3D Vision and
Robotics Lab

[ICCV 2023] ControlNet: Adding Conditional Control to Text-to-Image Diffusion Models

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala
Stanford University

Gyeongsu Cho @UNIST

Lab Seminar 2024.01.18. (Thu)

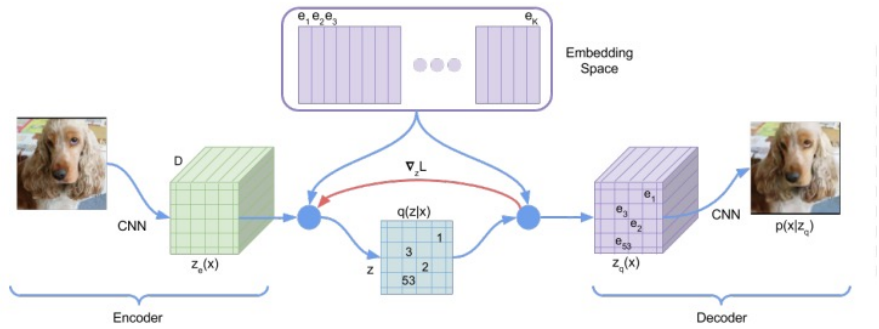
Contents

Introduction

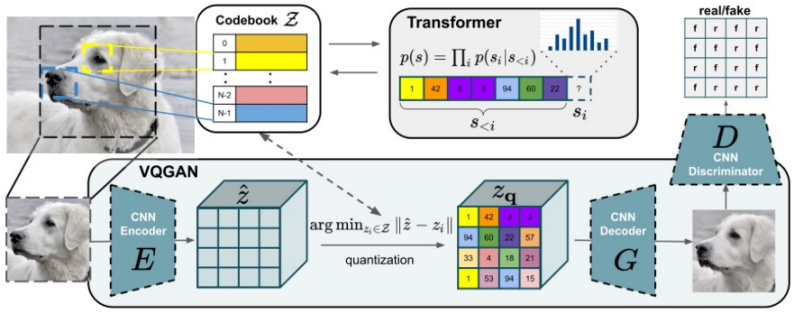
Controlling Stable Diffusion with learned conditions

Introduction

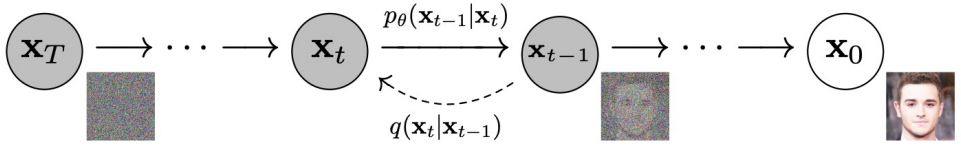
[NIPS 2017] VQVAE



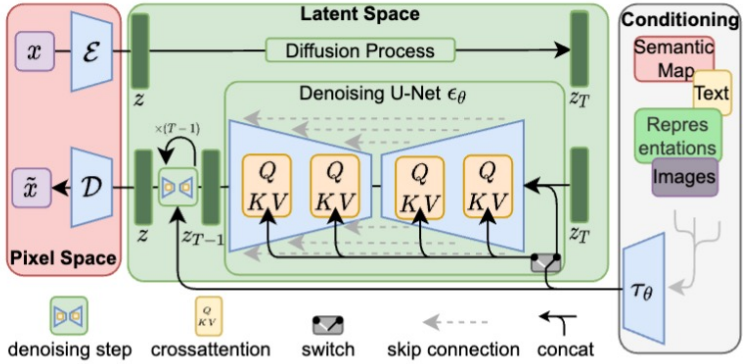
[CVPR 2021] VQGAN



[NeurIPS 2020] DDPM



[CVPR 2022] Latent-Diffusion



Introduction

Stable Diffusion?

Introduction

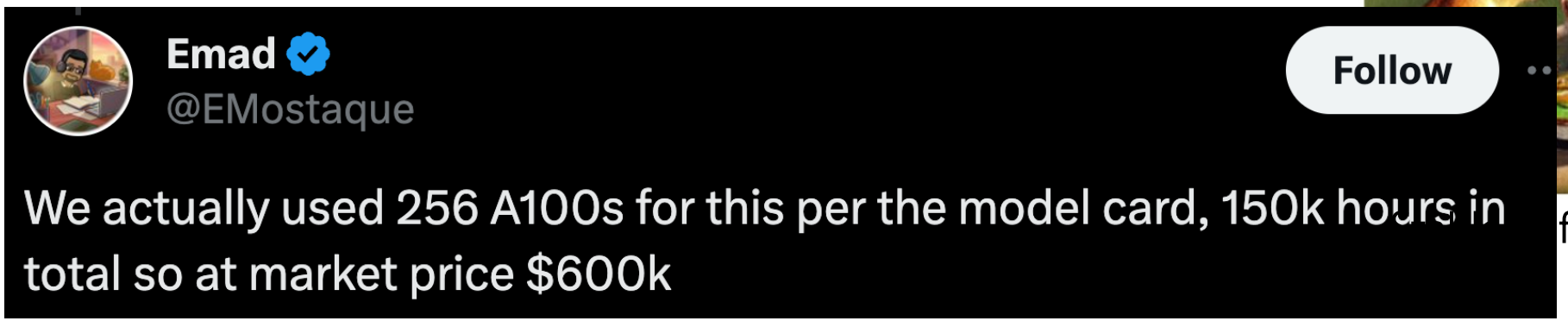
Limitations of Stable Diffusion

1. It doesn't generate the image that we desire.

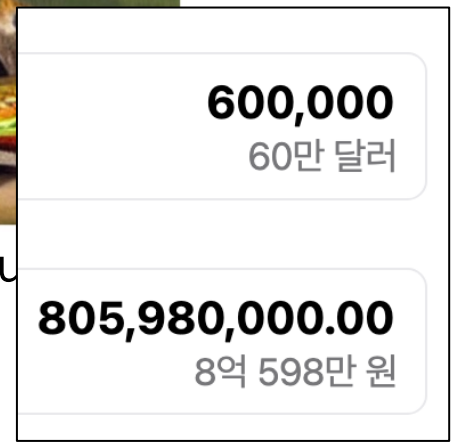
2. It is hard to train!

