Diffusion models

Jian Li
IIIS, Tsinghua

ATCS 2023

Review: score-based generative diffusion models

Review: reverse time SDE

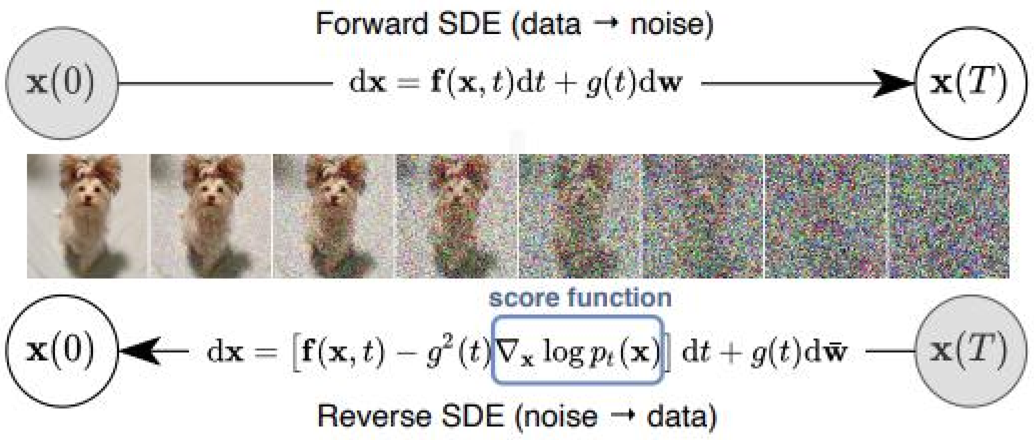

Forward SDE (data → noise)
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$
$\mathbf{x}(T)$
score function
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}]\, \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$
$\mathbf{x}(T)$
Reverse SDE (noise → data)

Figure 1: **Solving a reverse-time SDE yields a score-based generative model.** Transforming data to a simple noise distribution can be accomplished with a continuous-time SDE. This SDE can be reversed if we know the score of the distribution at each intermediate time step, $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$.

DENOISING DIFFUSION PROBABILISTIC MODELS (DDPM)

$$p(\mathbf{x}_i \mid \mathbf{x}_{i-1}) = \mathcal{N}(\mathbf{x}_i; \sqrt{1-\beta_i}\mathbf{x}_{i-1}, \beta_i \mathbf{I})$$

$$\mathbf{x}_i = \sqrt{1-\beta_i}\mathbf{x}_{i-1} + \sqrt{\beta_i}\mathbf{z}_{i-1}$$

which is a discretization of OU process: $\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\mathbf{w}.$

$$p_{\alpha_i}(\mathbf{x}_i \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_i; \sqrt{\alpha_i}\mathbf{x}_0, (1-\alpha_i)\mathbf{I}), \text{ where } \alpha_i := \prod_{j=1}^{i}(1-\beta_j)$$

Score matching loss:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} (1-\alpha_i)\mathbb{E}_{p_{\mathrm{data}}(\mathbf{x})}\mathbb{E}_{p_{\alpha_i}(\tilde{\mathbf{x}}|\mathbf{x})}[\|\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, i) - \nabla_{\tilde{\mathbf{x}}} \log p_{\alpha_i}(\tilde{\mathbf{x}} \mid \mathbf{x})\|_2^2].$$

The original DDPM paper uses the variational lower bound to derive the loss function (see later)

Probability flow ODE

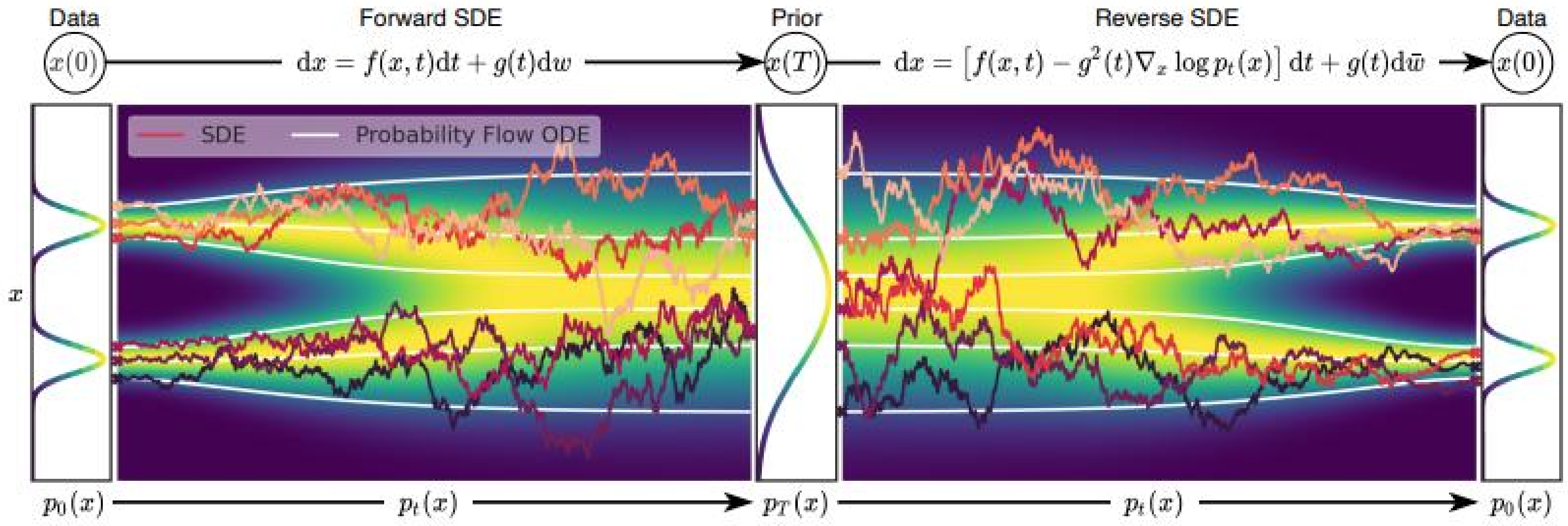

Data
Forward SDE
Prior
Reverse SDE
Data
$x(0)$
$\mathrm{d}x = f(x,t)\mathrm{d}t + g(t)\mathrm{d}w$
$x(T)$
$\mathrm{d}x = [f(x,t) - g^2(t)\nabla_x \log p_t(x)]\,\mathrm{d}t + g(t)\mathrm{d}\bar{w}$
$x(0)$
SDE
Probability Flow ODE
x
$p_0(x)$
$p_t(x)$
$p_T(x)$
$p_t(x)$
$p_0(x)$

Figure 2: **Overview of score-based generative modeling through SDEs**. We can map data to a noise distribution (the prior) with an SDE (Section 3.1), and reverse this SDE for generative modeling (Section 3.2). We can also reverse the associated probability flow ODE (Section 4.3), which yields a deterministic process that samples from the same distribution as the SDE. Both the reverse-time SDE and probability flow ODE can be obtained by estimating the score $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ (Section 3.3).

prob flow ODE: $\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t$

A Variational Perspective of Diffusion models

Variational Diffusion Models

Denoising Diffusion Probabilistic Models

Quick intro to ELBO

A quick intro to variational inference

- Typically, the posterior is hard to compute and sample from (MCMC approach can be pretty slow)
- We wish to use q (from some parametric family) to approximate the posterior p(z|x)

$$q^*(\mathbf{z}) = \underset{q(\mathbf{z}) \in \mathscr{Q}}{\arg\min}\, \mathrm{KL}\left(q(\mathbf{z}) \| p(\mathbf{z} \mid \mathbf{x})\right).$$

$$\mathrm{KL}\left(q(\mathbf{z}) \| p(\mathbf{z} \mid \mathbf{x})\right) = \mathbb{E}\left[\log q(\mathbf{z})\right] - \mathbb{E}\left[\log p(\mathbf{z} \mid \mathbf{x})\right]$$

$$\mathrm{KL}\left(q(\mathbf{z}) \| p(\mathbf{z} \mid \mathbf{x})\right) = \mathbb{E}\left[\log q(\mathbf{z})\right] - \mathbb{E}\left[\log p(\mathbf{z}, \mathbf{x})\right] + \log p(\mathbf{x}).$$

Minimizing KL is equivalent to maximizing ELBO (since evidence logp(x) doesn't depend on z)

ELBO: evidence lower bound
ELBO<=logp(x)

$$\mathrm{ELBO}(q) = \mathbb{E}\left[\log p(\mathbf{z}, \mathbf{x})\right] - \mathbb{E}\left[\log q(\mathbf{z})\right]$$

$$\begin{aligned}\mathrm{ELBO}(q) &= \mathbb{E}\left[\log p(\mathbf{z})\right] + \mathbb{E}\left[\log p(\mathbf{x} \mid \mathbf{z})\right] - \mathbb{E}\left[\log q(\mathbf{z})\right] \\ &= \mathbb{E}\left[\log p(\mathbf{x} \mid \mathbf{z})\right] - \mathrm{KL}\left(q(\mathbf{z}) \| p(\mathbf{z})\right).\end{aligned}$$

Variational Diffusion models

x: data
z: z_0 *to* z_1 diffusion latent variables (progressively adding noise)
q: forward process, adding noise $q(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}\left(\alpha_t \mathbf{x}, \sigma_t^2 \mathbf{I}\right)$
p: reverse process

Maximize likelihood of the data: use EBLO loss for data x

$$-\log p(\mathbf{x}) \leq -\mathrm{VLB}(\mathbf{x}) = \underbrace{D_{KL}(q(\mathbf{z}_1|\mathbf{x})||p(\mathbf{z}_1))}_{\text{Prior loss}} + \underbrace{\mathbb{E}_{q(\mathbf{z}_0|\mathbf{x})}\left[-\log p(\mathbf{x}|\mathbf{z}_0)\right]}_{\text{Reconstruction loss}} + \underbrace{\mathcal{L}_T(\mathbf{x})}_{\text{Diffusion loss}}.$$

In the case of finite T, using $s(i) = (i-1)/T$, $t(i) = i/T$, the diffusion loss is:

$$\mathcal{L}_T(\mathbf{x}) = \sum_{i=1}^{T} \mathbb{E}_{q(\mathbf{z}_{t(i)}|\mathbf{x})} D_{KL}[q(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)}, \mathbf{x})||p(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)})].$$ chain rule of KL

By specifying p and q, we can get a loss that mirror the score-matching loss.

