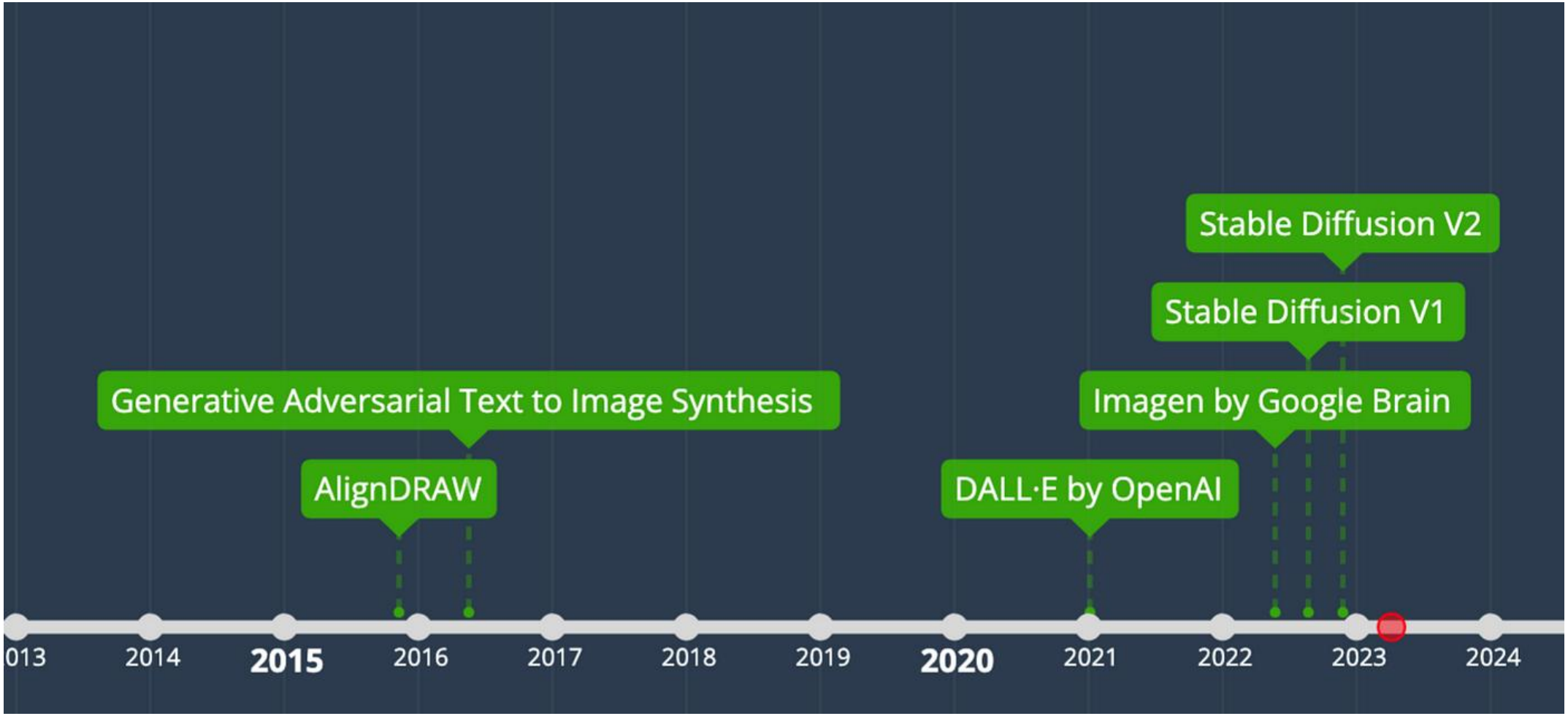
# SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis

ICLR 2024 (Spotlight)

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, Robin Rombach

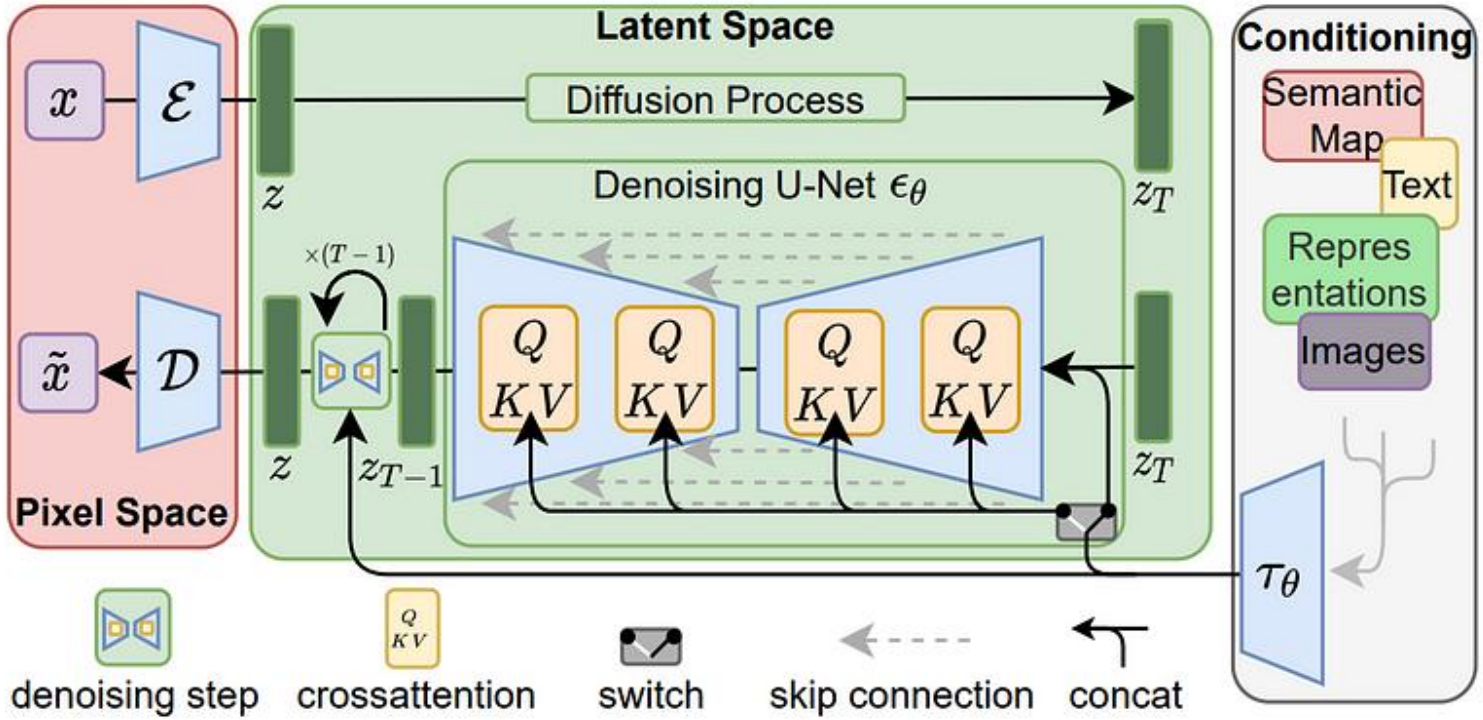
Wenjing Wang
2024.02.25

# Background


Stable Diffusion V2
Stable Diffusion V1
Generative Adversarial Text to Image Synthesis
Imagen by Google Brain
AlignDRAW
DALL·E by OpenAI
013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024

# Background

- Baseline:  Latent Diffusion



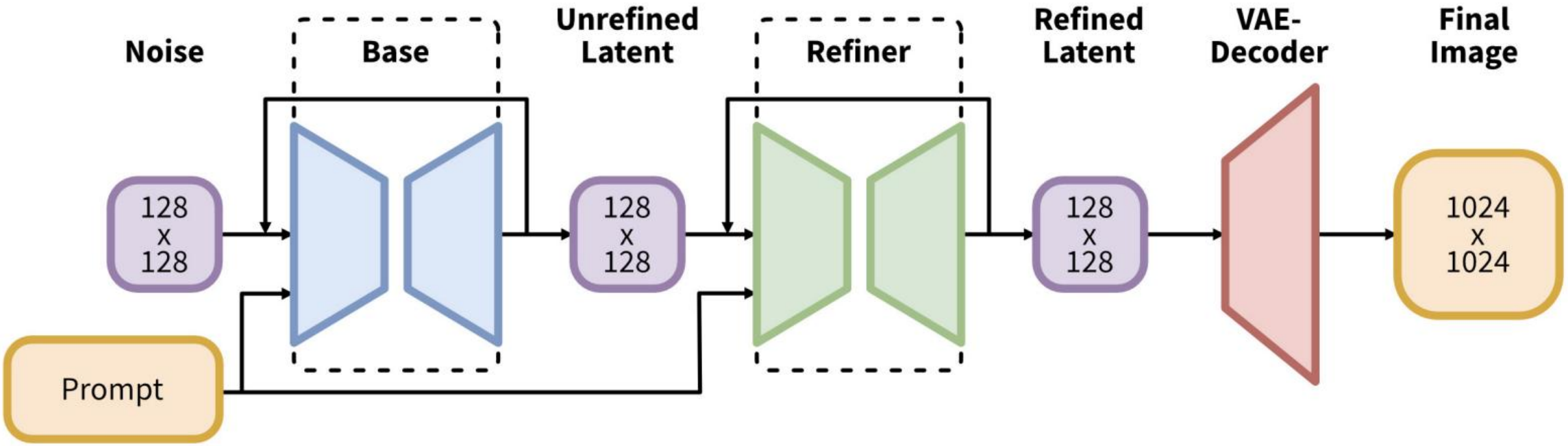

High–Resolution Image Synthesis with Latent Diffusion Models (CVPR–22)

# Background

**Models released by Stability AI**

- 07/23 SDXL 1.0
- 11/23 Stable Video Diffusion
- 11/23 SDXL Turbo
- 02/24 Stable Cascade
- 02/24 Stable Diffusion 3

# Stable Diffusion XL



# IMPROVING Stable Diffusion

## 1. Architecture & Scale

Table 1: Comparison of *SDXL* and older *Stable Diffusion* models.

| Model | *SDXL* | SD 1.4/1.5 | SD 2.0/2.1 |
|---|---|---|---|
| # of UNet params | 2.6B | 860M | 865M |
| Transformer blocks | [0, 2, 10] | [1, 1, 1, 1] | [1, 1, 1, 1] |
| Channel mult. | [1, 2, 4] | [1, 2, 4, 4] | [1, 2, 4, 4] |
| Text encoder | CLIP ViT-L & OpenCLIP ViT-bigG | CLIP ViT-L | OpenCLIP ViT-H |
| Context dim. | 2048 | 768 | 1024 |
| Pooled text emb. | OpenCLIP ViT-bigG | N/A | N/A |