'A painting of a squirrel eating a burger'

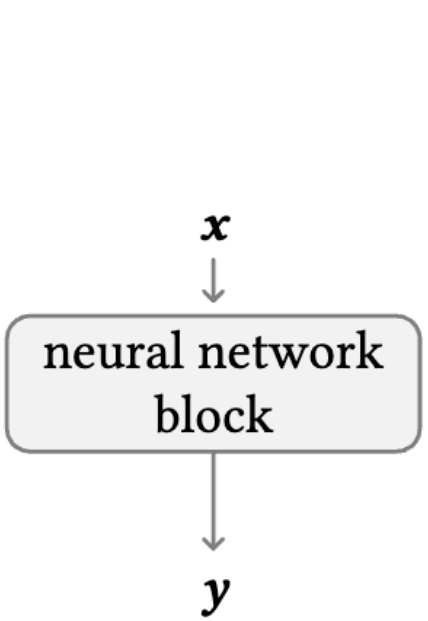
How can we solve it?



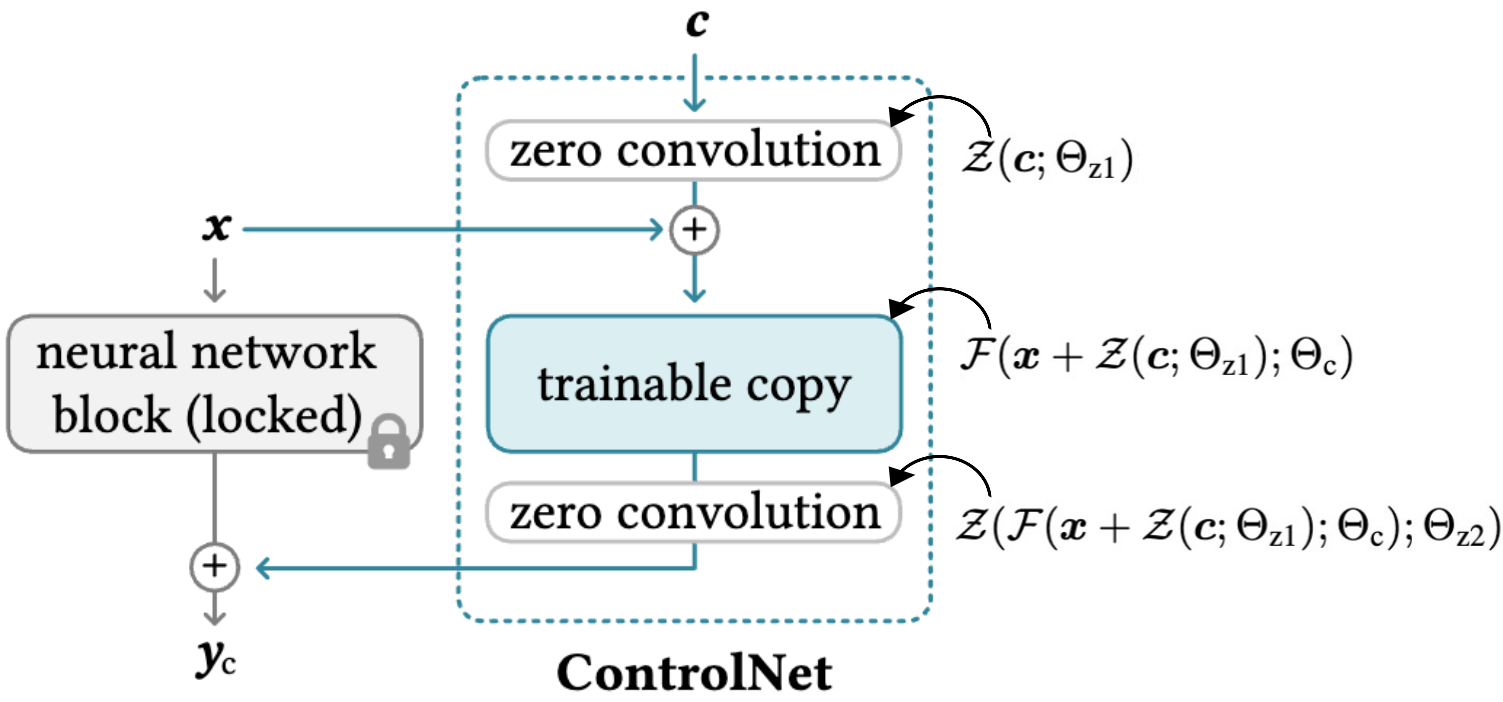
Stability.ai CEO's tweets



Method



(a) Before



(b) After