$\mathrm{SNR}(t) = \alpha_t^2/\sigma_t^2.$ $$\mathcal{L}_T(\mathbf{x}) = \frac{T}{2}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,\mathbf{I}), i\sim U\{1,T\}}\left[(\mathrm{SNR}(s) - \mathrm{SNR}(t))\,||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)||_2^2\right]$$

$$\mathcal{L}_T(\mathbf{x}) = \frac{T}{2}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,\mathbf{I}), i\sim U\{1,T\}}\left[\left(\exp(\gamma_{\boldsymbol{\eta}}(t) - \gamma_{\boldsymbol{\eta}}(s)) - 1\right)\|\boldsymbol{\epsilon} - \underline{\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)}\|_2^2\right]$$

Noise prediction model

VQ-GAN

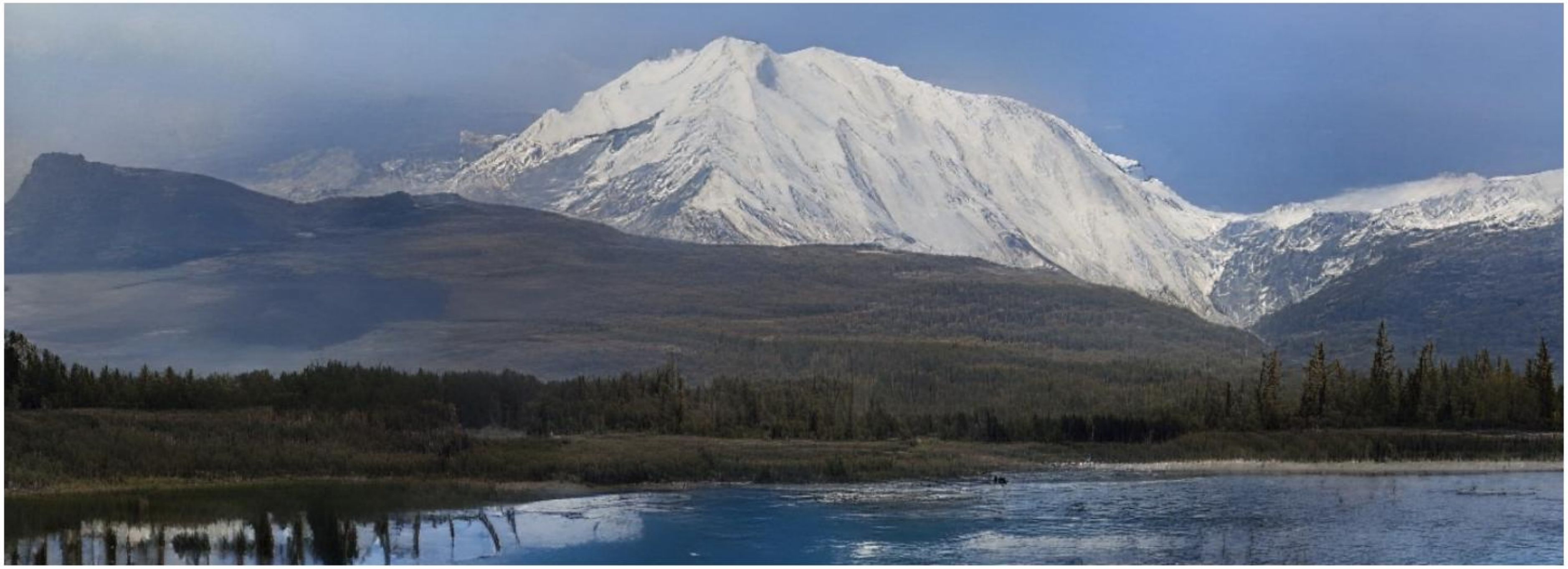


Figure 1. Our approach enables transformers to synthesize high-resolution images like this one, which contains 1280x460 pixels.

Taming transformers for high-resolution image synthesis.

VQ-GAN

- Taken together, convolutional and transformer architectures can model the compositional nature of our visual world
  - CNN: to efficiently learn a codebook of context-rich visual parts
  - Transformer: long-range interactions for the compositions of patches

Learnable codebook Z

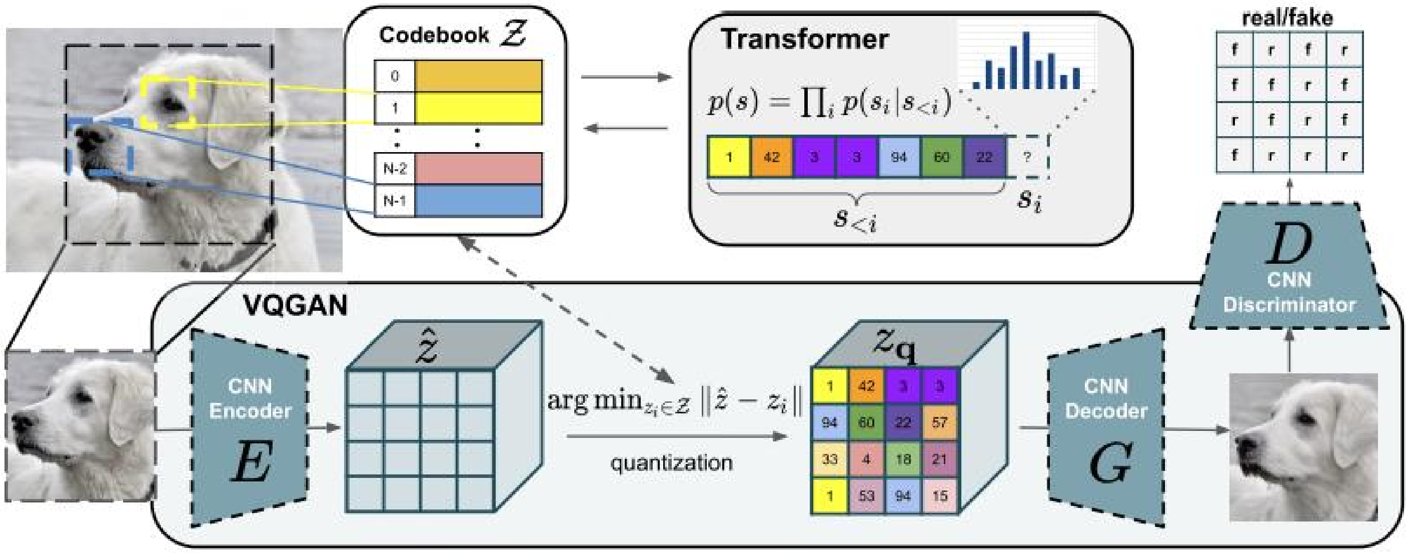


Figure 2. Our approach uses a convolutional *VQGAN* to learn a codebook of context-rich visual parts, whose composition is subsequently modeled with an autoregressive transformer architecture. A discrete codebook provides the interface between these architectures and a patch-based discriminator enables strong compression while retaining high perceptual quality. This method introduces the efficiency of convolutional approaches to transformer based high resolution image synthesis.

Details in [1] Taming transformers for high-resolution image synthesis.

VQ-GAN

Learnable codebook Z

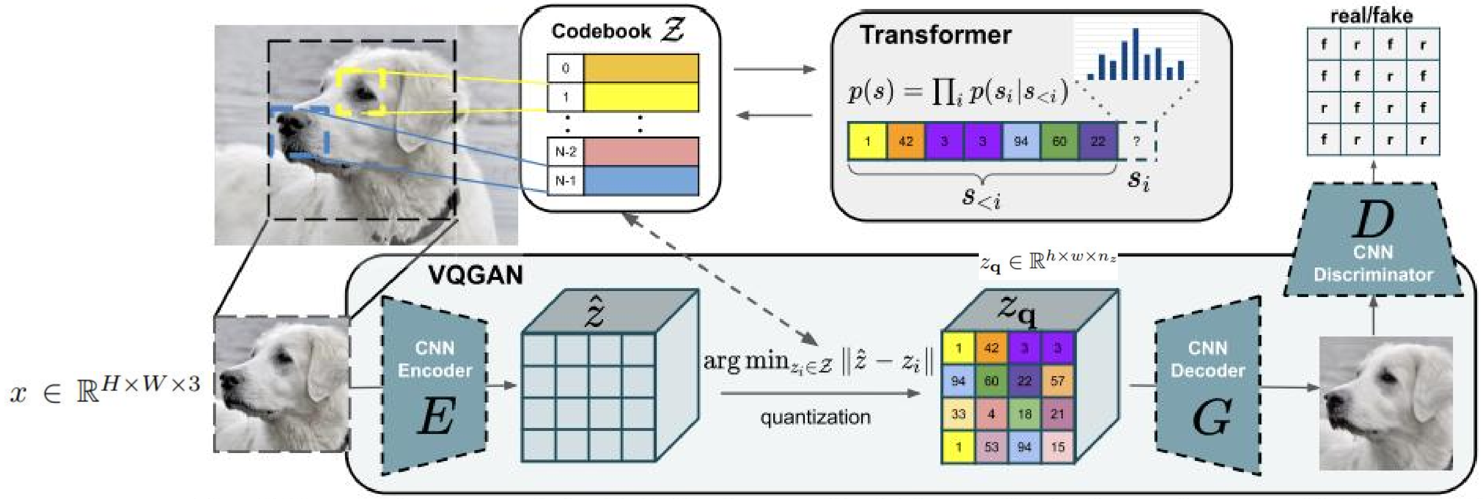


Figure 2. Our approach uses a convolutional *VQGAN* to learn a codebook of context-rich visual parts, whose composition is subsequently modeled with an autoregressive transformer architecture. A discrete codebook provides the interface between these architectures and a patch-based discriminator enables strong compression while retaining high perceptual quality. This method introduces the efficiency of convolutional approaches to transformer based high resolution image synthesis.

Z: Learnable discrete codebook $|\mathcal{Z}| = 1024$

$$z_{\mathbf{q}} = \mathbf{q}(\hat{z}) := \left(\underset{z_k\in\mathcal{Z}}{\arg\min}\|\hat{z}_{ij} - z_k\|\right) \in \mathbb{R}^{h\times w\times n_z}.$$

Final objective of VQGAN

$$\mathcal{Q}^* = \underset{E,G,\mathcal{Z}}{\arg\min}\,\max_{D}\,\mathbb{E}_{x\sim p(x)}\Big[\mathcal{L}_{\mathrm{VQ}}(E,G,\mathcal{Z}) + \lambda\mathcal{L}_{\mathrm{GAN}}(\{E,G,\mathcal{Z}\},D)\Big]$$

Backpropagation through the non-differentiable quantization operation is achieved by a straight-through gradient estimator, which simply copies the gradients from the decoder to the encoder

$$\mathcal{L}_{\mathrm{VQ}}(E,G,\mathcal{Z}) = \|x-\hat{x}\|^2 + \|\mathrm{sg}[E(x)] - z_{\mathbf{q}}\|_2^2 + \|\mathrm{sg}[z_{\mathbf{q}}] - E(x)\|_2^2.$$

Sg: stop gradient

$$\mathcal{L}_{\mathrm{GAN}}(\{E,G,\mathcal{Z}\},D) = [\log D(x) + \log(1-D(\hat{x}))]$$

patch-based adversarial GAN loss

VQ-GAN

Generate images using VQGAN

- Using transformer to learning the sequence of codebooks (attention is all you need in latent space)
  - Generate the sequence autoregressively

$$p(s) = \prod_i p(s_i|s_{<i})$$

  - Adding the condition

$$p(s|c) = \prod_i p(s_i|s_{<i}, c).$$

  - Generation:

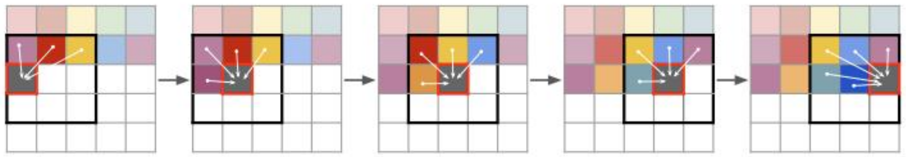


Figure 3. Sliding attention window.

Stable Diffusion

'A painting of the last supper by Picasso.'