- A heterogeneous distribution of transformer
- Remove the lowest level (8×downsampling)

# IMPROVING Stable Diffusion

## 1. Architecture & Scale

Table 1: Comparison of *SDXL* and older *Stable Diffusion* models.

| Model | *SDXL* | SD 1.4/1.5 | SD 2.0/2.1 |
|---|---|---|---|
| # of UNet params | 2.6B | 860M | 865M |
| Transformer blocks | [0, 2, 10] | [1, 1, 1, 1] | [1, 1, 1, 1] |
| Channel mult. | [1, 2, 4] | [1, 2, 4, 4] | [1, 2, 4, 4] |
| Text encoder | CLIP ViT-L & OpenCLIP ViT-bigG | CLIP ViT-L | OpenCLIP ViT-H |
| Context dim. | 2048 | 768 | 1024 |
| Pooled text emb. | OpenCLIP ViT-bigG | N/A | N/A |

- ViT–bigG (694M), ViT–H/14 (354M), ViT–L/14 (123M)
- Concatenate the along the channel–axis (1280+768)

# IMPROVING Stable Diffusion

## 1. Architecture & Scale

Table 1: Comparison of *SDXL* and older *Stable Diffusion* models.

| Model | *SDXL* | SD 1.4/1.5 | SD 2.0/2.1 |
|---|---|---|---|
| # of UNet params | 2.6B | 860M | 865M |
| Transformer blocks | [0, 2, 10] | [1, 1, 1, 1] | [1, 1, 1, 1] |
| Channel mult. | [1, 2, 4] | [1, 2, 4, 4] | [1, 2, 4, 4] |
| Text encoder | CLIP ViT-L & OpenCLIP ViT-bigG | CLIP ViT-L | OpenCLIP ViT-H |
| Context dim. | 2048 | 768 | 1024 |
| Pooled text emb. | OpenCLIP ViT-bigG | N/A | N/A |

- Condition: cross–attention layers + (following GLIDE) add the pooled text embedding to time embedding

# IMPROVING Stable Diffusion

2. Micro–Conditioning

- The shortcoming of the LDM: training a model requires a minimal image size
- Current solutions:
  - Discard all training images below a certain minimal resolution

    (SD 1.4/1.5 discarded all images with any size below 512 pixels)

# IMPROVING Stable Diffusion

## 2. Micro–Conditioning

- For this particular choice of data, discarding all samples below our pretraining resolution of 2562 pixels would lead to a significant 39% of discarded data.

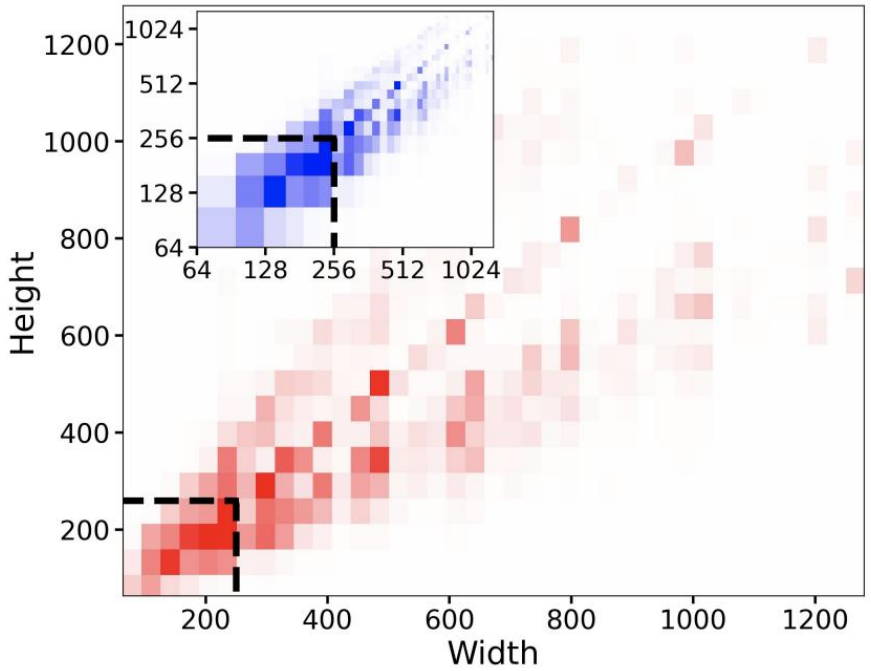
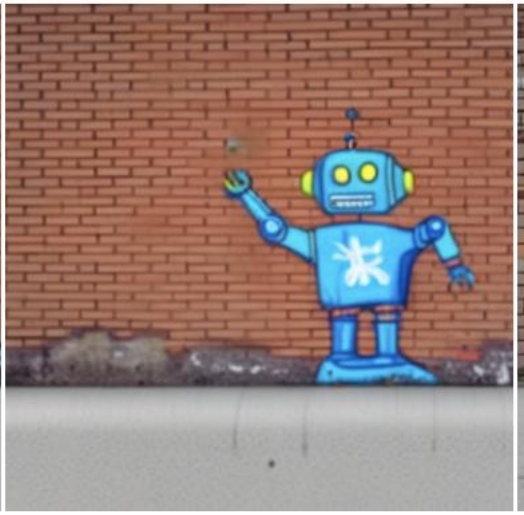
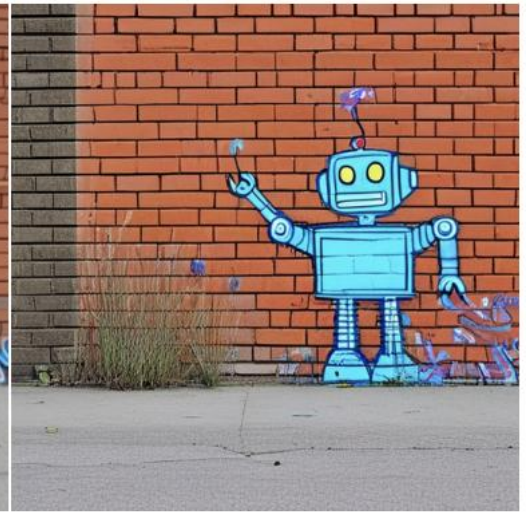



Figure 2: Height-vs-Width distribution of our pre-training dataset. Without the proposed size-conditioning, 39% of the data would be discarded due to edge lengths smaller than 256 pixels as visualized by the dashed black lines. Color intensity in each visualized cell is proportional to the number of samples.

# IMPROVING Stable Diffusion

2. Micro–Conditioning

- Solution: Size conditioning

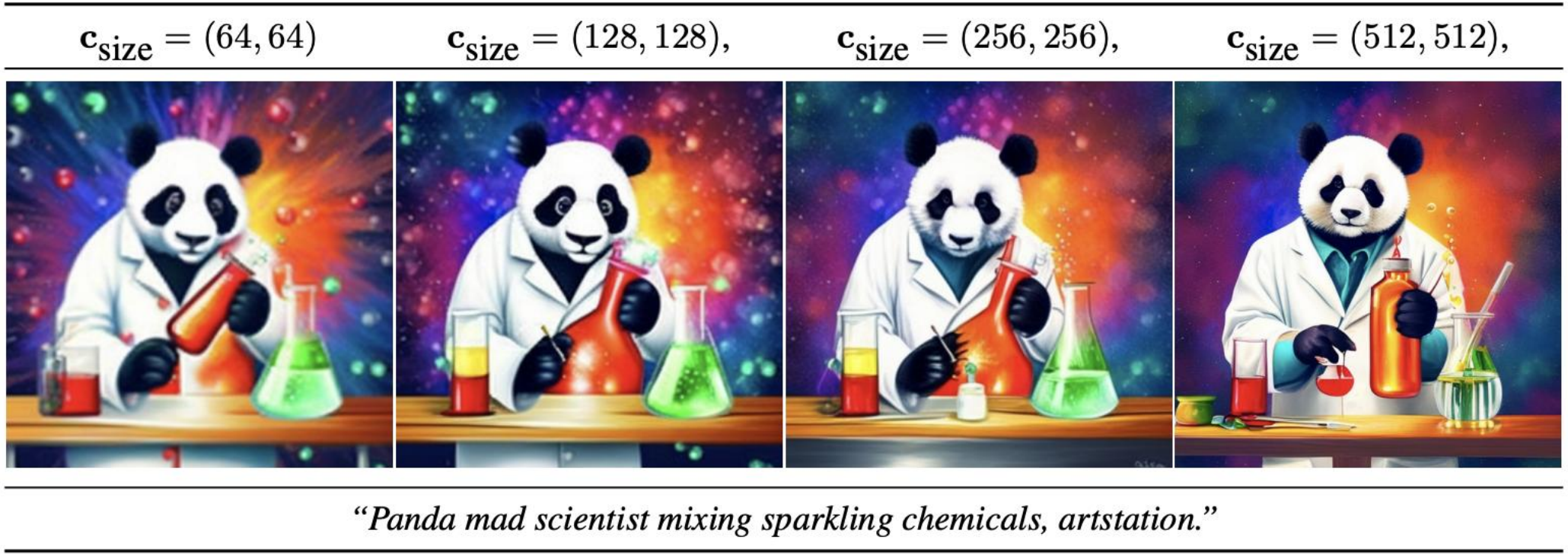
| $\mathbf{c}_{\text{size}} = (64, 64)$ | $\mathbf{c}_{\text{size}} = (128, 128)$, | $\mathbf{c}_{\text{size}} = (256, 256)$, | $\mathbf{c}_{\text{size}} = (512, 512)$, |
|---|---|---|---|
|  |  |  |  |

*"A robot painted as graffiti on a brick wall. a sidewalk is in front of the wall, and grass is growing out of cracks in the concrete."*

# IMPROVING Stable Diffusion

2. Micro–Conditioning

- Solution: Size conditioning



# IMPROVING Stable Diffusion

2. Micro–Conditioning

- Solution: Size conditioning

Table 2: Conditioning on the original spatial size of the training examples improves performance on class-conditional ImageNet Deng et al. (2009) on $512^2$ resolution.

| model | FID-5k ↓ | IS-5k ↑ |
|---|---|---|
| *CIN-512-only* | 43.84 | 110.64 |
| *CIN-nocond* | 39.76 | 211.50 |
| *CIN-size-cond* | **36.53** | **215.34** |

- 512–only: we discard all training examples with at least one edge smaller than 512 pixels what results in a train dataset of only 70k images
- nocond: we use all training examples but without size conditioning [cause blurry samples]
- FID and IS: reasonable metrics on ImageNet as the neural

# IMPROVING Stable Diffusion

## 2. Micro–Conditioning

| | *"A propaganda poster depicting a cat dressed as french emperor napoleon holding a piece of cheese."* | *"a close-up of a fire spitting dragon, cinematic shot."* |
|---|---|---|
| ***SD* 1-5** |  |  |
| ***SD* 2-1** |  |  |

# IMPROVING Stable Diffusion

2. Micro–Conditioning

- Cause:  the use of random cropping during training
- Solution: Cropping parameters
  - Uniformly sample crop coordinates $c_{top}$ and $c_{left}$ and feed them into the model as conditioning parameters via Fourier feature embeddings.
  - Concatenate the feature embedding along the channel dimension, before adding it to the timestep embedding in the UNet.