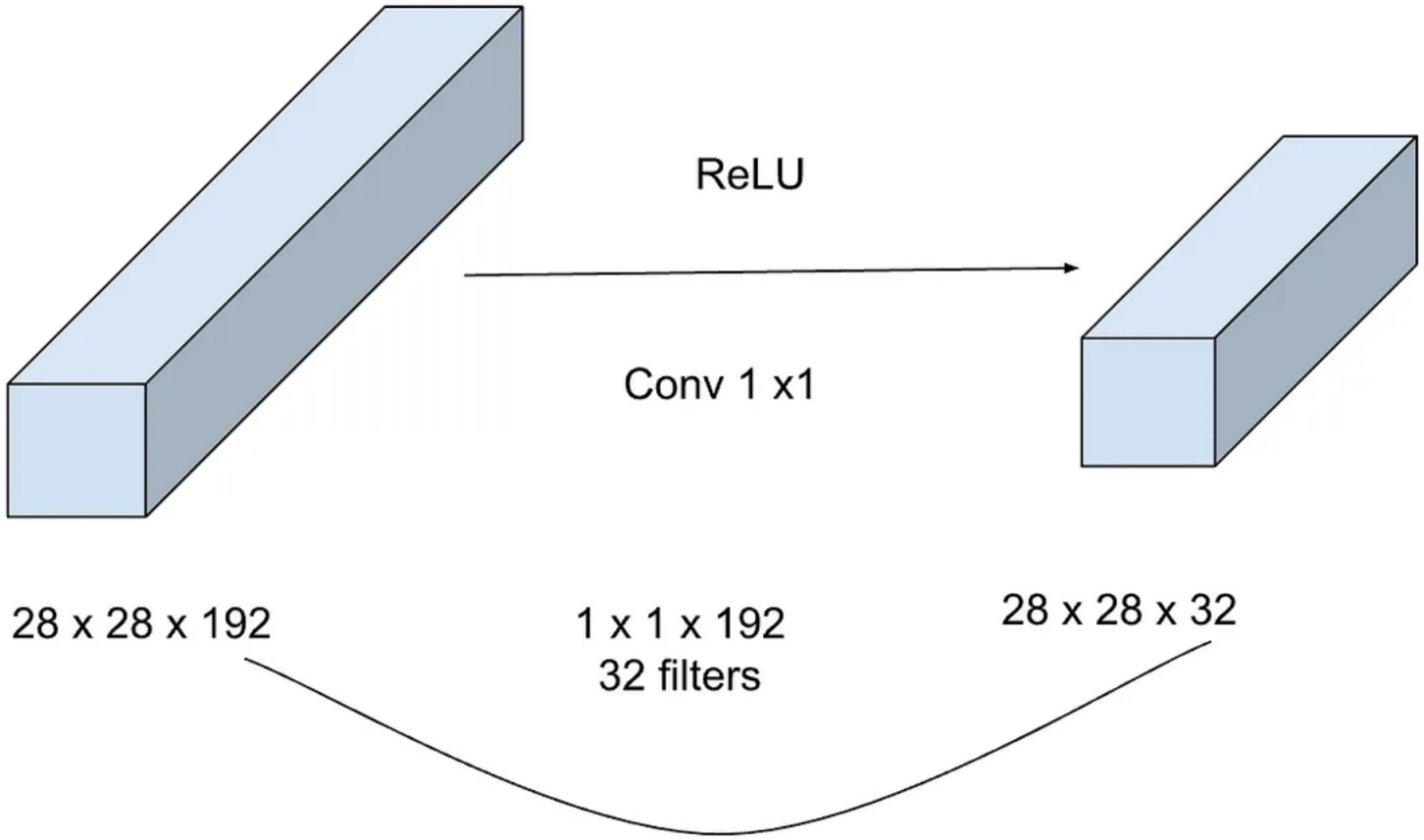
$$y = \mathcal{F}(x;\Theta)$$

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

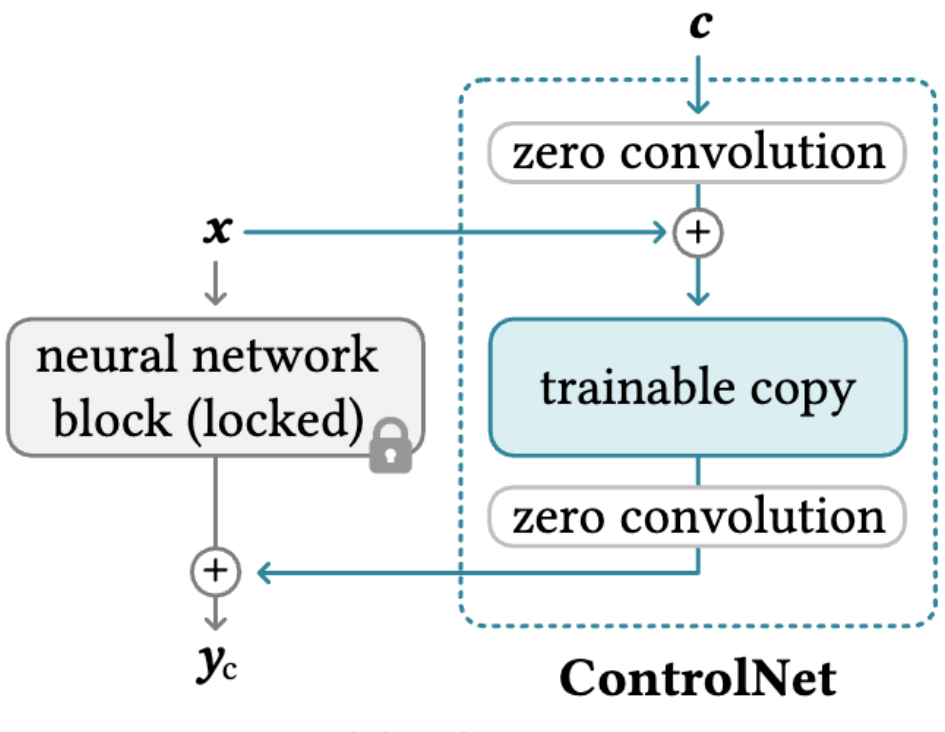
Method

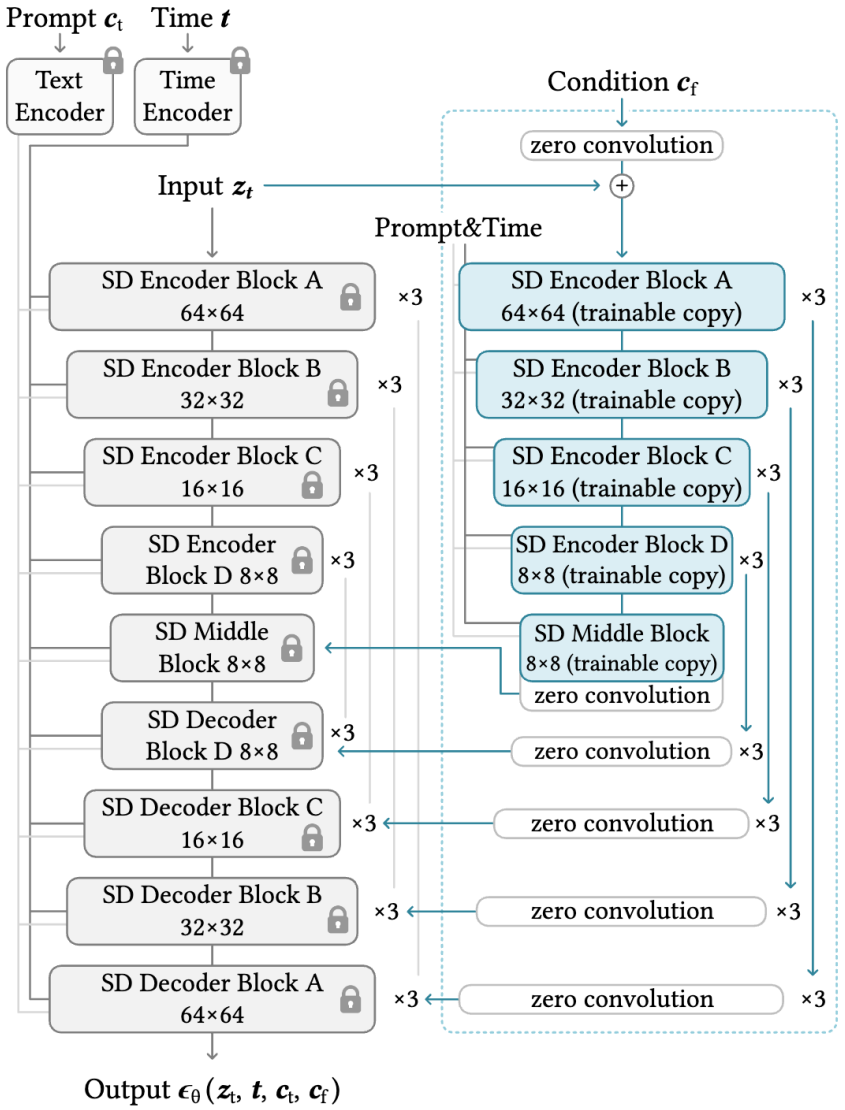
Zero Convolution?