High-Resolution Image Synthesis with Latent Diffusion Models

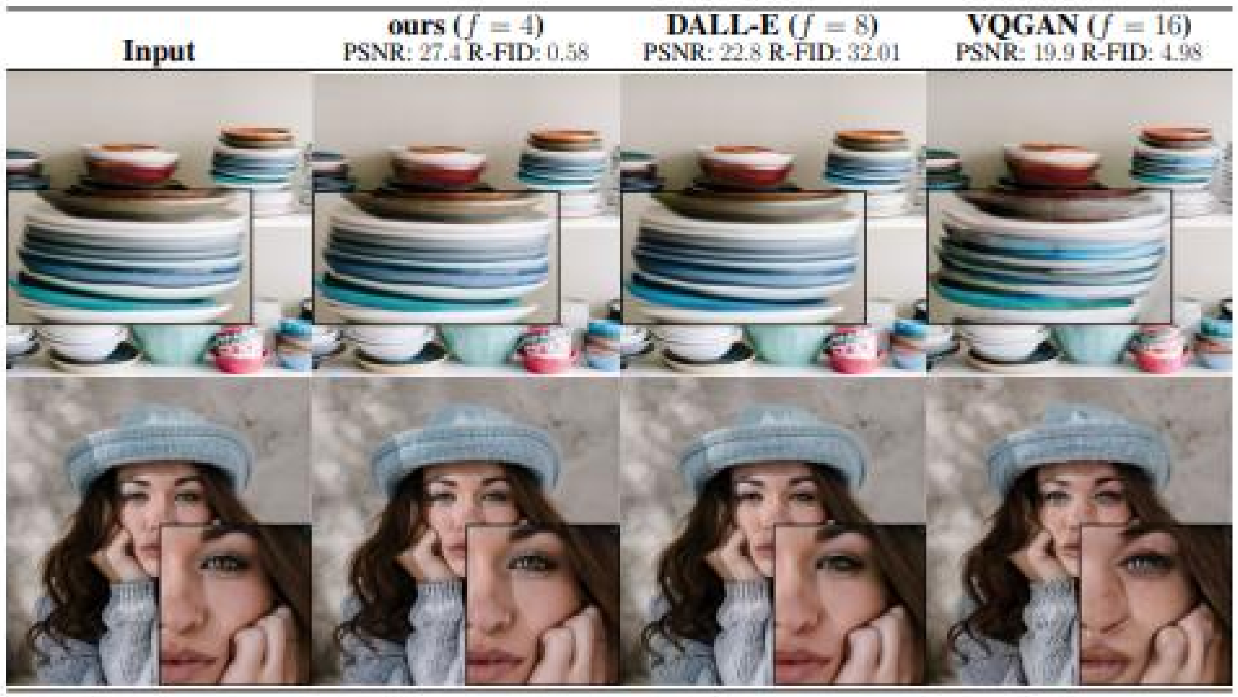


Figure 1. Boosting the upper bound on achievable quality with less agressive downsampling. Since diffusion models offer excellent inductive biases for spatial data, we do not need the heavy spatial downsampling of related generative models in latent space, but can still greatly reduce the dimensionality of the data via suitable autoencoding models, see Sec. 3. Images are from the DIV2K [1] validation set, evaluated at 512^2 px. We denote the spatial downsampling factor by f. Reconstruction FIDs [28] and PSNR are calculated on ImageNet-val. [12]; see also Tab. 8.

Stable diffusion

Idea: the reconstructions are confined to the image manifold (enforcing local realism), rather than on pixel-space **(Diffusion in latent space)**

Use an autoencoder in VQGAN (trained by combination of a perceptual loss and a patch-based adversarial objective)

- Encoder E encodes image x into a latent representation z = E(x)
  - The encoder downsamples the image by a factor f
- Decoder D reconstructs the image from the latent, giving x˜ = D(z) = D(E(x))

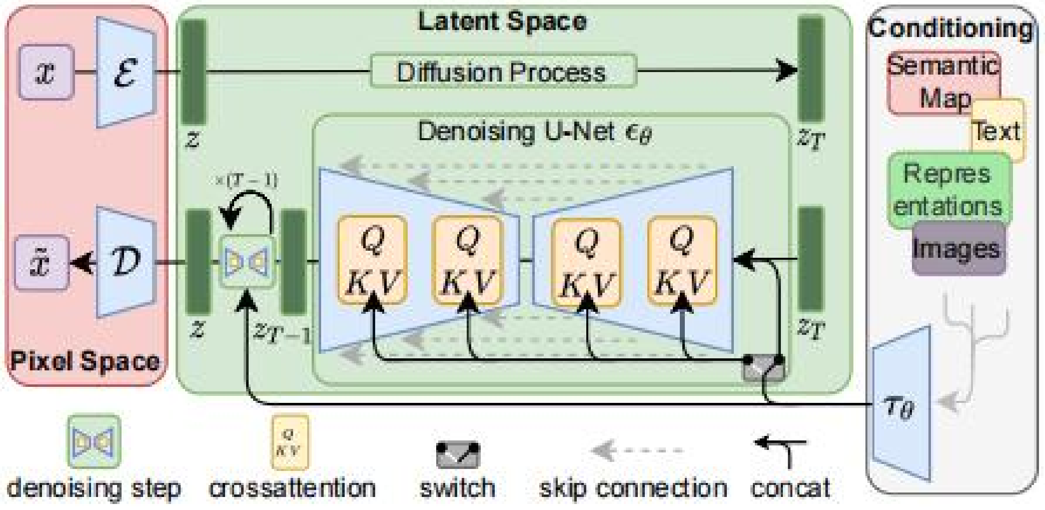


Figure 3. We condition LDMs either via concatenation or by a more general cross-attention mechanism. See Sec. 3.3

Stable diffusion

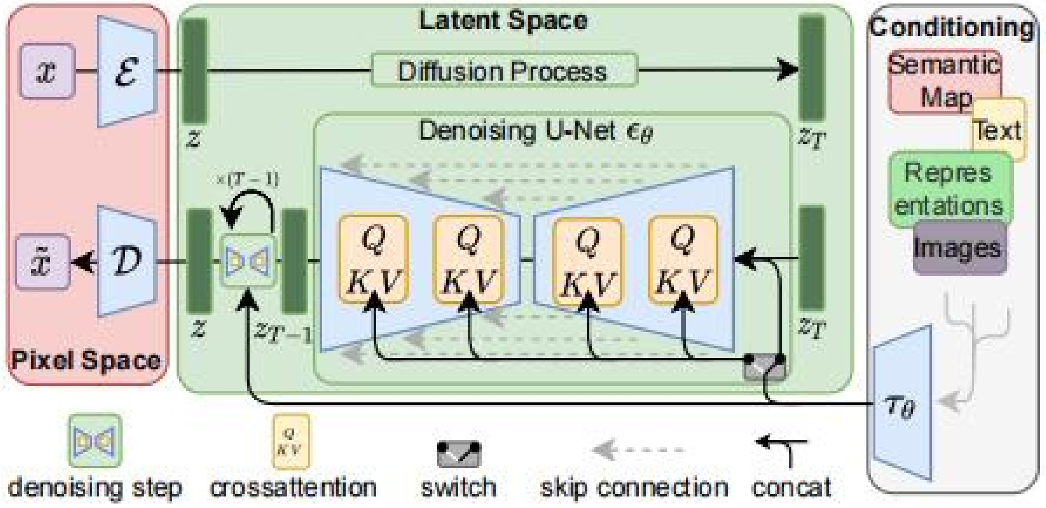


Figure 3. We condition LDMs either via concatenation or by a more general cross-attention mechanism. See Sec. 3.3

Loss from variational lower bound on p(x), which mirrors denoising score-matching

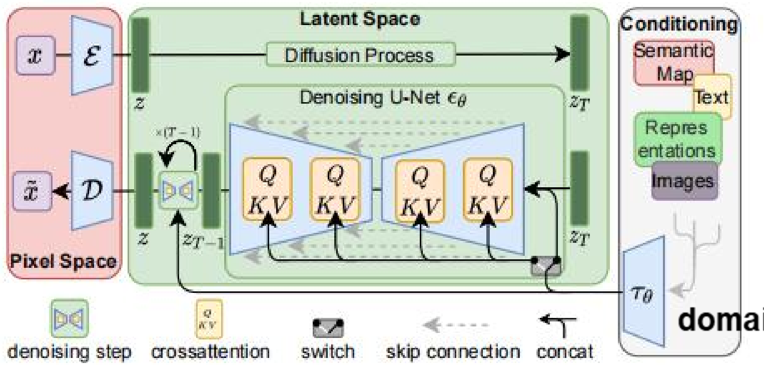
Loss of Diffusion model: $L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\Big[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\Big]$

denoising autoencoders $\epsilon_\theta(x_t, t)$

Loss of Diffusion model in latent space: $L_{LDM} := \mathbb{E}_{\mathcal{E}(x),\epsilon\sim\mathcal{N}(0,1),t}\Big[\|\epsilon - \epsilon_\theta(z_t, t)\|_2^2\Big]$.

The neural backbone of the model $\epsilon_\theta(z_t, t)$ is realized as a time-conditional UNet

Stable diffusion



inputs y: text, semantic maps or other image-to-image translation tasks

domain specific encoder τ_θ

Figure 3. We condition LDMs either via concatenation or by a more general cross-attention mechanism. See Sec. 3.3

Conditioning Mechanisms:

a conditional denoising autoencoder $\epsilon_\theta(z_t, t, y)$

the cross-attention mechanism:

- First pre-process y from various modalities (such as language prompts) use a domain specific encoder τ_θ that projects y to an intermediate representation
- Cross attention: $\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \cdot V$, with

$$Q = W_Q^{(i)} \cdot \underbrace{\varphi_i(z_t)}_{\text{a (flattened) intermediate representation of the UNet implementing } \epsilon_\theta},\ K = W_K^{(i)} \cdot \tau_\theta(y),\ V = W_V^{(i)} \cdot \tau_\theta(y).$$

- Loss of Conditional LDM: $L_{LDM} := \mathbb{E}_{\mathcal{E}(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2\right]$

Such a attention mechanism is complicated and may interfere the UNet generation See **ControlNet**

Some details about UNet

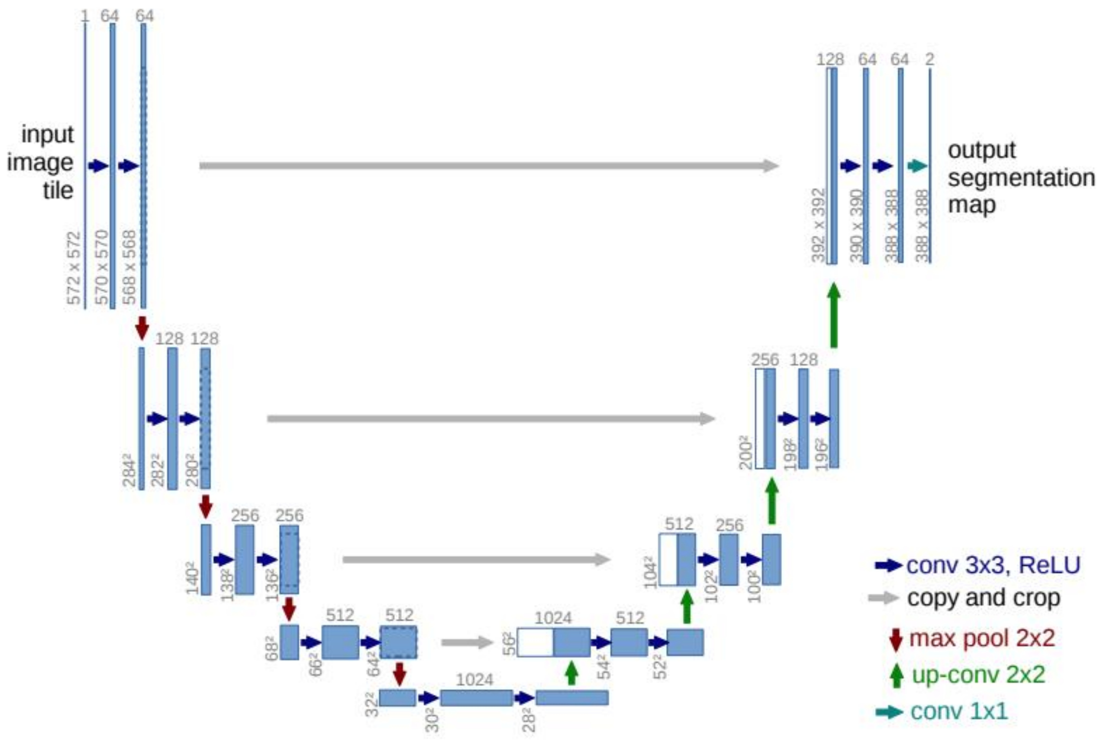


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

See: U-Net: Convolutional Networks for Biomedical Image Segmentation

Stable Diffusion: Text-to-Image

Text-to-Image Synthesis on LAION. 1.45B Model.