# IMPROVING Stable Diffusion

2. Micro–Conditioning

- Cause:  the use of random cropping during training
- Solution: Cropping parameters

**if** $h_{\text{original}} \leq w_{\text{original}}$ **then**
  $c_{\text{left}} \sim \mathcal{U}(0, \text{width}(x) - s_w)$ ▷ sample $c_{\text{left}}$
  $c_{\text{top}} = 0$
**else if** $h_{\text{original}} > w_{\text{original}}$ **then**
  $c_{\text{top}} \sim \mathcal{U}(0, \text{height}(x) - s_h)$ ▷ sample $c_{\text{top}}$
  $c_{\text{left}} = 0$
**end if**

| | *"A propaganda poster depicting a cat dressed as french emperor napoleon holding a piece of cheese."* | *"a close-up of a fire spitting dragon, cinematic shot."* |
|---|---|---|
| ***SD 1-5*** |  |  |
| ***SD 2-1*** |  |  |
| ***SDXL*** |  |  |

| $\mathbf{c}_{\text{crop}} = (0, 0)$ | $\mathbf{c}_{\text{crop}} = (0, 256)$, | $\mathbf{c}_{\text{crop}} = (256, 0)$, | $\mathbf{c}_{\text{crop}} = (512, 512)$, |
|---|---|---|---|
|  |  |  |  |

*'An astronaut riding a pig, highly realistic dslr photo, cinematic shot.'*

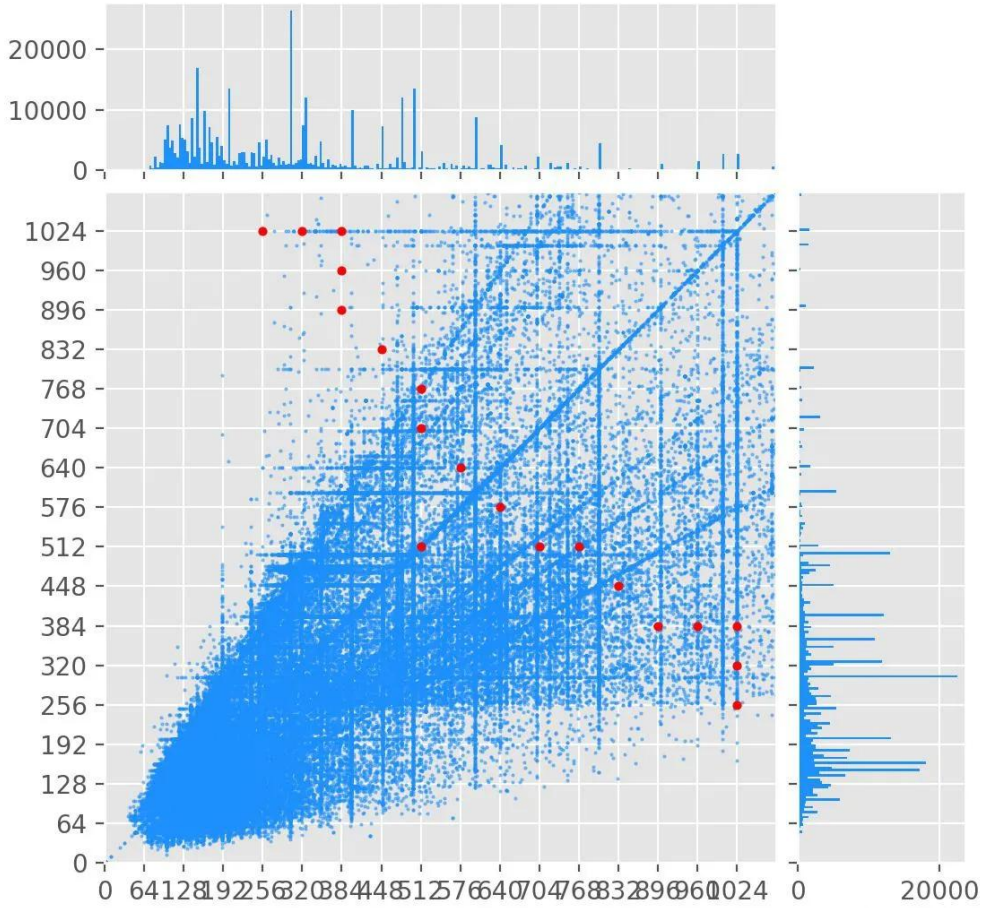
*'A capybara made of lego sitting in a realistic, natural field.'*

# IMPROVING Stable Diffusion

2. Micro–Conditioning

Another solution: **data bucketing**

- Partition the data into buckets of different aspect ratios
- Alternate between bucket sizes for each training step




https://github.com/NovelAI/novelai–aspect–ratio–bucketing

# IMPROVING Stable Diffusion

## 3. Multi–Aspect Training

- Finetune the model to handle multiple aspect–ratios
- Use data bucking
  - Receives the bucket size (or, target size) as a conditioning, represented as a tuple of integers $c_{ar} = (h_{tgt}, w_{tgt})$

Keep the pixel count ~ $1024^2$ pixels as possibly

| Height | Width | Aspect Ratio | Height | Width | Aspect Ratio |
|---|---|---|---|---|---|
| 512 | 2048 | 0.25 | 1024 | 1024 | 1.0 |
| 512 | 1984 | 0.26 | 1024 | 960 | 1.07 |
| 512 | 1920 | 0.27 | 1088 | 960 | 1.13 |
| 512 | 1856 | 0.28 | 1088 | 896 | 1.21 |
| 576 | 1792 | 0.32 | 1152 | 896 | 1.29 |
| 576 | 1728 | 0.33 | 1152 | 832 | 1.38 |
| 576 | 1664 | 0.35 | 1216 | 832 | 1.46 |
| 640 | 1600 | 0.4 | 1280 | 768 | 1.67 |
| 640 | 1536 | 0.42 | 1344 | 768 | 1.75 |
| 704 | 1472 | 0.48 | 1408 | 704 | 2.0 |
| 704 | 1408 | 0.5 | 1472 | 704 | 2.09 |
| 704 | 1344 | 0.52 | 1536 | 640 | 2.4 |
| 768 | 1344 | 0.57 | 1600 | 640 | 2.5 |
| 768 | 1280 | 0.6 | 1664 | 576 | 2.89 |
| 832 | 1216 | 0.68 | 1728 | 576 | 3.0 |
| 832 | 1152 | 0.72 | 1792 | 576 | 3.11 |
| 896 | 1152 | 0.78 | 1856 | 512 | 3.62 |
| 896 | 1088 | 0.82 | 1920 | 512 | 3.75 |
| 960 | 1088 | 0.88 | 1984 | 512 | 3.88 |
| 960 | 1024 | 0.94 | 2048 | 512 | 4.0 |

# IMPROVING Stable Diffusion

## 4. Improved Autoencoder

- Train the same AE in SD at a larger batch–size (256 vs 9), and track the weights with an exponential movingaverage (EMA)

Table 3: Autoencoder reconstruction performance on the COCO2017 [26] validation split, images of size $256 \times 256$ pixels. Note: *Stable Diffusion* 2.x uses an improved version of *Stable Diffusion* 1.x's autoencoder, where the decoder was finetuned with a reduced weight on the perceptual loss [55], and used more compute. Note that our new autoencoder is trained from scratch.

| model | PNSR ↑ | SSIM ↑ | LPIPS ↓ | rFID ↓ |
|---|---|---|---|---|
| *SDXL*-VAE | **24.7** | **0.73** | **0.88** | **4.4** |
| *SD*-VAE 1.x | 23.4 | 0.69 | 0.96 | 5.0 |
| *SD*-VAE 2.x | 24.5 | 0.71 | 0.92 | 4.7 |

- **SDXL–VAE is trained from scratch**

# IMPROVING Stable Diffusion

4. Improved Autoencoder

- In rebuttal: assess the effect of larger batch size and EMA
- Train the autoencoder (from scratch) on with (a) batch size = 8 and (b) batch size = 256. Models are not trained until convergence

| Model Name | Batch Size | EMA | Global Steps | rFID [↓] | PSNR [↑] | SSIM [↑] | LPIPS [↓] |
|---|---|---|---|---|---|---|---|
| B | 256 | ✗ | 500k | 7.52 | 24.30 | 0.71 | 1.27 |
| Be | 256 | ✓ | 500k | 7.35 | 24.33 | 0.72 | 1.24 |
| A1 | 8 | ✗ | 500k | 9.14 | 24.42 | 0.72 | 1.32 |
| A1e | 8 | ✓ | 500k | 8.85 | 24.51 | 0.72 | 1.29 |
| A2 | 8 | ✗ | 1.9M | 6.28 | 25.05 | 0.74 | 1.08 |
| A2e | 8 | ✓ | 1.9M | 5.57 | 25.05 | 0.74 | 1.07 |

# IMPROVING Stable Diffusion

5. Putting Everything Together

- First, we pretrain a base model on an internal dataset for 600k optimization steps at a resolution of 256 × 256 pixels and a batchsize of 2048.
- Then train 512 px for another 200k optimization steps.
- Finally use multi-aspect training in combination with an offset-noise level of 0.05 to train the model on different aspect ratios of ~ 1024 × 1024 pixel area.