Zero convolution layers : 1×1 convolution with both **weight and bias** initialized to **zero**.



Method





$$\mathcal{L} = \mathbb{E}_{\boldsymbol{z}_0, \boldsymbol{t}, \boldsymbol{c}_t, \boldsymbol{c}_f, \epsilon \sim \mathcal{N}(0,1)} \left[\| \epsilon - \epsilon_\theta(\boldsymbol{z}_t, \boldsymbol{t}, \boldsymbol{c}_t, \boldsymbol{c}_f)) \|_2^2 \right],$$

Experiments

- Dataset

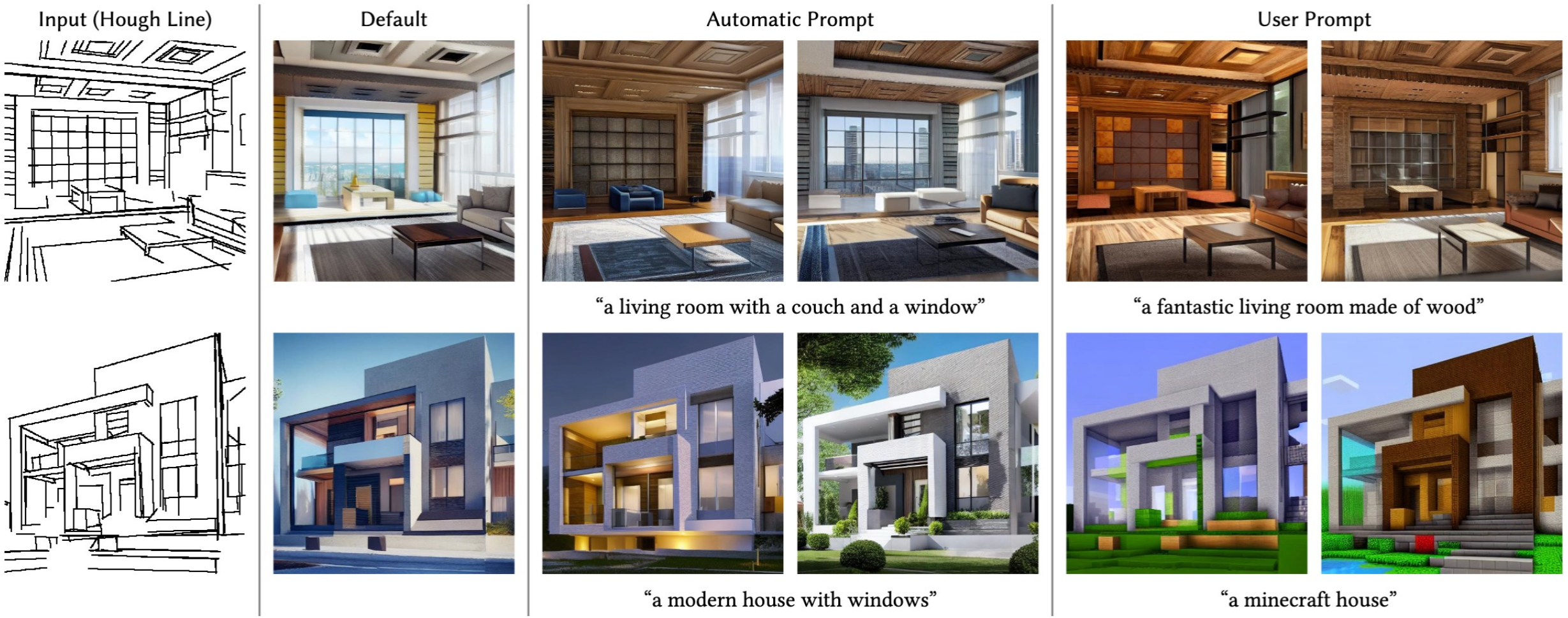
  Conditions + Images + Captions(BLiP)

Conditions	Training samples	Training GPU Type and Hours	Base model
Canny Edge	3M Internet	~600 A100	Stable Diffusion V1.5
Hough Line	600K Places2	~150 A100	Resumed from the Canny model
HED Boundary	3M Internet	~300 A100	Stable Diffusion V1.5
User Sketching	500K Internet	~150 A100	Resumed from the Canny model
Human Pose (Openpifpaf)	200K Openpifpaf	~400 3090TI	Stable Diffusion V2.1
Human Pose (Openpose)	200K Openpose	~300 A100	Stable Diffusion V1.5
Semantic Mask (COCO)	164K COCO	~400 3090TI	Stable Diffusion V1.5
Semantic Mask (ADE20K)	20K ADE20K	~200 A100	Stable Diffusion V1.5
Depth	3M Internet	~500 A100	Stable Diffusion V1.5
Normal Maps	25K DIODE	~100 A100	Stable Diffusion V1.5
Cartoon Line Drawing	1M Internet	~300 A100	Waifu Diffusion