'A street sign that reads "Latent Diffusion" '	'A zombie in the style of Picasso'	'An image of an animal half mouse half octopus'	'An illustration of a slightly conscious neural network'	'A painting of a squirrel eating a burger'	'A watercolor painting of a chair that looks like an octopus'	'A shirt with the inscription: "I love generative models!" '

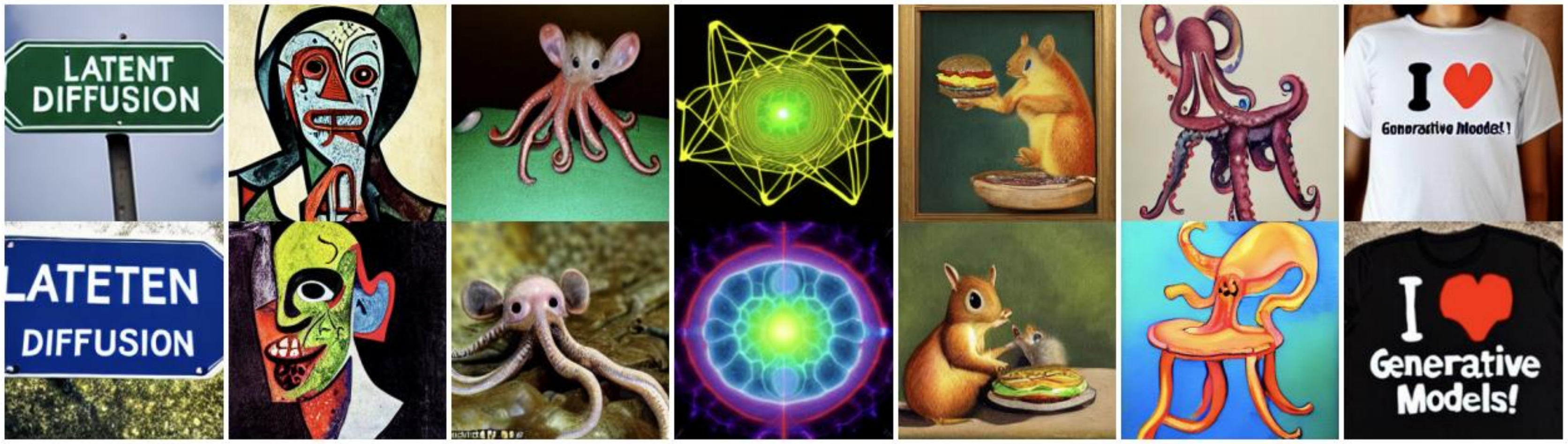


Figure 5. Samples for user-defined text prompts from our model for text-to-image synthesis, *LDM-8 (KL)*, which was trained on the LAION [78] database. Samples generated with 200 DDIM steps and $\eta = 1.0$. We use unconditional guidance [32] with $s = 10.0$.

Stable Diffusion: Layout-to-Image

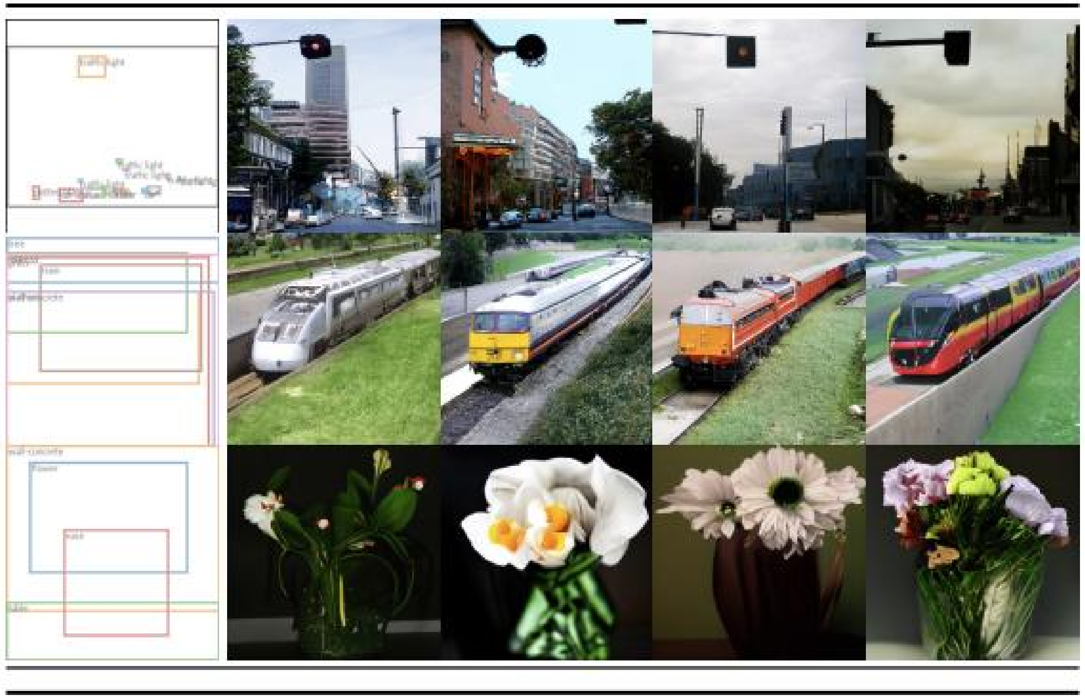


Figure 8. Layout-to-image synthesis with an *LDM* on COCO [4], see Sec. 4.3.1. Quantitative evaluation in the supplement D.3.

Stable Diffusion: super-resolution

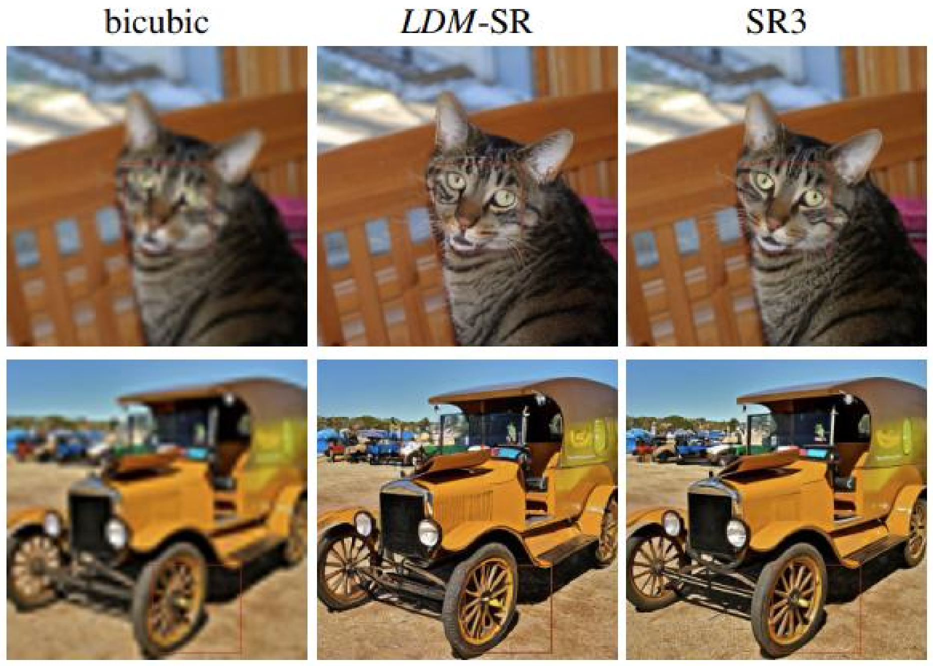


Figure 10. ImageNet 64→256 super-resolution on ImageNet-Val. *LDM-SR* has advantages at rendering realistic textures but SR3 can synthesize more coherent fine structures. See appendix for additional samples and cropouts. SR3 results from [72].

Stable Diffusion: inpainting

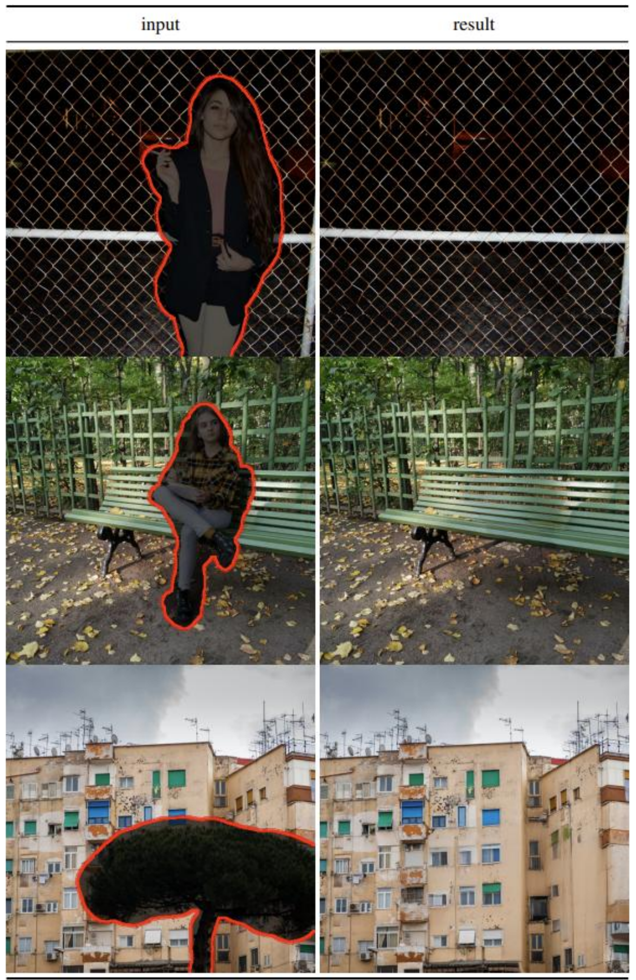


Figure 11. Qualitative results on object removal with our *big, w/ ft* inpainting model. For more results, see Fig. 22.

ControlNet

ControlNet is a neural network architecture that can enhance pretrained image diffusion models with task-specific conditions.