https://www.crosslabs.org//blog/diffusion-with-offset-noise

# IMPROVING Stable Diffusion

5. Putting Everything Together

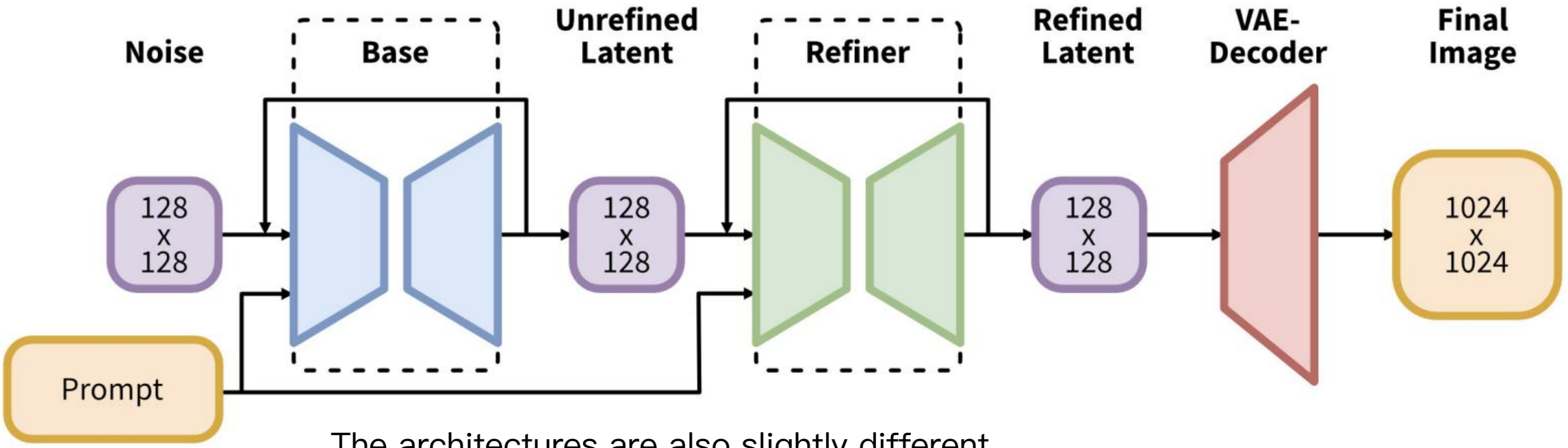
**Refinement Stage**

- Train a separate LDM in the same latent space, which is specialized on high–quality, high resolution data and employ a noising–denoising process
- Specialize it on the first 200 (discrete) noise scales.
- During inference, we render latents from the base SDXL, and directly diffuse and denoise them in latent space with the refinement.