Experiments

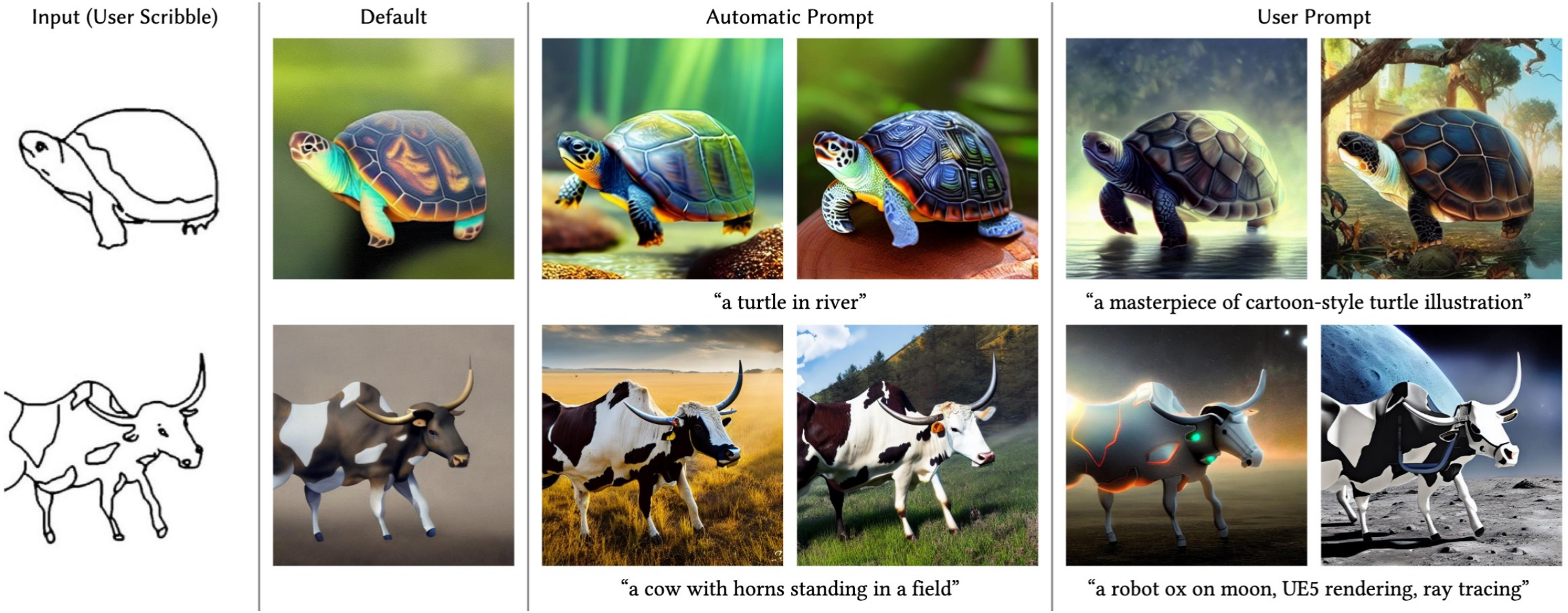
Controlling Stable Diffusion with Canny edges

Experiments



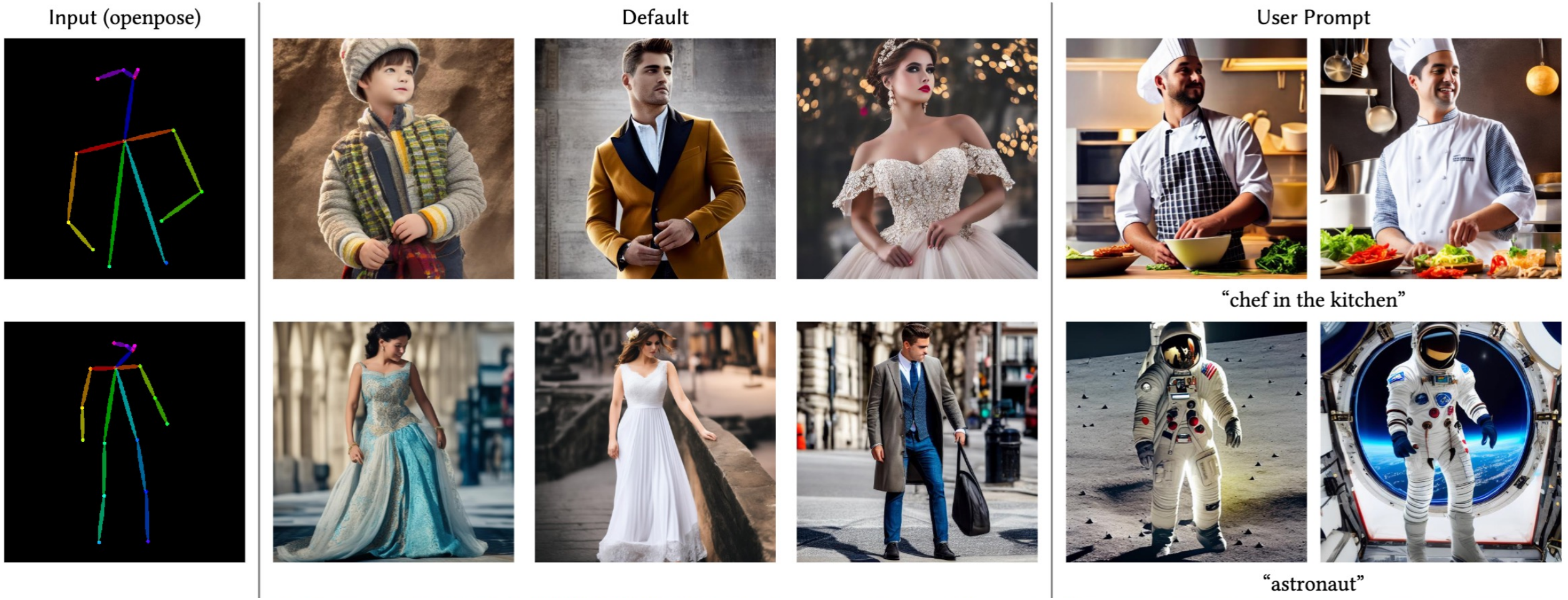
Controlling Stable Diffusion with Hough lines