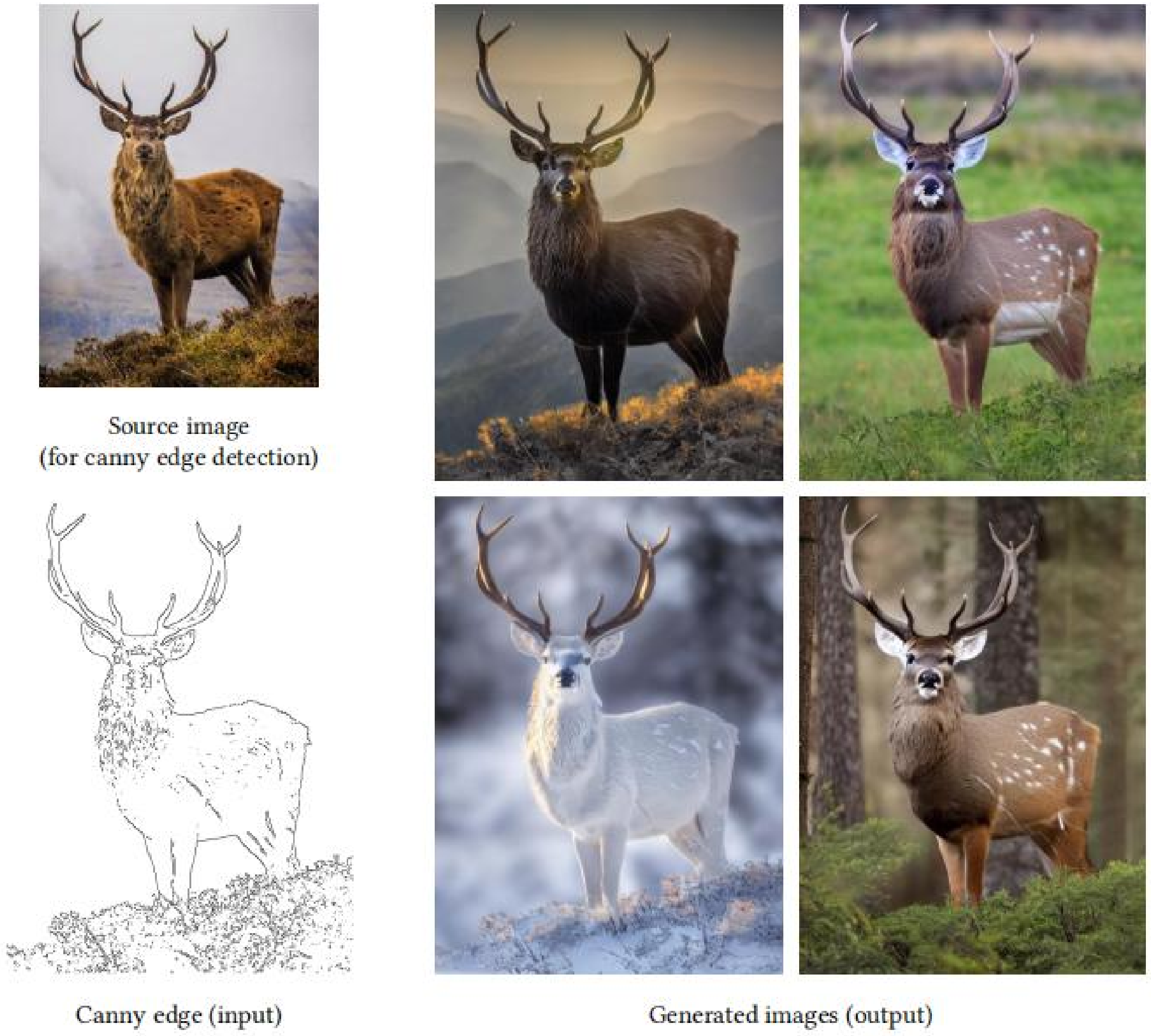


Figure 1: Control Stable Diffusion with Canny edge map. The canny edge map is input, and the source image is not used when we generate the images on the right. The outputs are achieved with a default prompt *"a high-quality, detailed, and professional image"*. This prompt is used in this paper as a default prompt that does not mention anything about the image contents and object names. Most of figures in this paper are high-resolution images and best viewed when zoomed in.

ControlNet

Z: “zero convolution”, i.e., 1×1 convolution layer with both weight and bias initialized with zeros. $\mathcal{Z}(I;\{W,B\})_{p,i} = B_i + \sum_j^c I_{p,i} W_{i,j}$

$$y = \mathcal{F}(x;\Theta) \qquad y_c = \mathcal{F}(x;\Theta) + \mathcal{Z}(\mathcal{F}(x + \mathcal{Z}(c;\Theta_{z1});\Theta_c);\Theta_{z2})$$

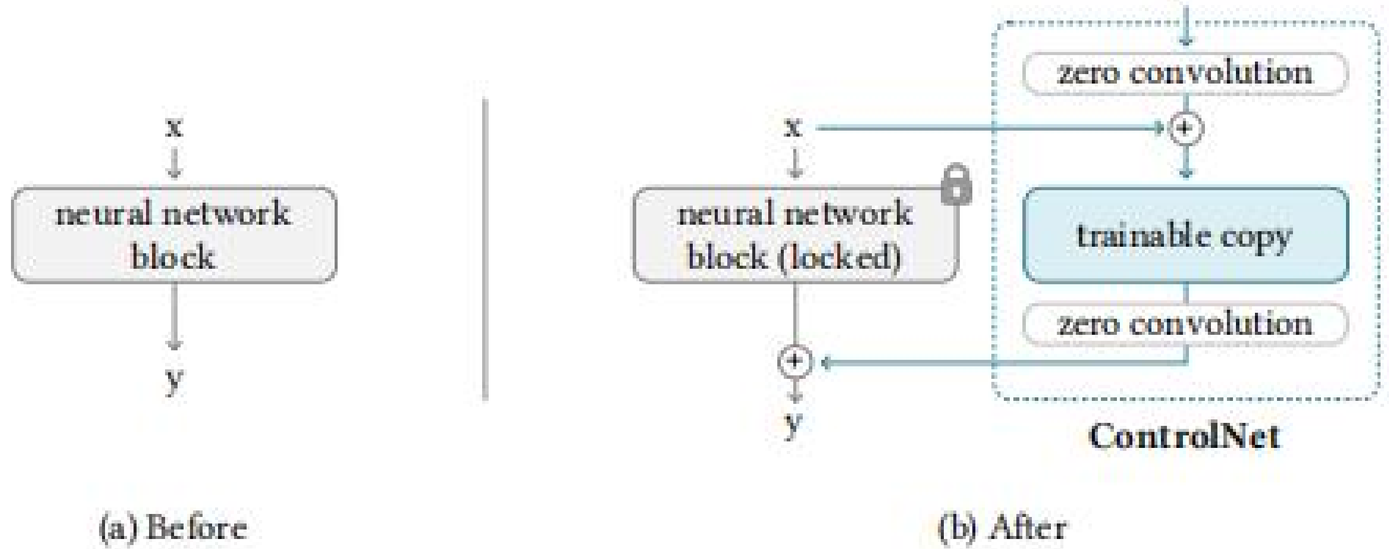


Figure 2: ControlNet. We show the approach to apply a ControlNet to an arbitrary neural network block. The x, y are deep features in neural networks. The “+” refers to feature addition. The “c” is an extra condition that we want to add to the neural network. The “zero convolution” is an 1×1 convolution layer with both weight and bias initialized as zeros.

ControlNet in Stable Diffusion

The texts are encoded by OpenAI CLIP

diffusion time steps are encoded by positional encoding.

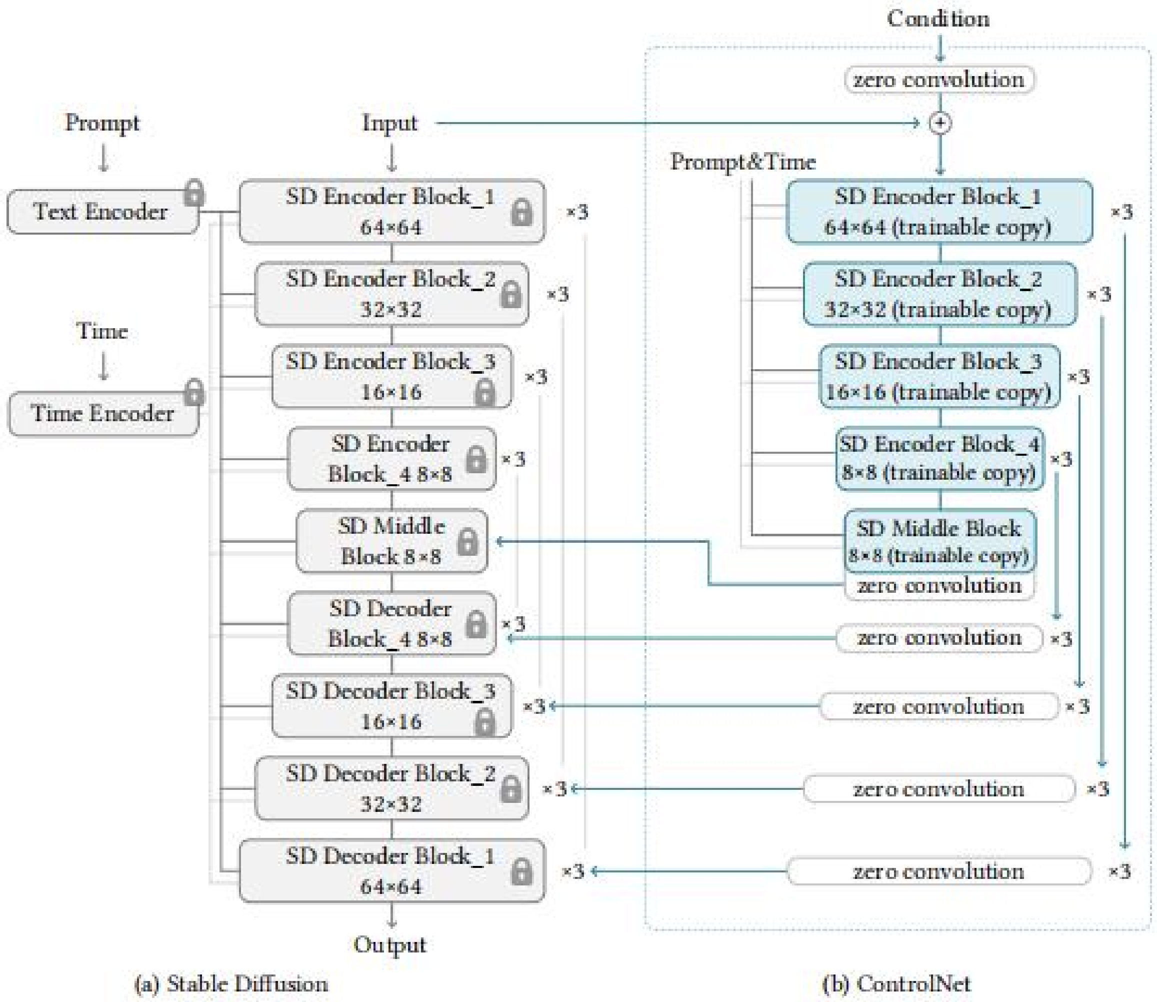


Figure 3: ControlNet in Stable Diffusion. The gray blocks are the structure of Stable Diffusion 1.5 (or SD V2.1, since they use the same U-Net architecture), while the blue blocks are ControlNet.

Controlnet tries to revise/control the latent code (middle block).
It does not change the decoder.

