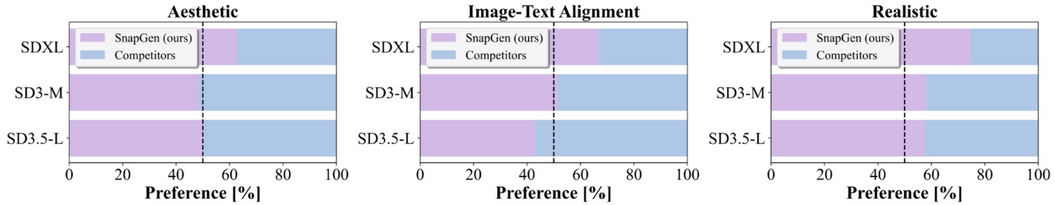


Qualitative Results



Quantitative Results

Human evaluation vs. SDXL, SD3-Medium and SD3.5-Large:



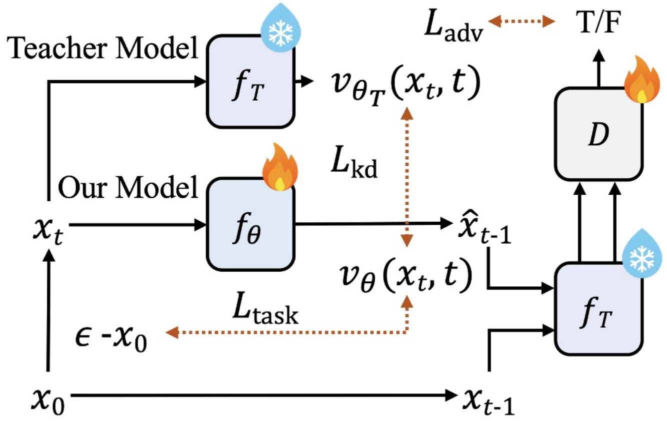
Comparison with existing T2I models across various benchmarks:

Model	Param	Throughput	GenEval ↑	DPG ↑	CLIP ↑	Image Reward ↑
PixArt-α	0.6B	0.42	0.48	71.1	0.316	1.15
PixArt-Σ	0.6B	0.46	0.53	80.5	0.317	1.13
SD-1.5	0.9B	-	0.43	63.2	0.287	0.19
SD-2.1	0.9B	-	0.50	64.2	0.281	0.29
Sana	1.6B	1.00	**0.66**	**84.8**	<u>0.327</u>	1.25
LUMINA-Next	2.0B	0.06	0.46	74.6	0.309	0.88
SDXL	2.6B	0.18	0.55	74.7	0.301	0.99
Playgroundv2	2.6B	0.18	0.59	74.5	0.317	1.25
Playgroundv2.5	2.6B	0.18	0.56	75.5	0.319	**1.34**
IF-XL	5.5B	0.06	<u>0.61</u>	75.6	0.311	0.65
Ours w/o KD	0.38B	1.04	<u>0.61</u>	76.3	0.321	1.20
SnapGen (ours)	0.38B	1.04	**0.66**	<u>81.1</u>	**0.332**	<u>1.32</u>

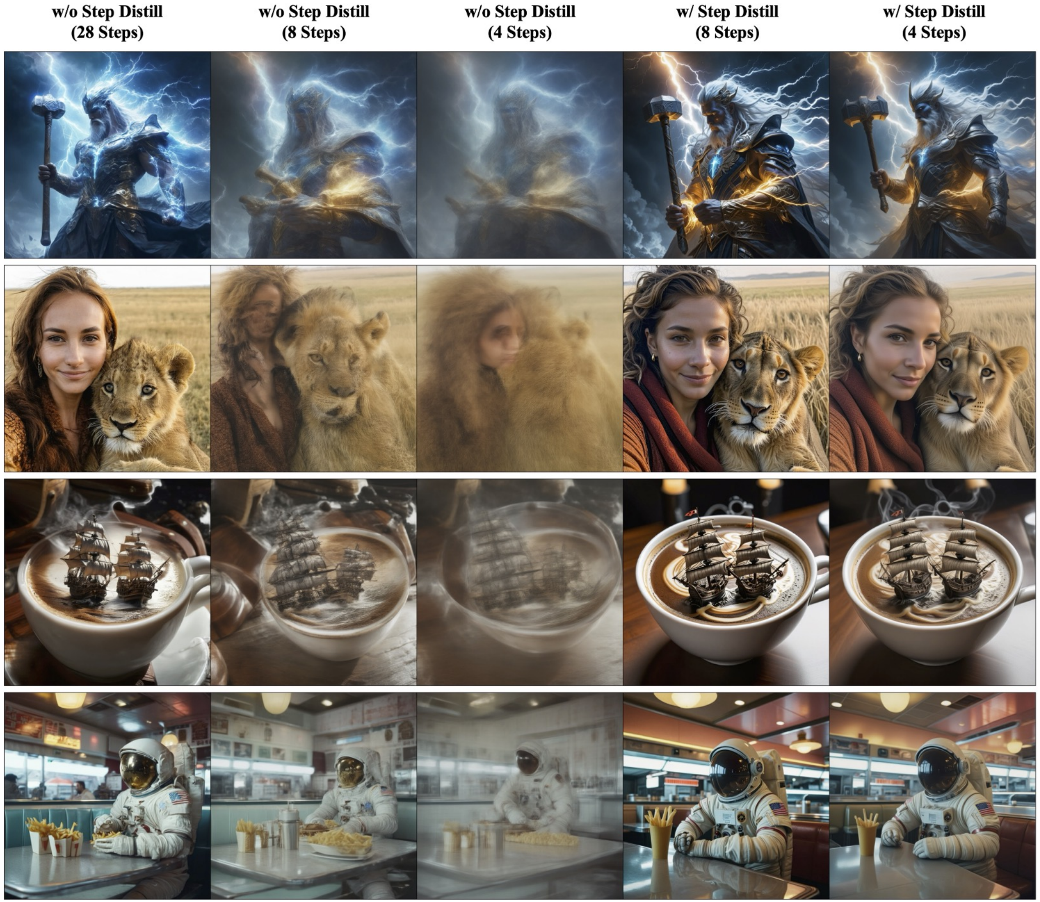
Advanced Step Distillation

1. Teacher Model: SD3.5-Large-Turbo (heterogeneous architecture)
2. Method: diffusion-GAN
3. Advanced Objective: a few-step diffusion model with adversarial refinement and knowledge distillation

$$\min_{D_{\theta_T}} \max_{G_\theta} \mathbb{E}\Big[[\log(D_{\theta_T}(x_{t-1}, t))] + [\log(1 - D_{\theta_T}(x'_{t-1}, t))] - \mathcal{S}(\mathcal{L}_{\text{task}}, \mathcal{L}_{\text{kd}})\Big]$$



Qualitative Results



Thanks for watching!