# IMPROVING Stable Diffusion

## 5. Putting Everything Together

**Refinement Stage**



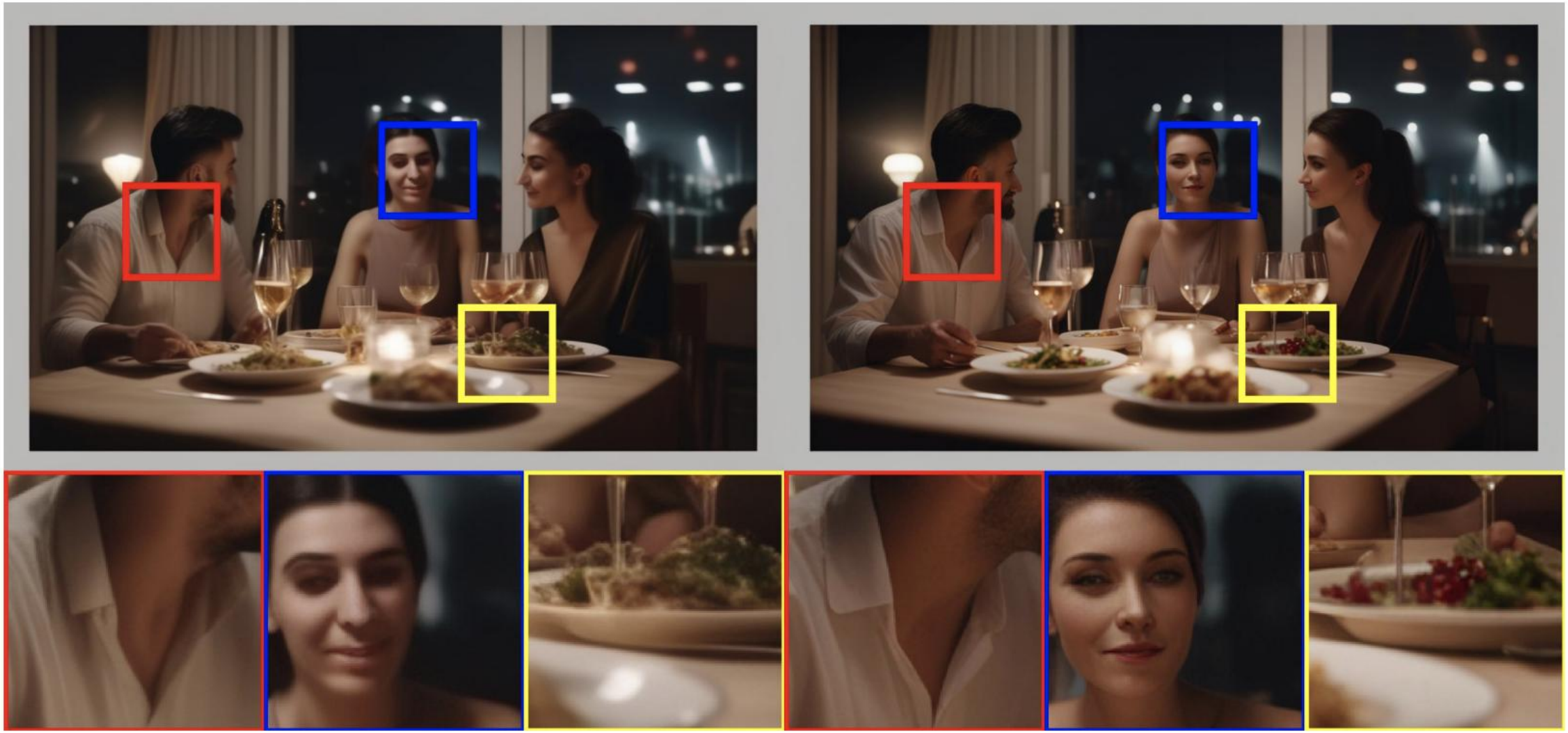

The architectures are also slightly different

w/o

w/

w/o

w/

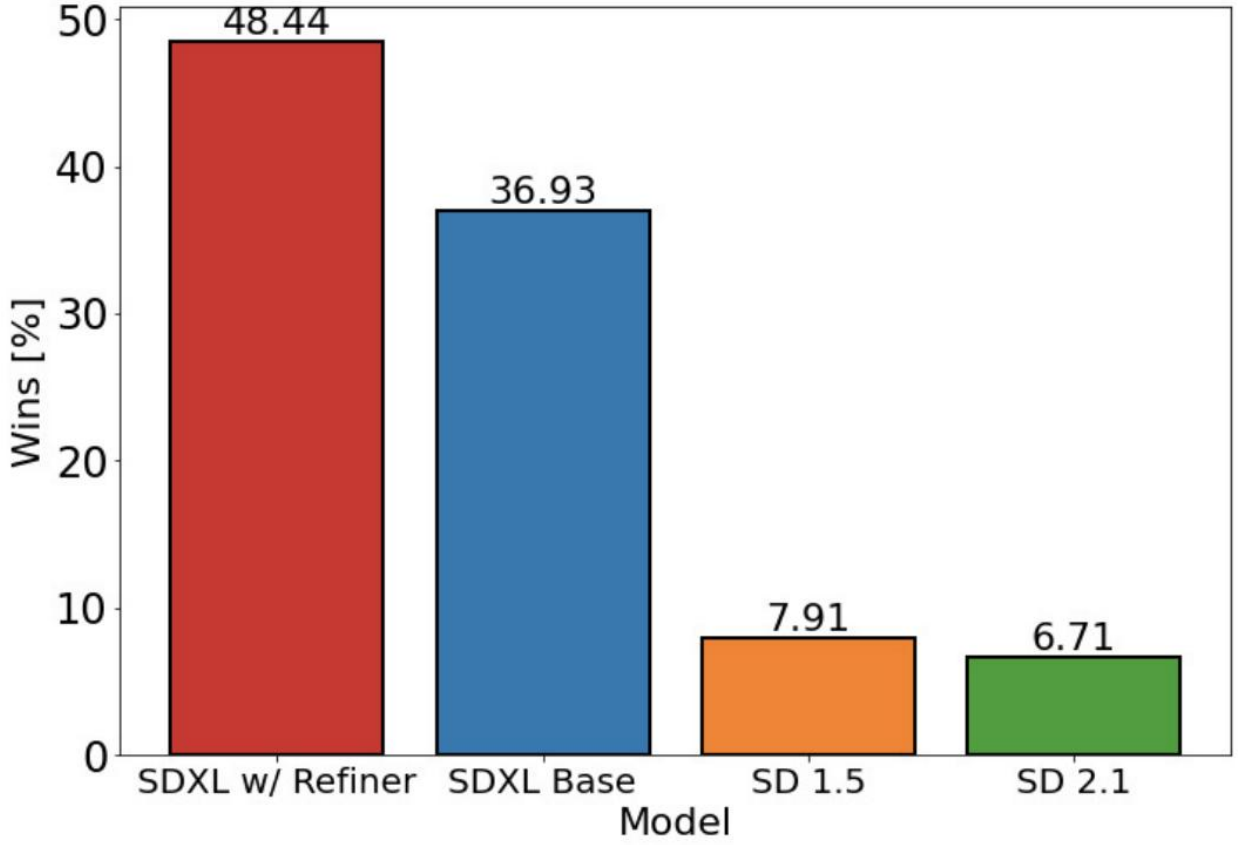


# IMPROVING Stable Diffusion

5. Putting Everything Together

**Refinement Stage**

- User study



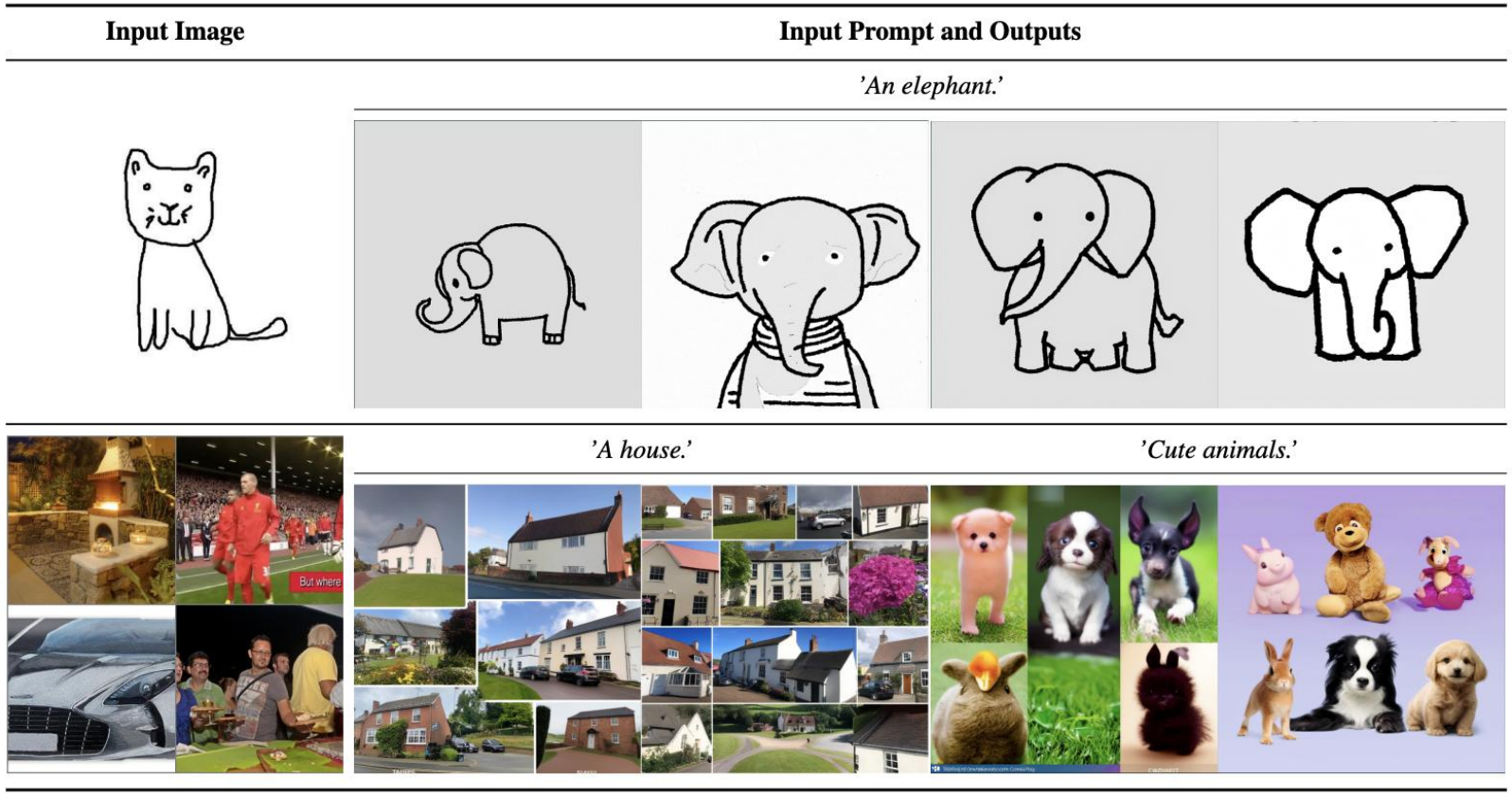
# IMPROVING Stable Diffusion

5. Multimodel Control

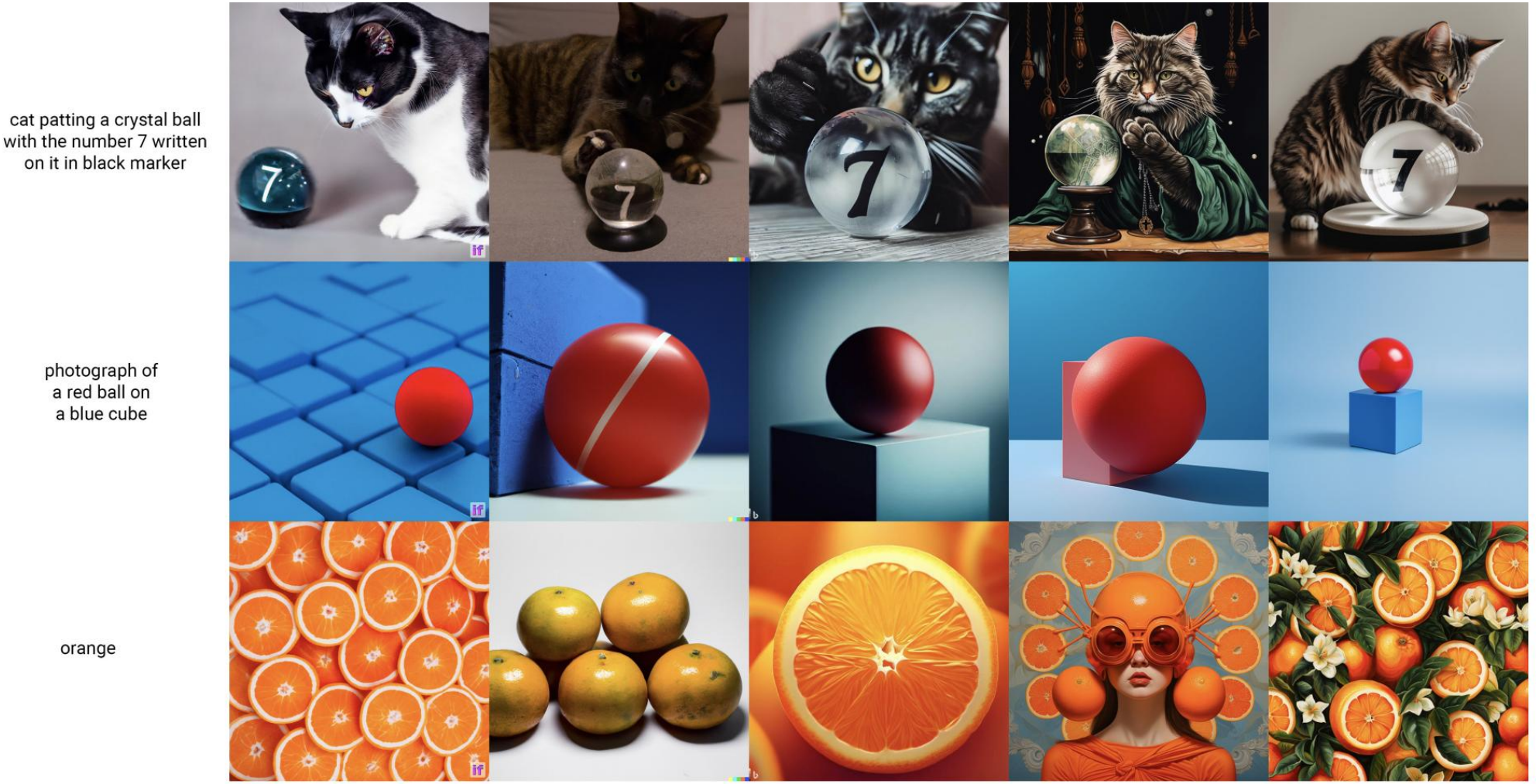
- Replacing the pooled text representations of CLIP which were used during training, with CLIP image features
- 1000 finetuning steps for the embedding layer that maps the CLIP embedding to the UNet's timestep embedding space (where they are added), and leave the remaining parameters frozen.

# IMPROVING Stable Diffusion

## 5. Multimodel Control



# Experiments



# Experiments


Figure 10: Results from 17,153 user preference comparisons between *SDXL* v0.9 and Midjourney v5.1, which was the latest version available at the time. The comparisons span all "categories" and "challenges" in the PartiPrompts (P2) benchmark. Notably, *SDXL* was favored 54.9% of the time over Midjourney V5.1. Preliminary testing indicates that the recently-released Midjourney V5.2 has lower prompt comprehension than its predecessor, but the laborious process of generating multiple prompts hampers the speed of conducting broader tests.

# Experiments

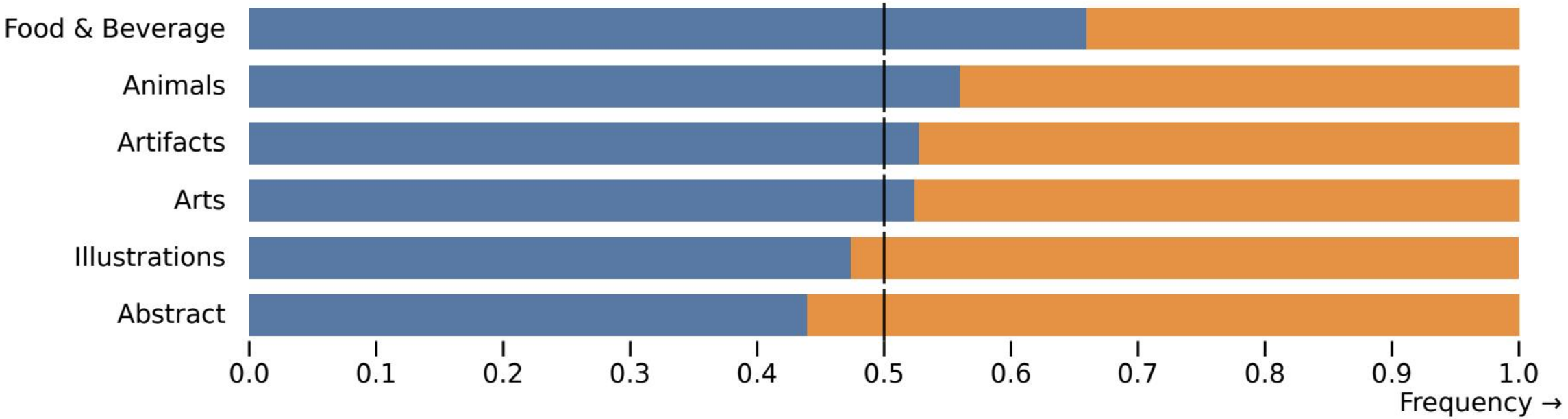



Figure 11: User preference comparison of *SDXL* (without refinement model) and Midjourney V5.1 across particular text categories. *SDXL* outperforms Midjourney V5.1 in all but two categories.