ControlNet

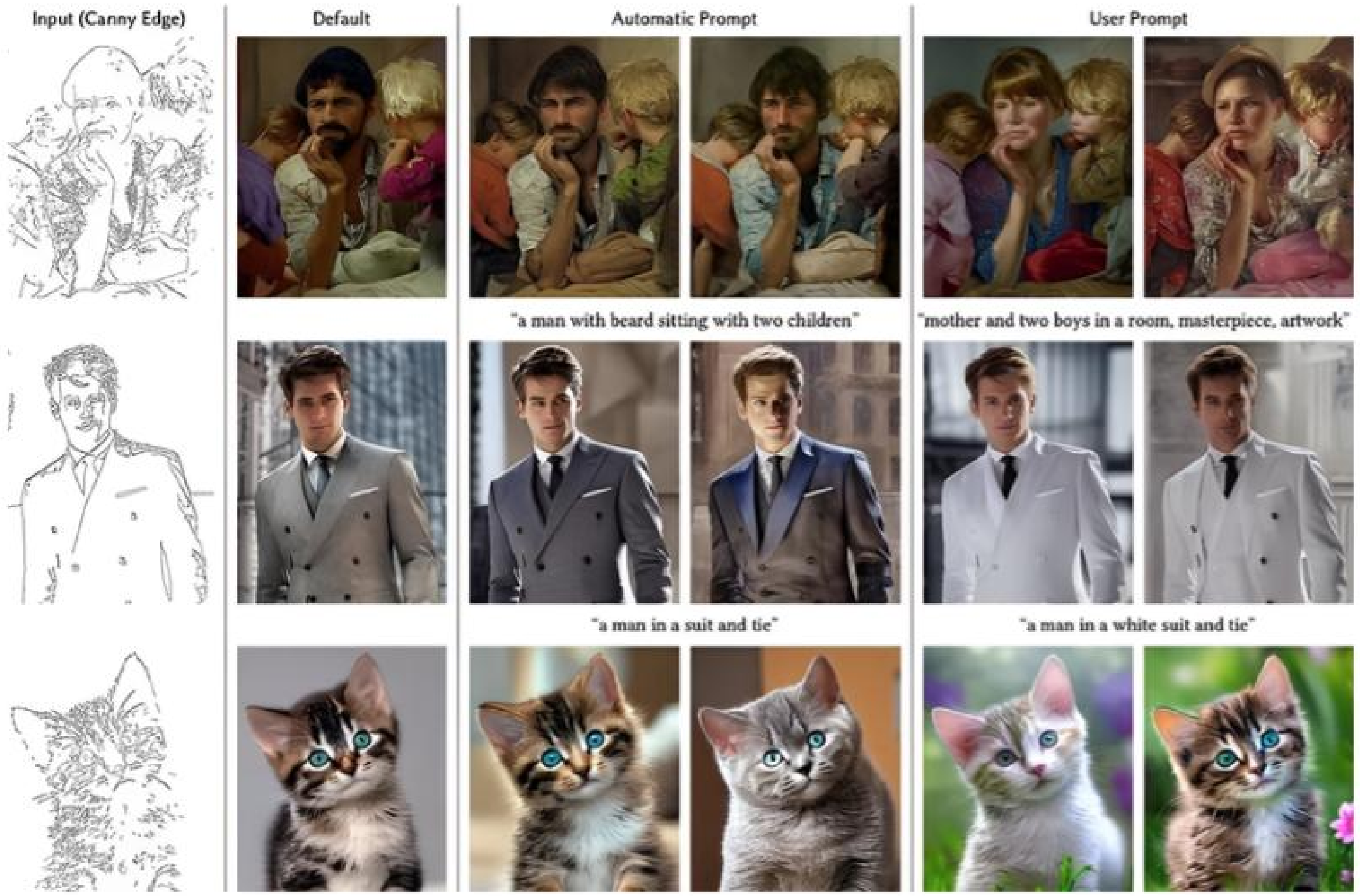
- the locked copy preserves the network capability learned from billions of images
- the trainable copy is trained on task-specific datasets to learn the conditional control.
- Since the zero convolution does not add new noise to deep features, the training is as fast as fine tuning a diffusion model, compared to training new layers from scratch.

Training

- Loss: $\mathcal{L} = \mathbb{E}_{\boldsymbol{z}_0, t, \boldsymbol{c}_t, \boldsymbol{c}_\mathrm{f}, \epsilon \sim \mathcal{N}(0,1)} \Big[\| \epsilon - \epsilon_\theta(z_t, t, \boldsymbol{c}_t, \boldsymbol{c}_\mathrm{f}) \|_2^2 \Big]$

- During the training, we randomly replace 50% text prompts c_t with empty strings. This facilitates ControlNet's capability to recognize semantic contents from input condition maps, e.g., Canny edge maps or human scribbles, etc.
- This is mainly because when the prompt is not visible for the SD model, the encoder tends to learn more semantics from input control maps as a replacement for the prompt.

- Small-Scale Training: When computation device is limited, one can accelerate convergence by disconnecting the link to decoder 1,2,3,4 and only connecting the middle block (this can improve the training speed by about a factor of 1.6) use BLIP to generate captions


Input (Canny Edge)
Default
Automatic Prompt
User Prompt
"a man with beard sitting with two children"
"mother and two boys in a room, masterpiece, artwork"
"a man in a suit and tie"
"a man in a white suit and tie"

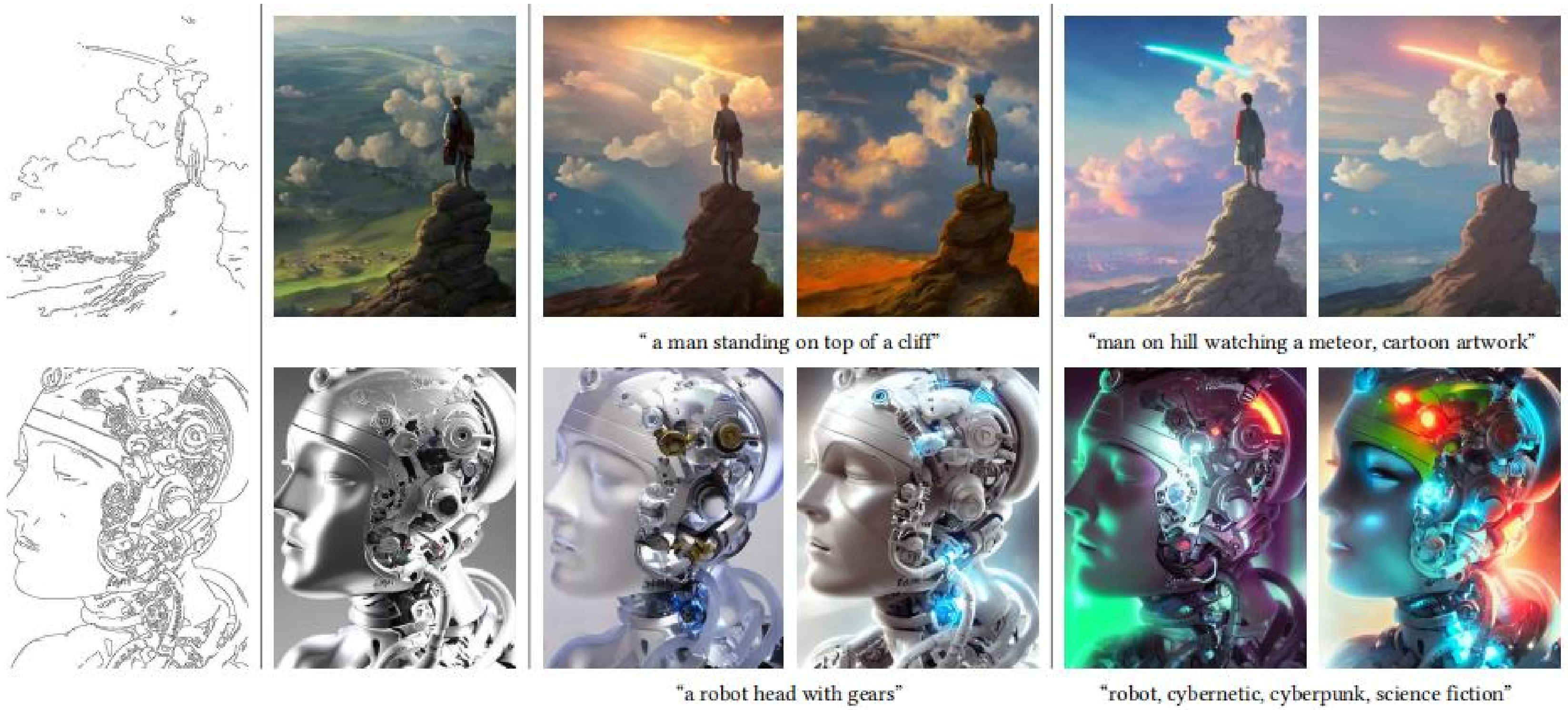

" a man standing on top of a cliff"
"man on hill watching a meteor, cartoon artwork"
"a robot head with gears"
"robot, cybernetic, cyberpunk, science fiction"

Figure 4: Controlling Stable Diffusion with Canny edges. The "automatic prompts" are generated by BLIP based on the default result images without using user prompts. See also the Appendix for source images for canny edge detection.

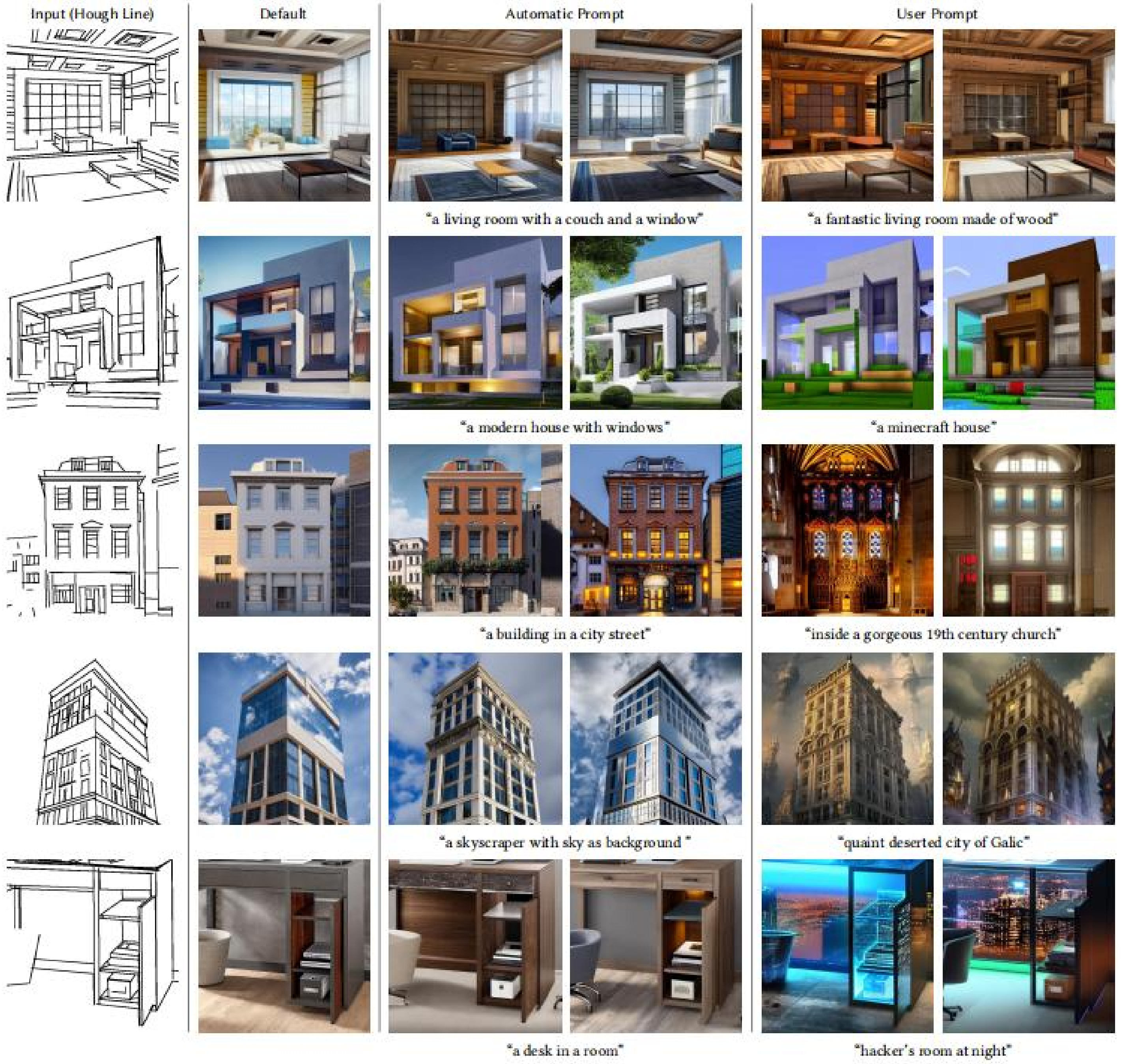

Input (Hough Line)
Default
Automatic Prompt
User Prompt
"a living room with a couch and a window"
"a fantastic living room made of wood"
"a modern house with windows"
"a minecraft house"
"a building in a city street"
"inside a gorgeous 19th century church"
"a skyscraper with sky as background "
"quaint deserted city of Galic"
"a desk in a room"
"hacker's room at night"