Experiments



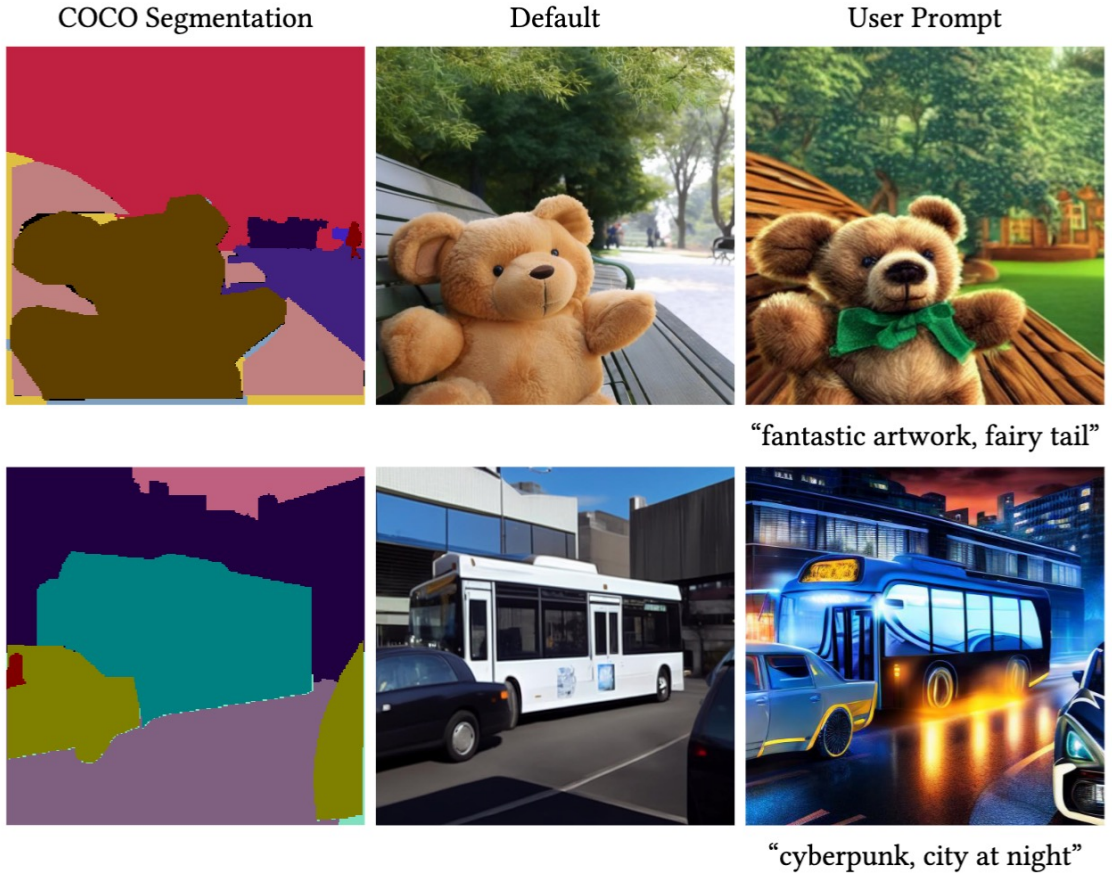
Controlling Stable Diffusion with User scribbles