# Experiments

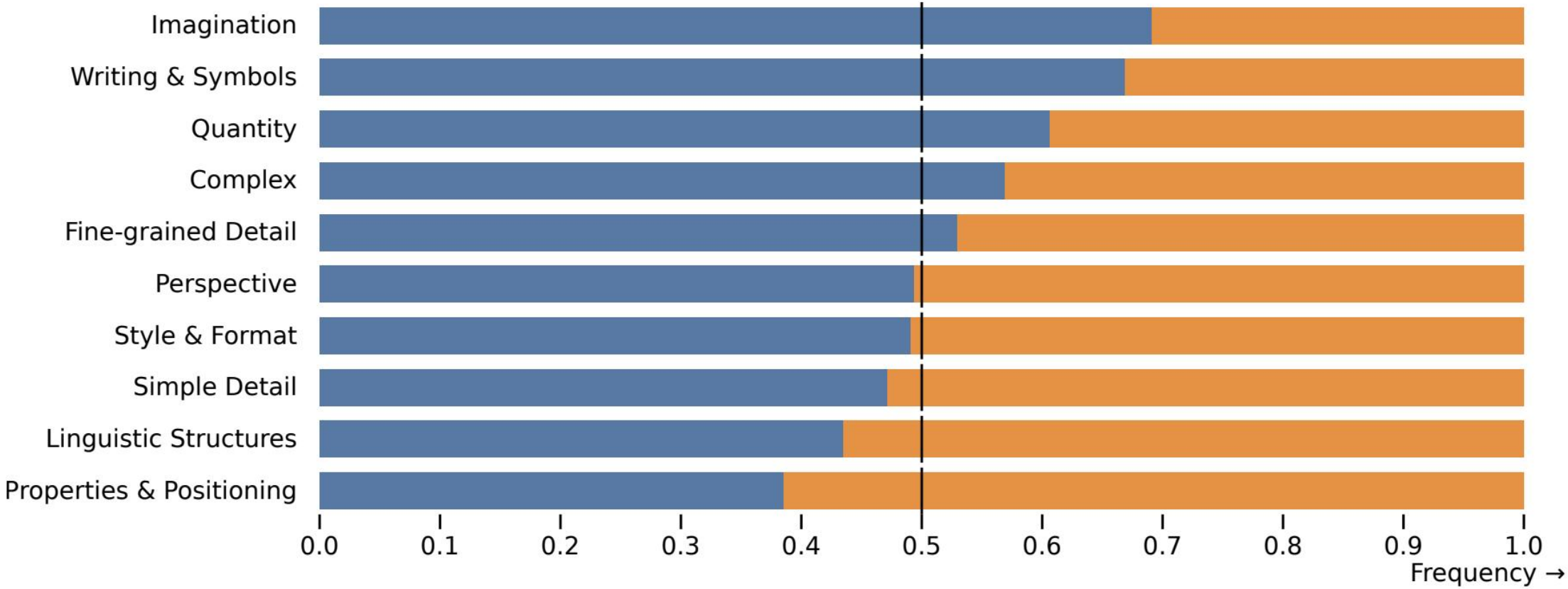
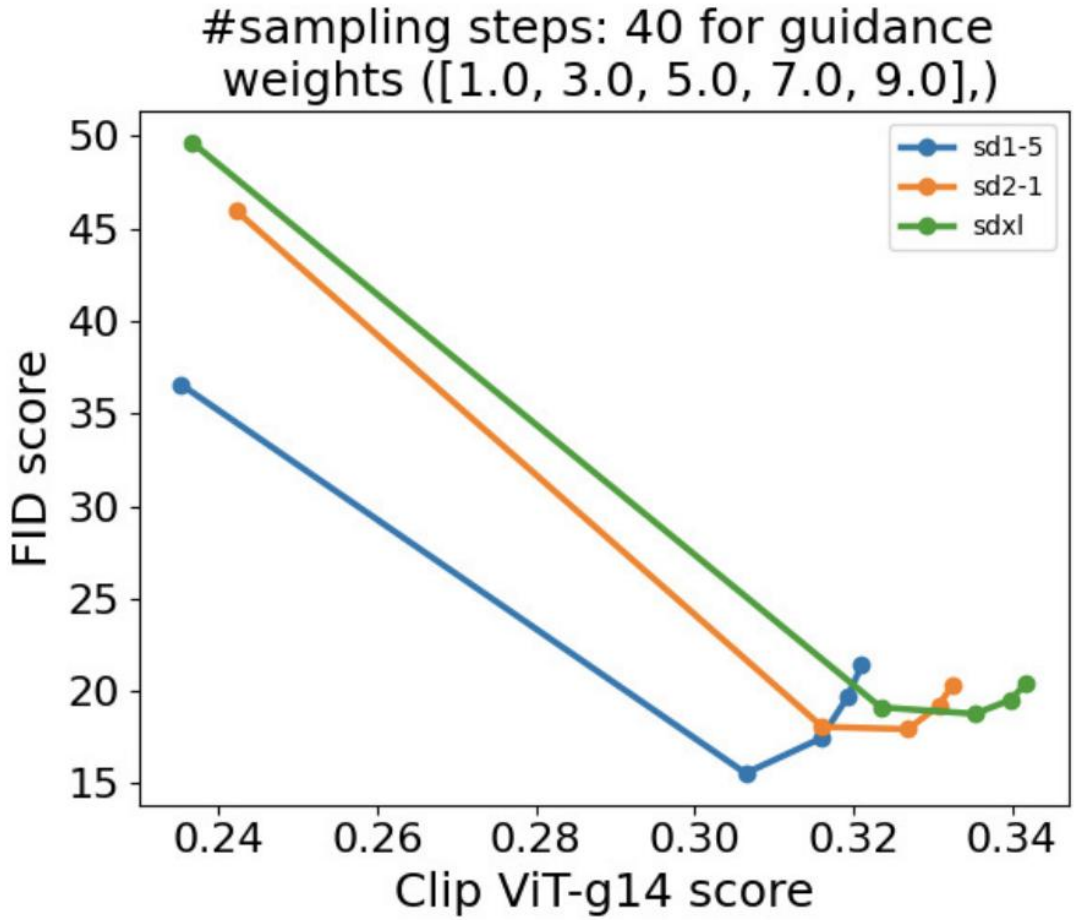
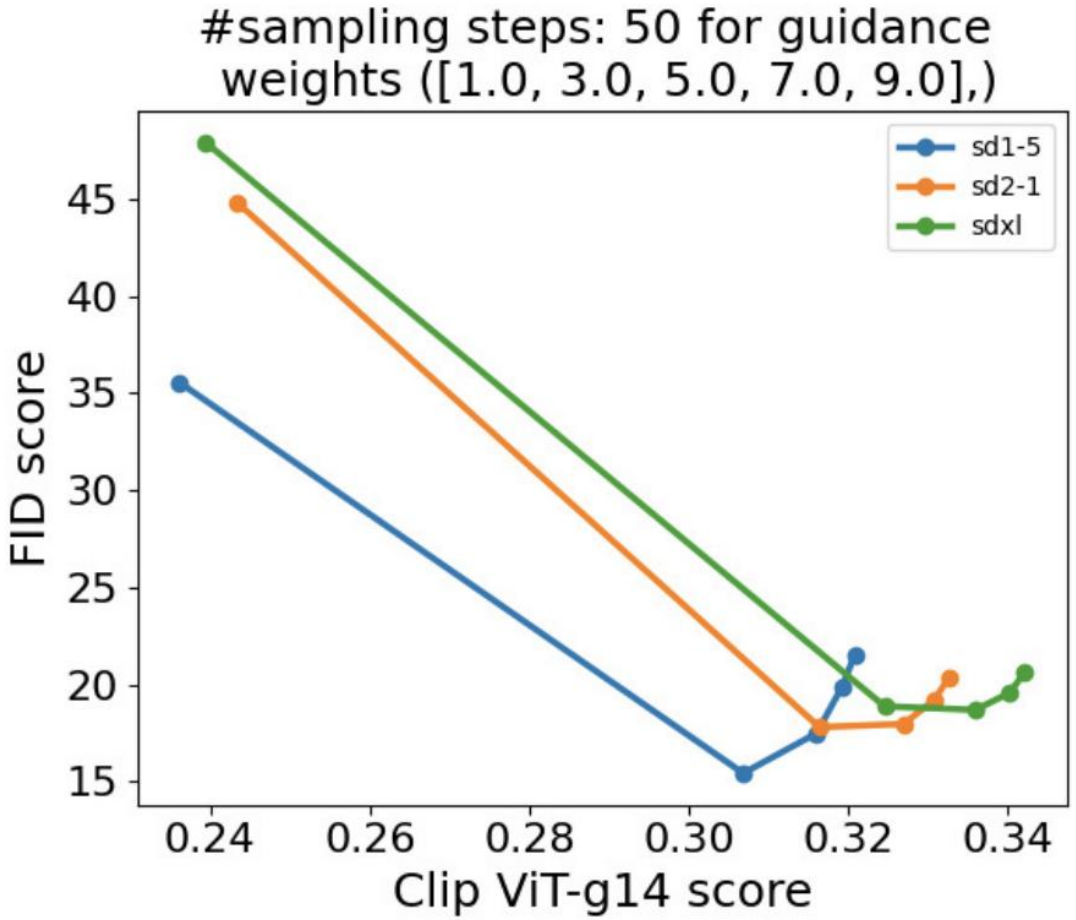



Figure 12: Preference comparisons of *SDXL* (with refinement model) to Midjourney V5.1 on complex prompts. *SDXL* either outperforms or is statistically equal to Midjourney V5.1 in 7 out of 10 categories.

# Experiments



# Conclusion

Improve SD

- The network complexity is increased
- Various refinement modules

**Justification For Why Not Higher Score**: While SDXL clearly demonstrates its compelling performance in text–to–image synthesis, the analysis of its modules and comparison to other methods are relatively lacking. Moreover, the overall pipeline remains similar to previous methods, although with cleverly designed modules. Therefore, the AC recommends a spotlight.

# Stable Video Diffusion

*"A robot dj is playing the turntables, in heavy raining futuristic tokyo, rooftop, sci-fi, fantasy"*