Figure 5: Controlling Stable Diffusion with Hough lines (M-LSD). The "automatic prompts" are generated by BLIP based on the default

"Michael Jackson's concert"

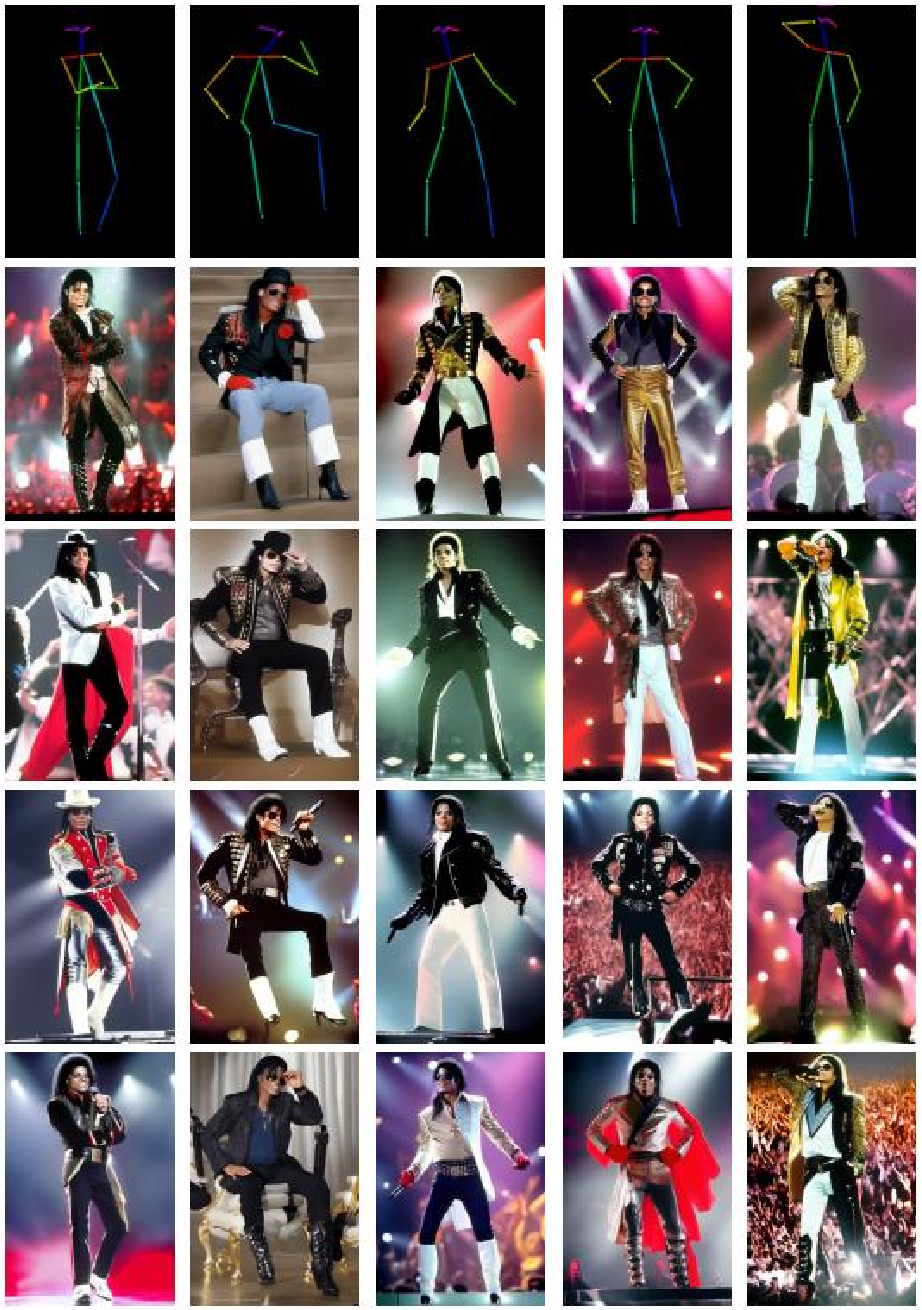


Figure 10: Controlling Stable Diffusion with human pose to generate different poses for a same person ("Michael Jackson's concert"). Images are not cherry picked. See also the Appendix for source images for Openpose pose detection.

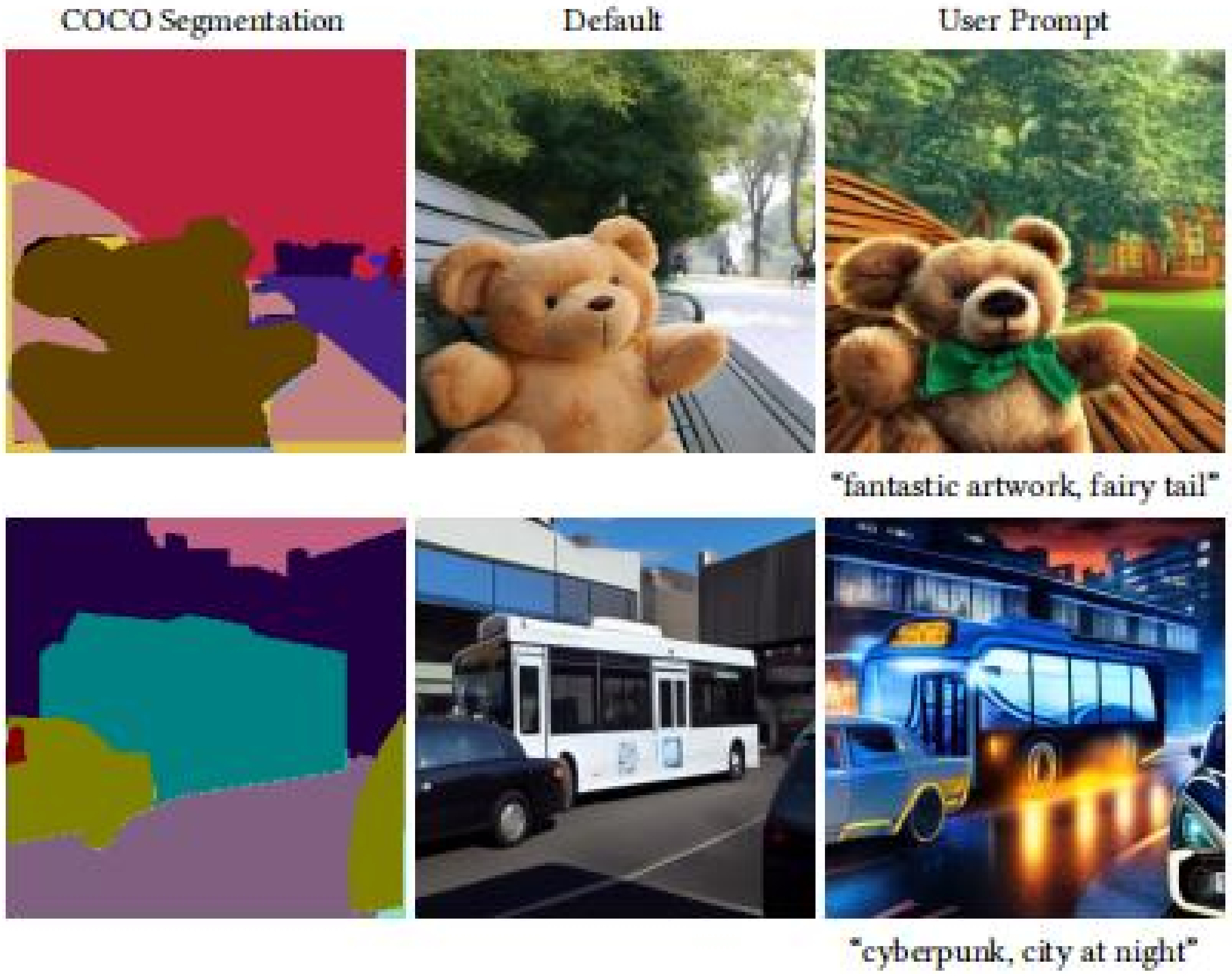


Figure 12: Controlling Stable Diffusion with COCO-Stuff [4] segmentation map.