Experiments

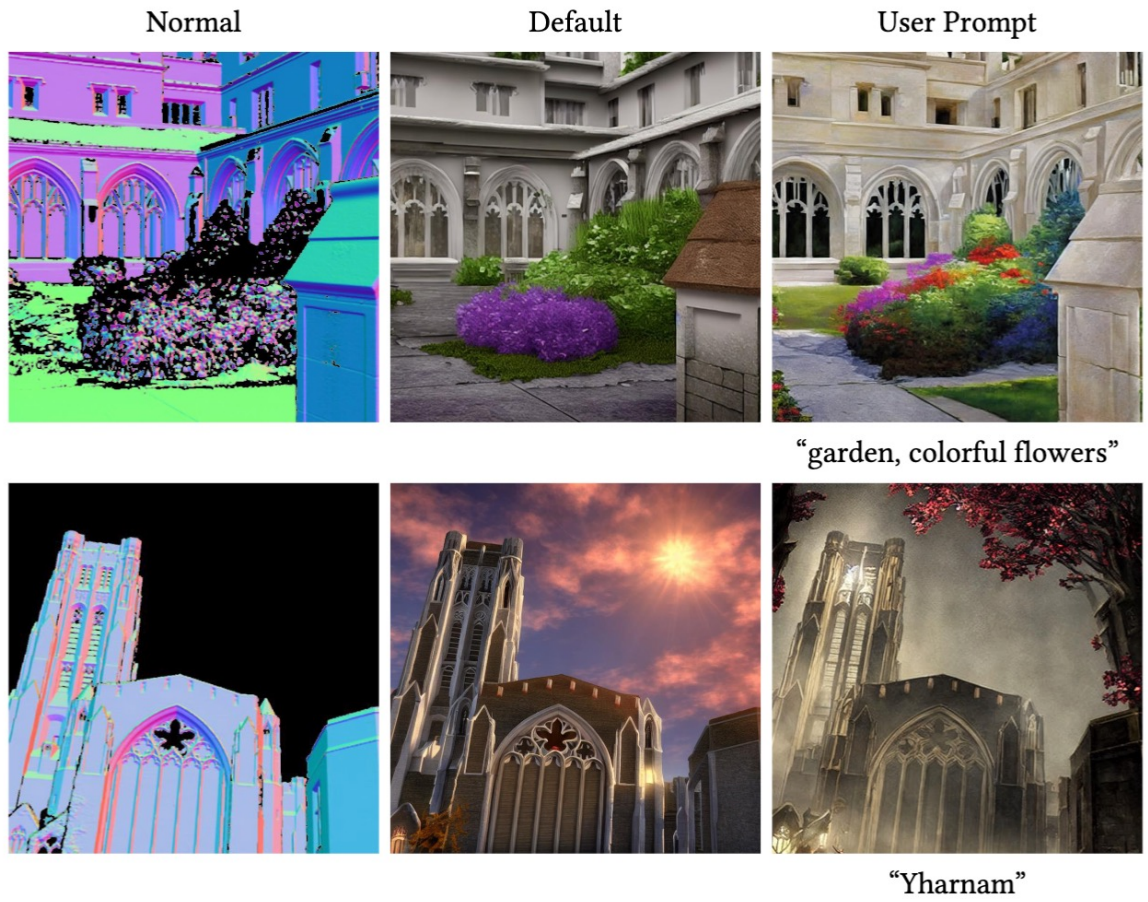


Controlling Stable Diffusion with Human poses

Experiments



Controlling Stable Diffusion with Segmentation



Controlling Stable Diffusion with normal map

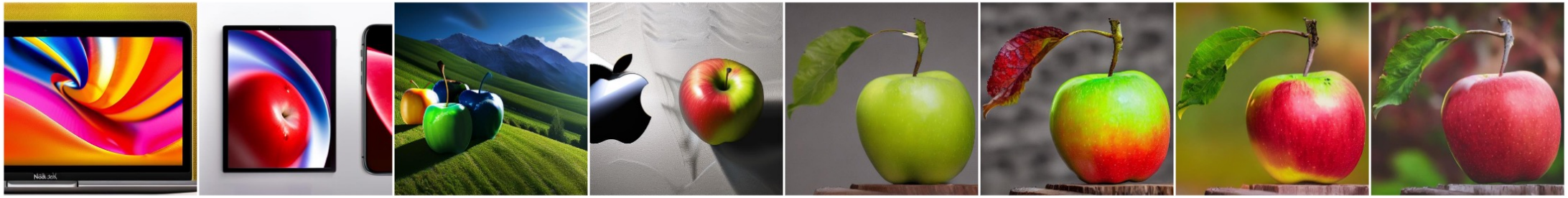
Experiments

The sudden convergence phenomenon

Test condition

Same prompt:
"apple"
+ default "a detailed high-quality professional image"
Same CFG scale (9.0)

Learning rate 1e-5
AdamW
without using tricks like ema



100 steps | 1000 steps | 2000 steps | 6100 steps | 6133 steps | 8000 steps | 10000 steps | 12000 steps

Training steps

The phenomenon of sudden convergence

Experiments

Interpreting contents

Input "a high-quality and extremely detailed image"