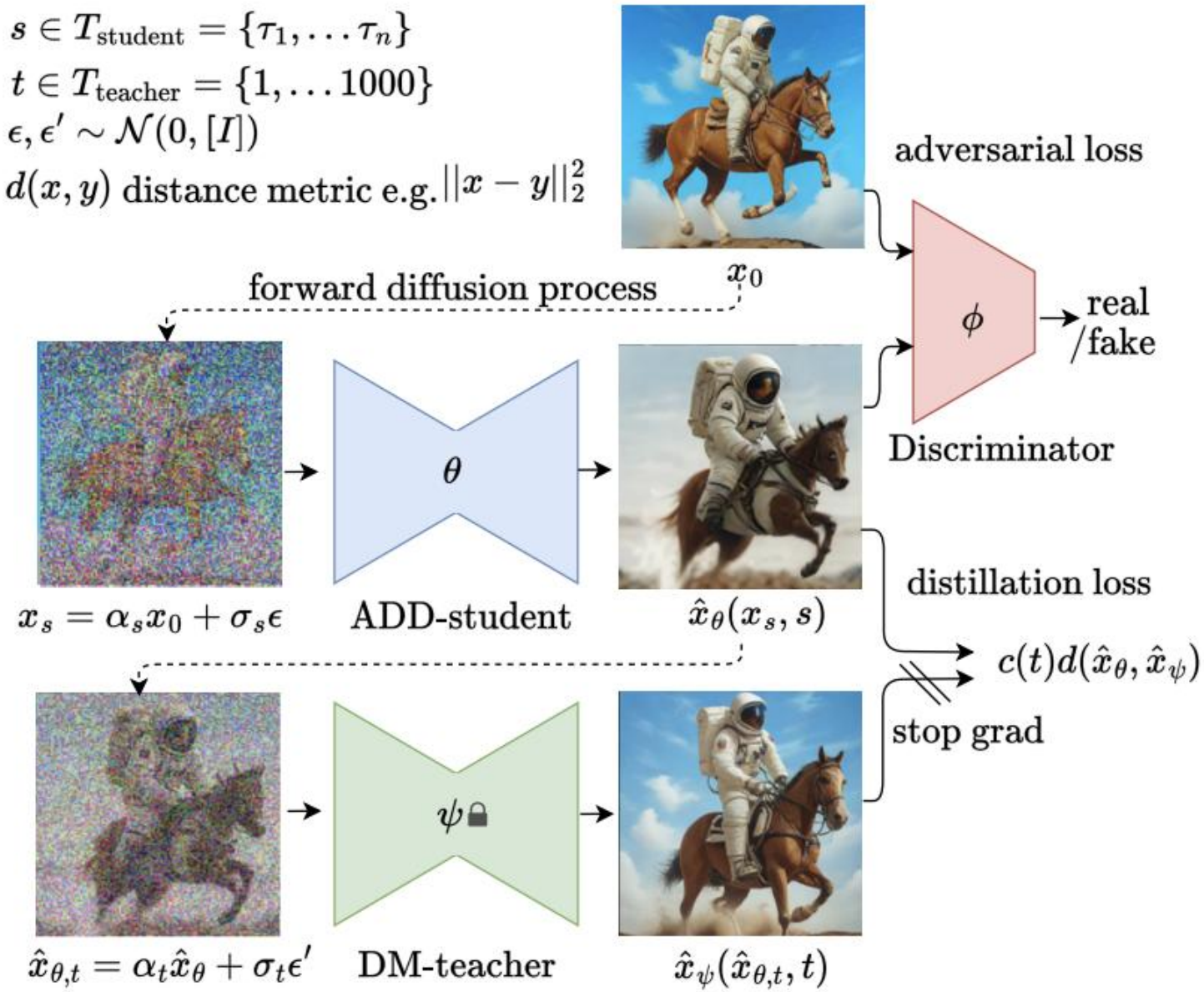
*"An exploding cheese house"*

*"A fat rabbit wearing a purple robe walking through a fantasy landscape"*

# SDXL Turbo

- Adversarial Diffusion Distillation

$$\mathcal{L} = \mathcal{L}^{\mathrm{G}}_{\mathrm{adv}}(\hat{x}_\theta(x_s, s), \phi) + \lambda \mathcal{L}_{\mathrm{distill}}(\hat{x}_\theta(x_s, s), \psi)$$



# SDXL Turb

- Adversarial

  Diffusion

  Distillation

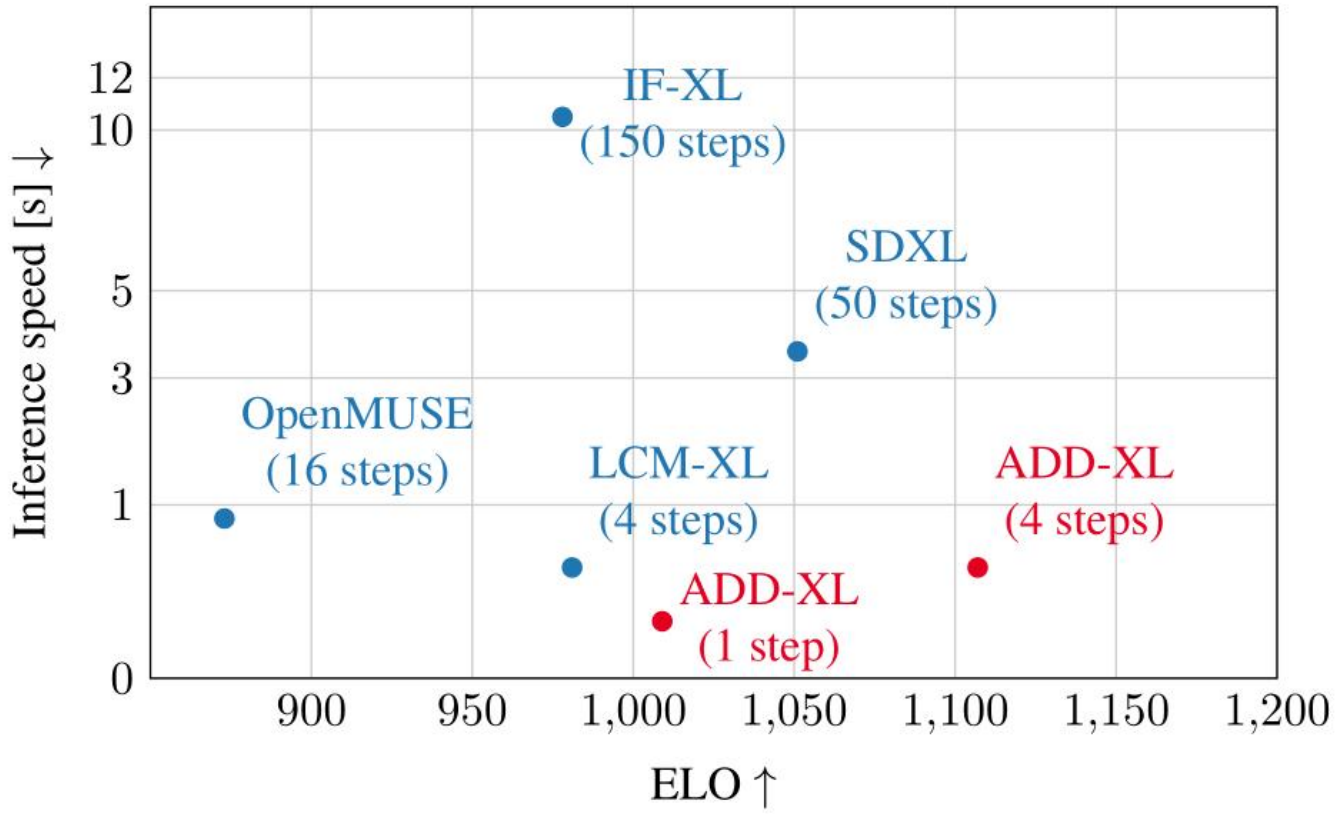
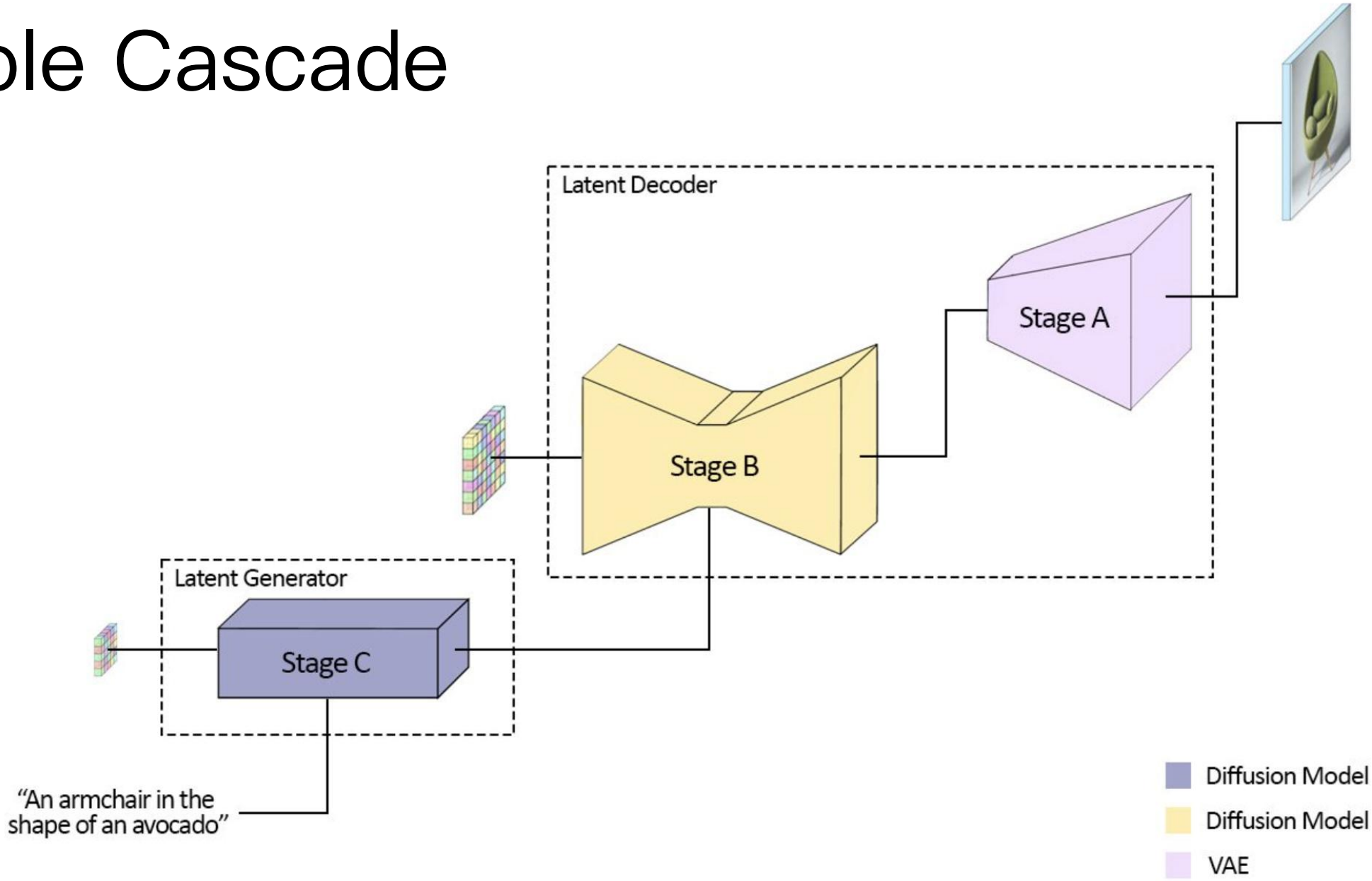
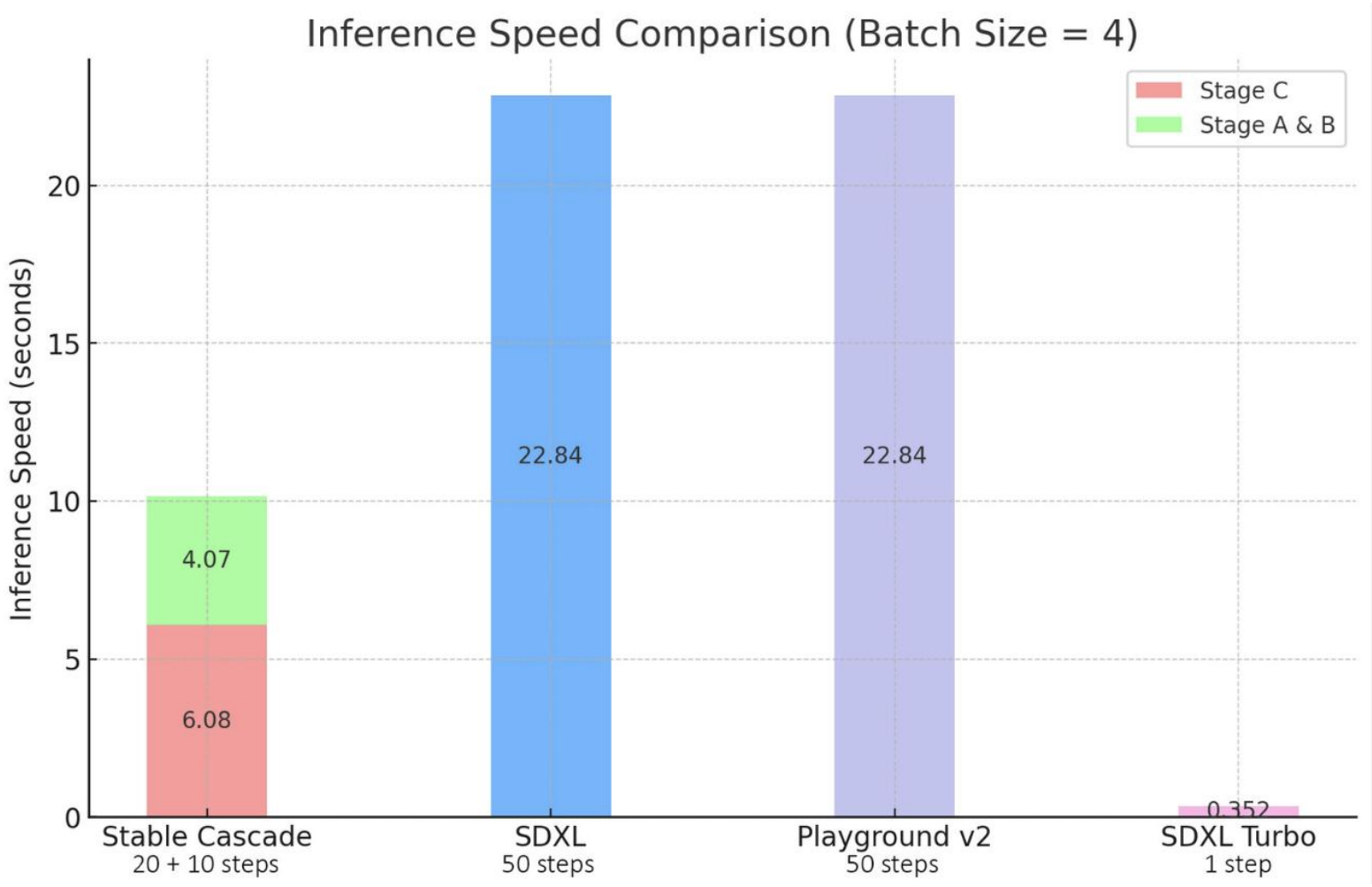