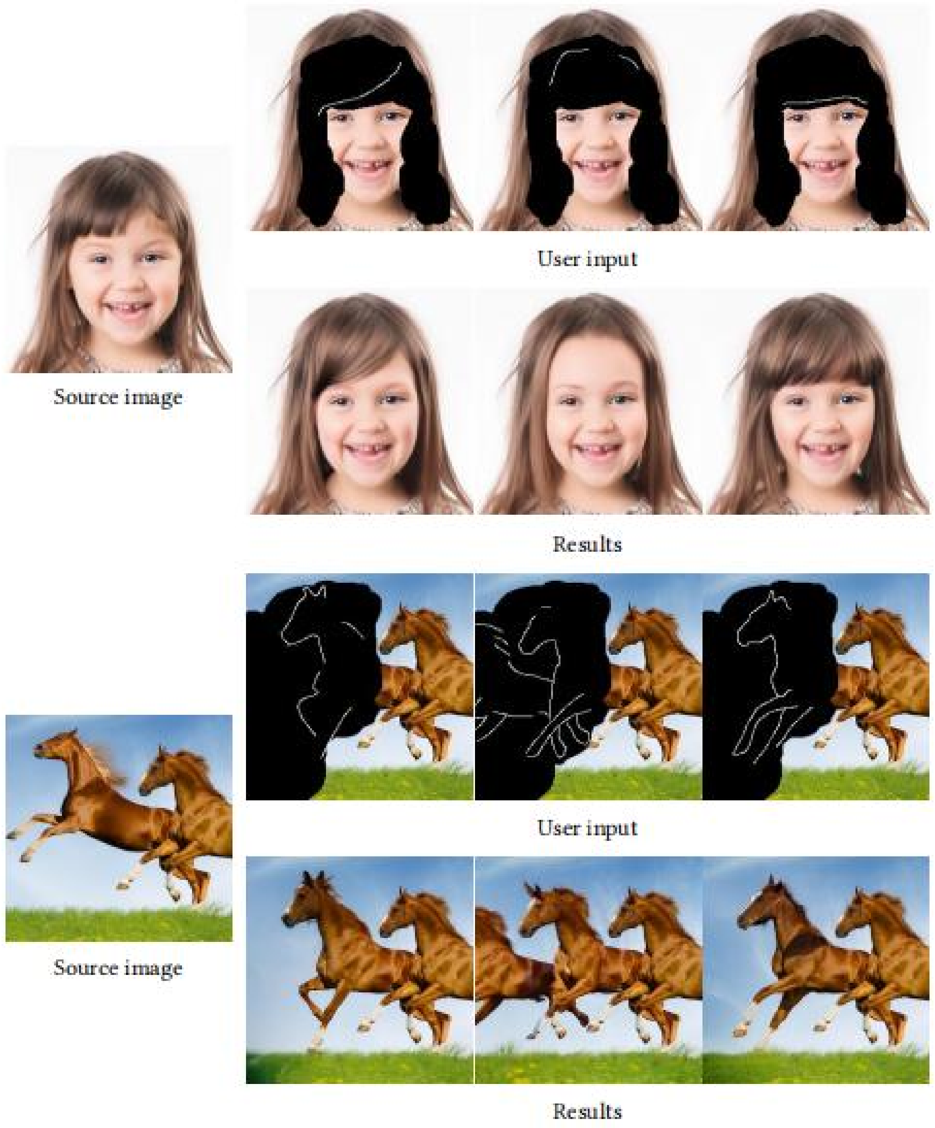
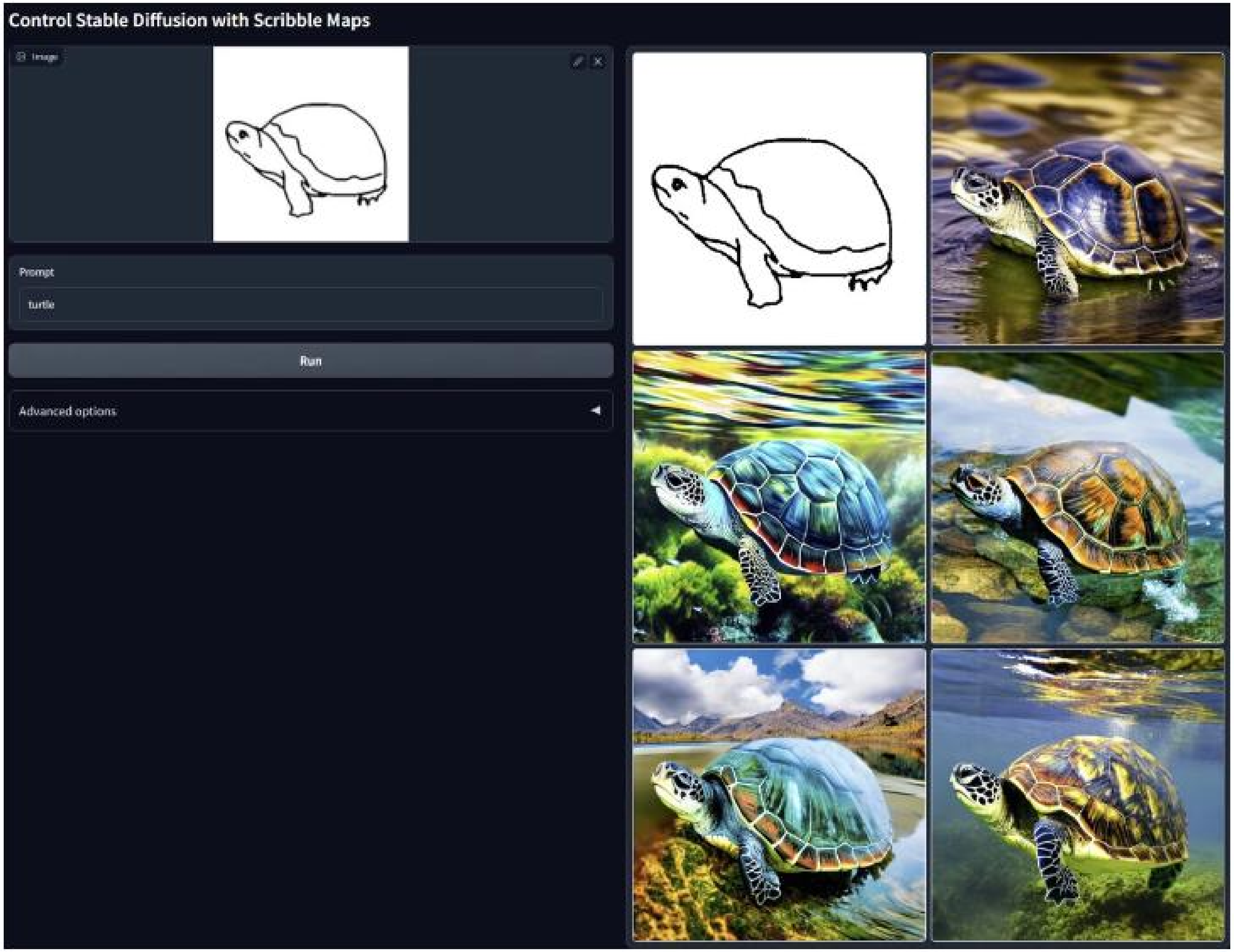


Figure 16: Masked Diffusion. By diffusing images in masked areas, the Canny-edge model can be used to support pen-based editing of image contents. Since all diffusion models naturally support masked diffusion, the other models are also likely to be used in manipulating images.



To use it, **you have to download it as an extension in WebUI.** Go to the Extensions tab and choose the Install from URL option, then paste this link where it says the extension's git repository URL: https://github.com/Mikubill/sd-webui-controlnet.

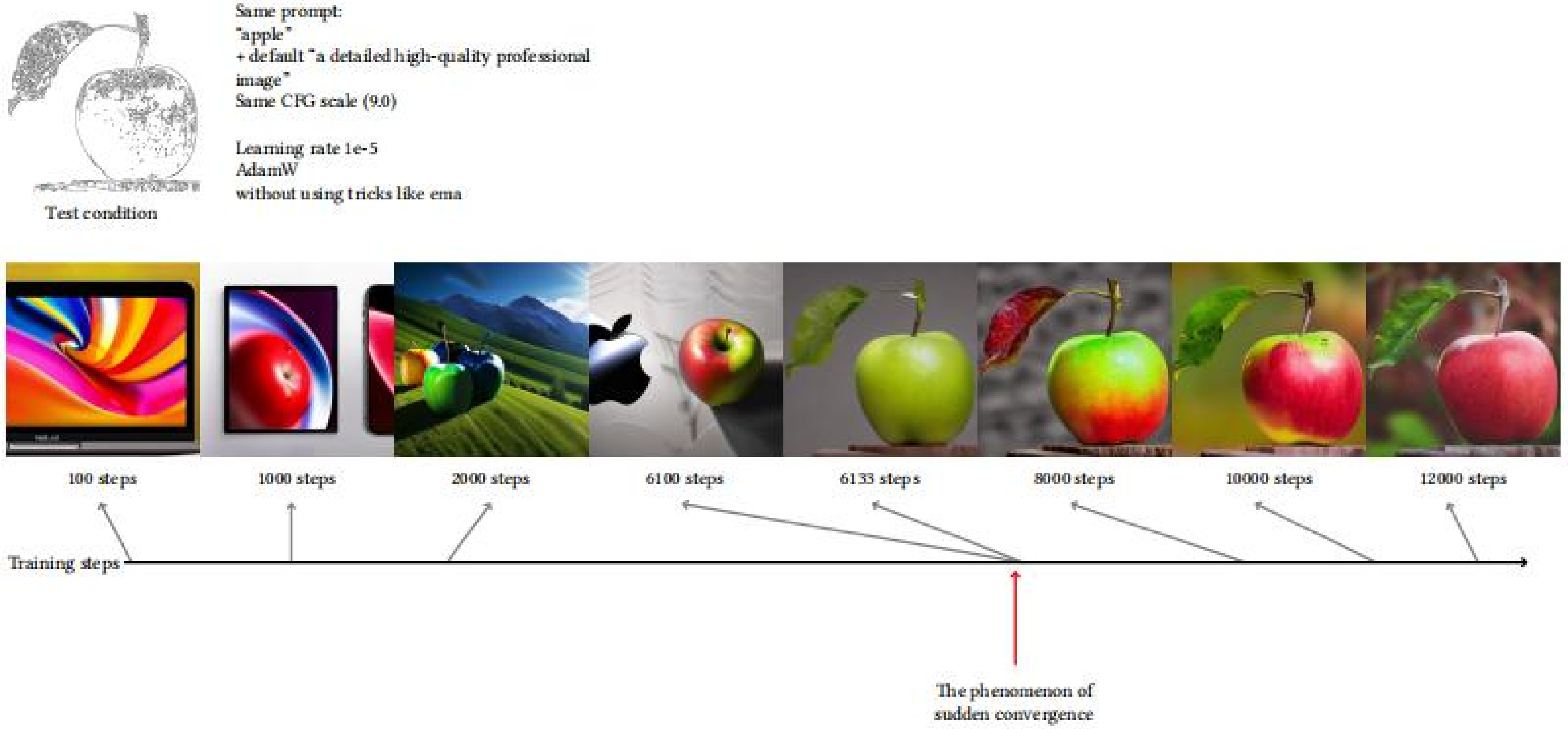
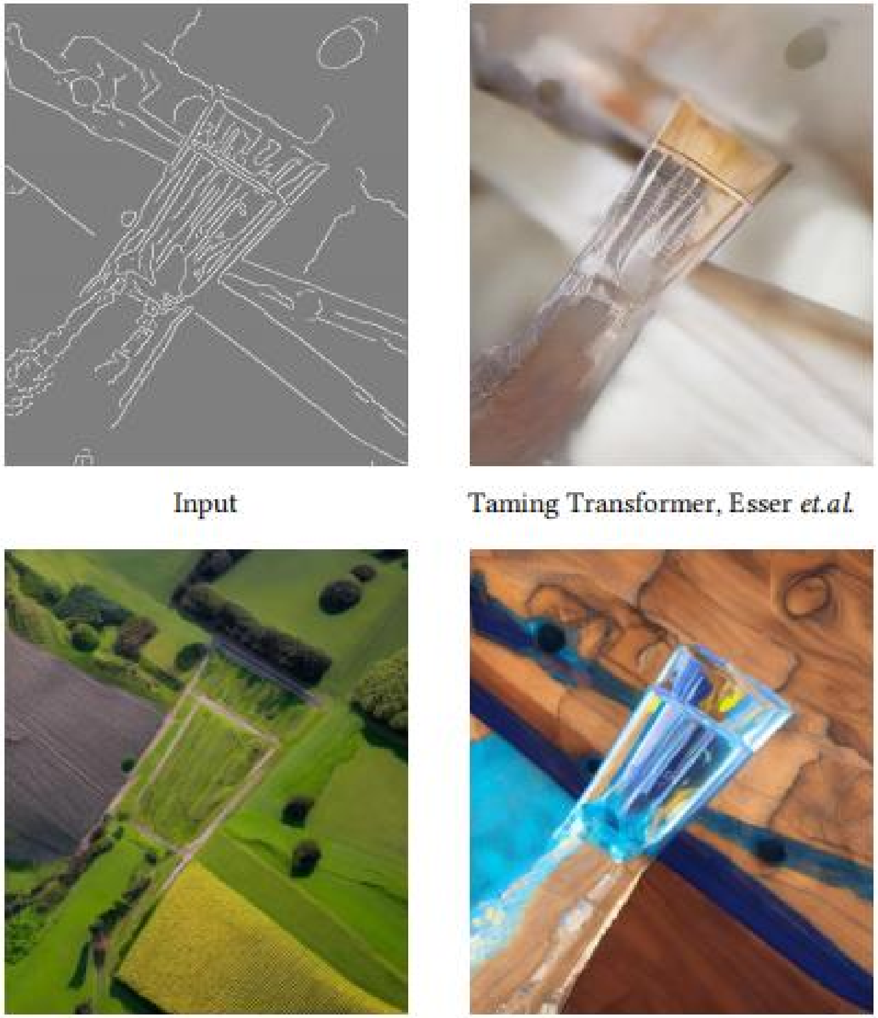


Figure 21: The sudden converge phenomenon. Because we use zero convolutions, the neural network always predict high-quality images during the entire training. At a certain point of training step, the model suddenly learns to adapt to the input conditions. We call this "sudden converge phenomenon".



Input

Taming Transformer, Esser *et.al.*

Ours default
(Seems to be interpreted as a bird's eye view of an agricultural field)

Ours "a glass of water"
(Seems unable to eliminate the effects of mistaken recognitions)

Figure 28: Limitation. When the semantic of input image is mistakenly recognized, the negative effects seem difficult to be eliminated, even if a strong prompt is provided.