Figure 7. **Performance vs. speed.** We visualize the results reported in Fig. 6 in combination with the inference speeds of the respective models. The speeds are calculated for generating a single sample at resolution 512x512 on an A100 in mixed precision.

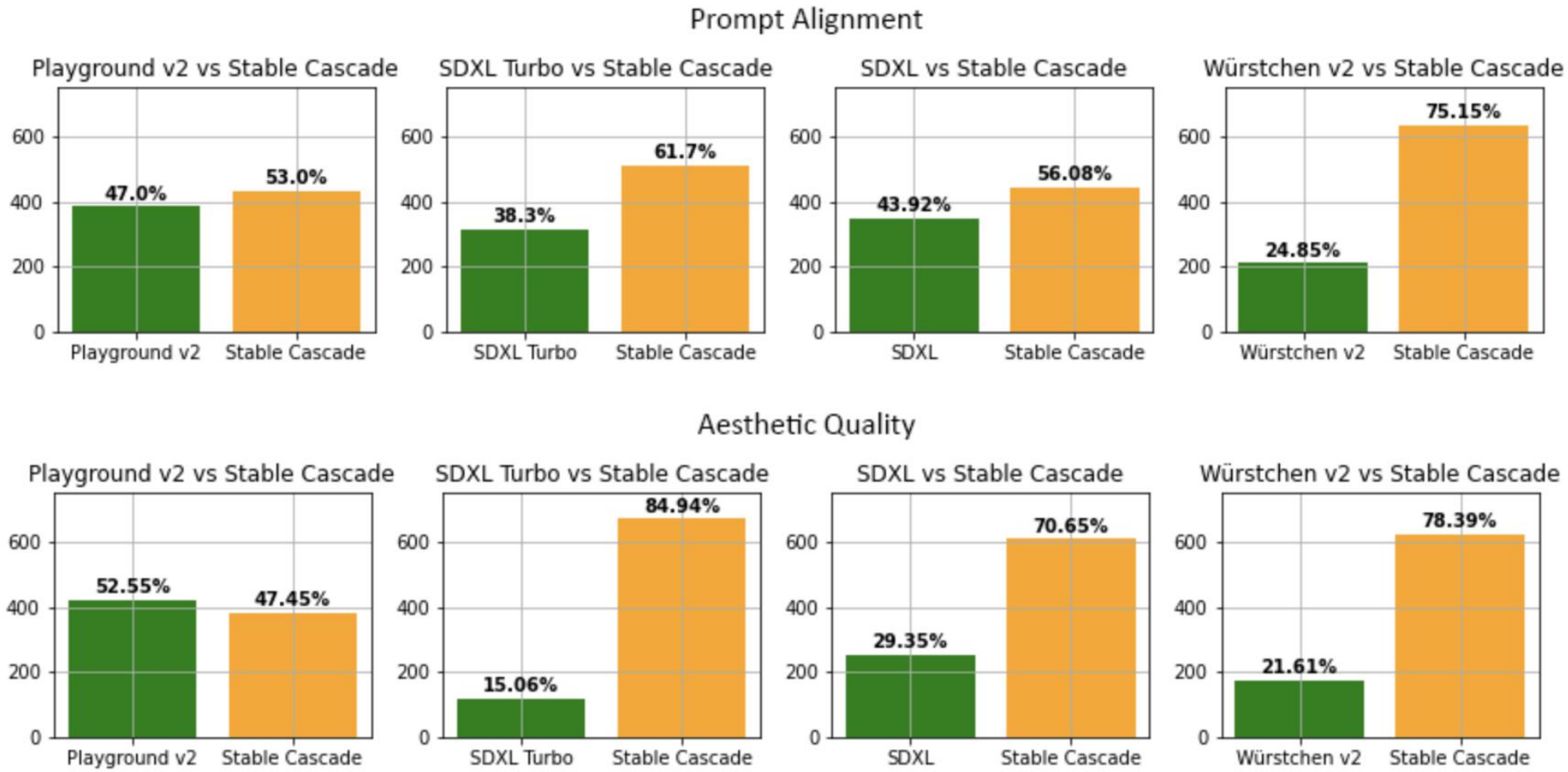
# Stable Cascade



# Stable Cascade



# Stable Cascade



# Stable Cascade

- WURSTCHEN (ICLR 2024)

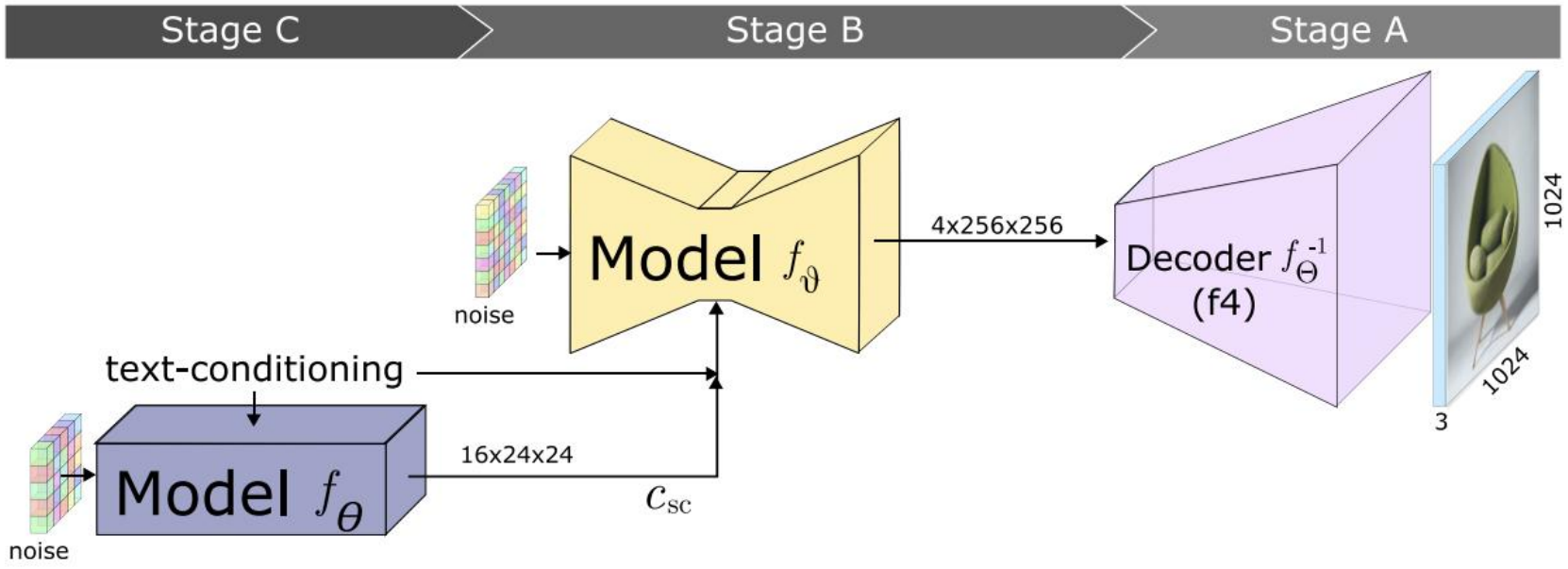



Figure 2: Inference architecture for text-conditional image generation.

# Stable Diffusion 3

- Diffusion transformer
- Flow matching
- Models currently ranges from 800M to 8B parameters