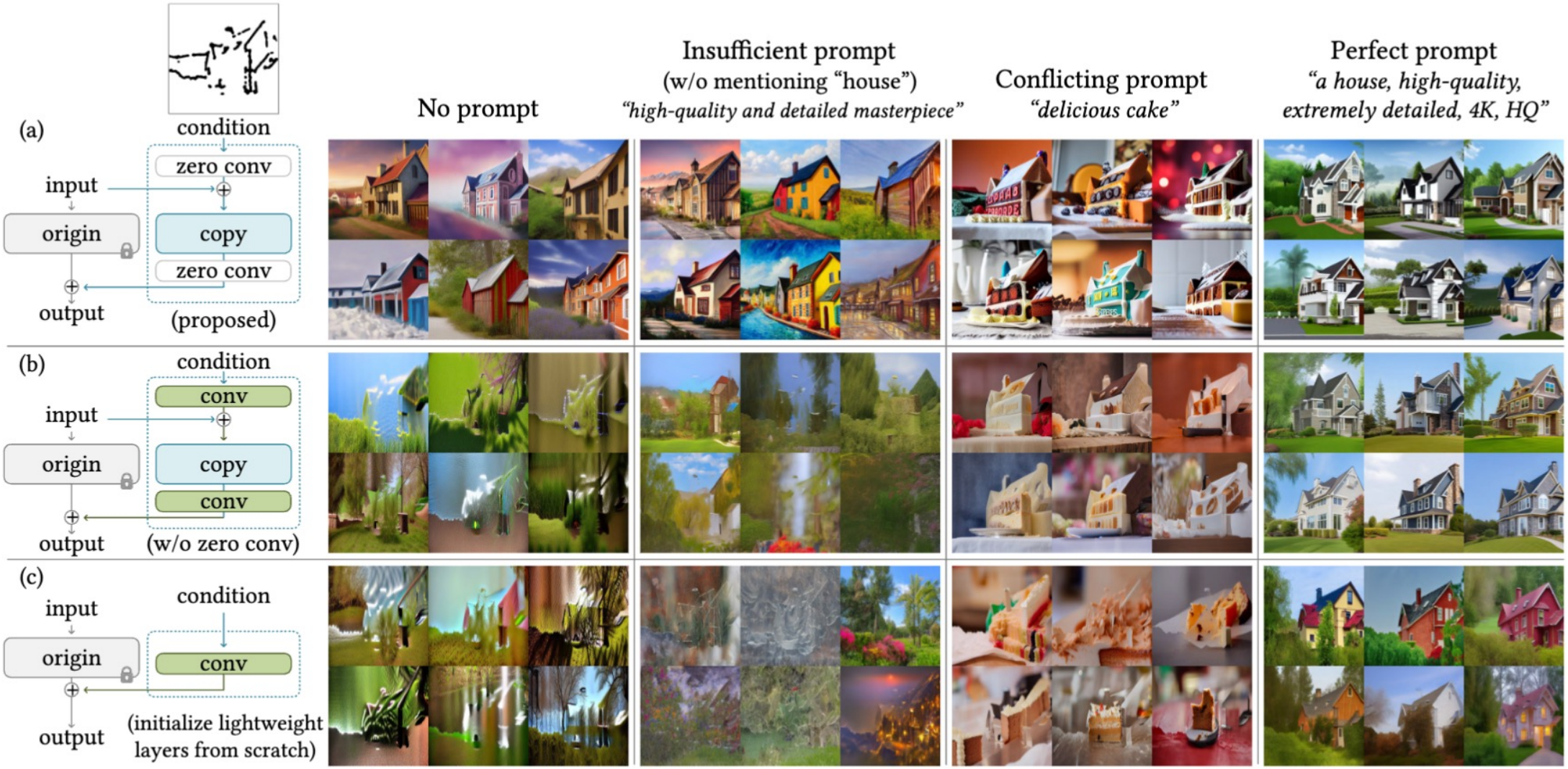
Experiments

Other community models

Transferring pretrained ControlNets to community models

Experiments

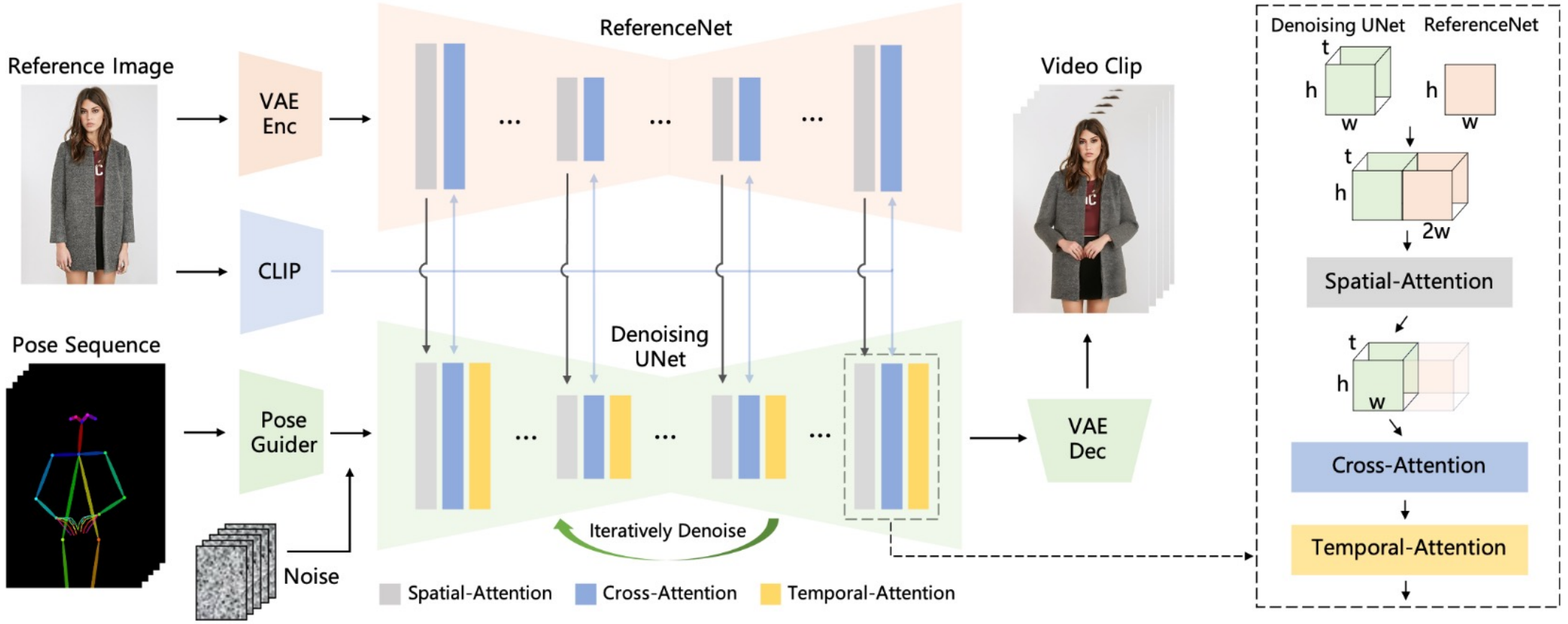
Ablation study



Ablative study of different architectures on a sketch condition and different prompt settings

Conclusion

Animate Anyone

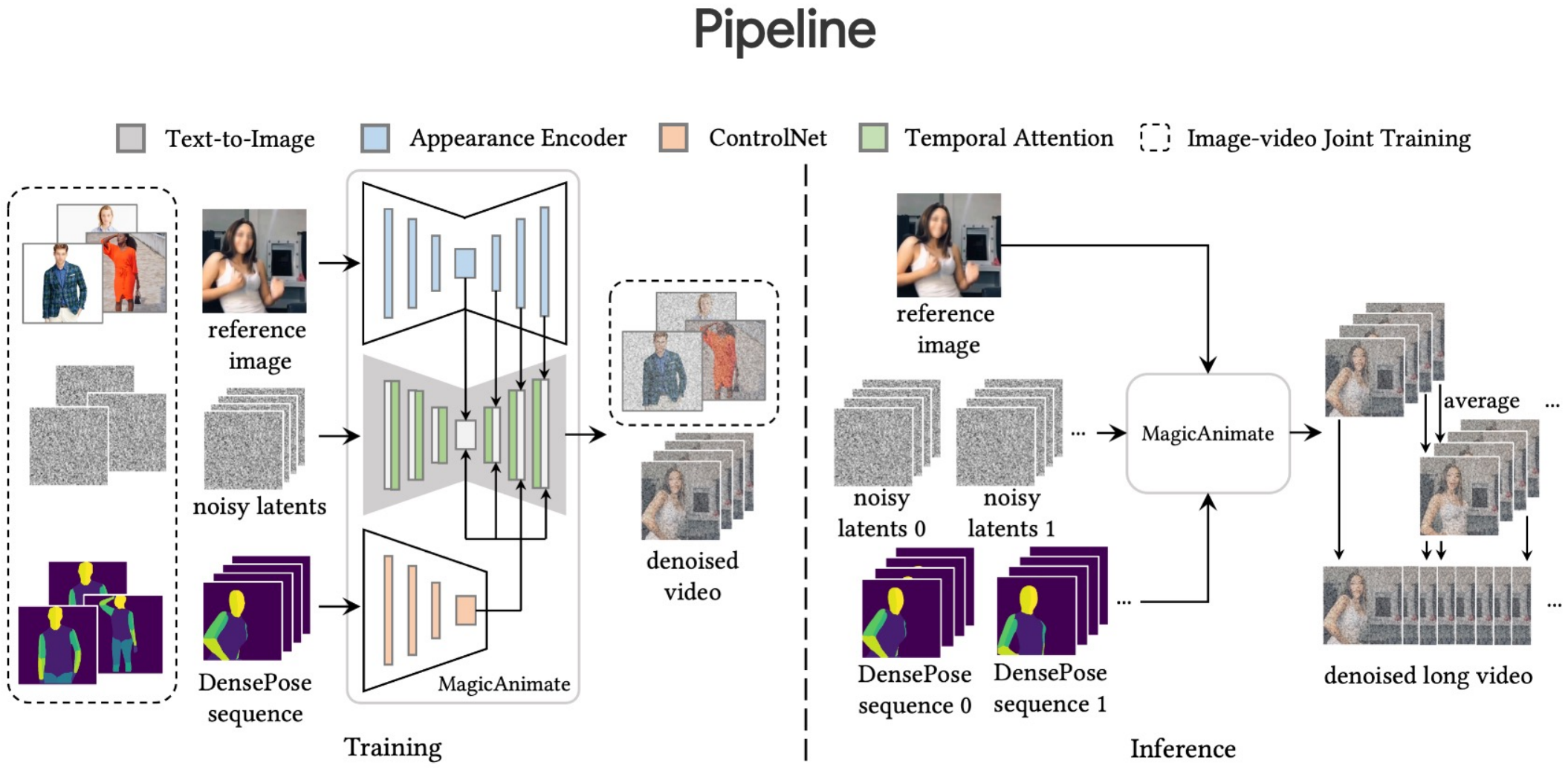


Conclusion

Animate Anyone

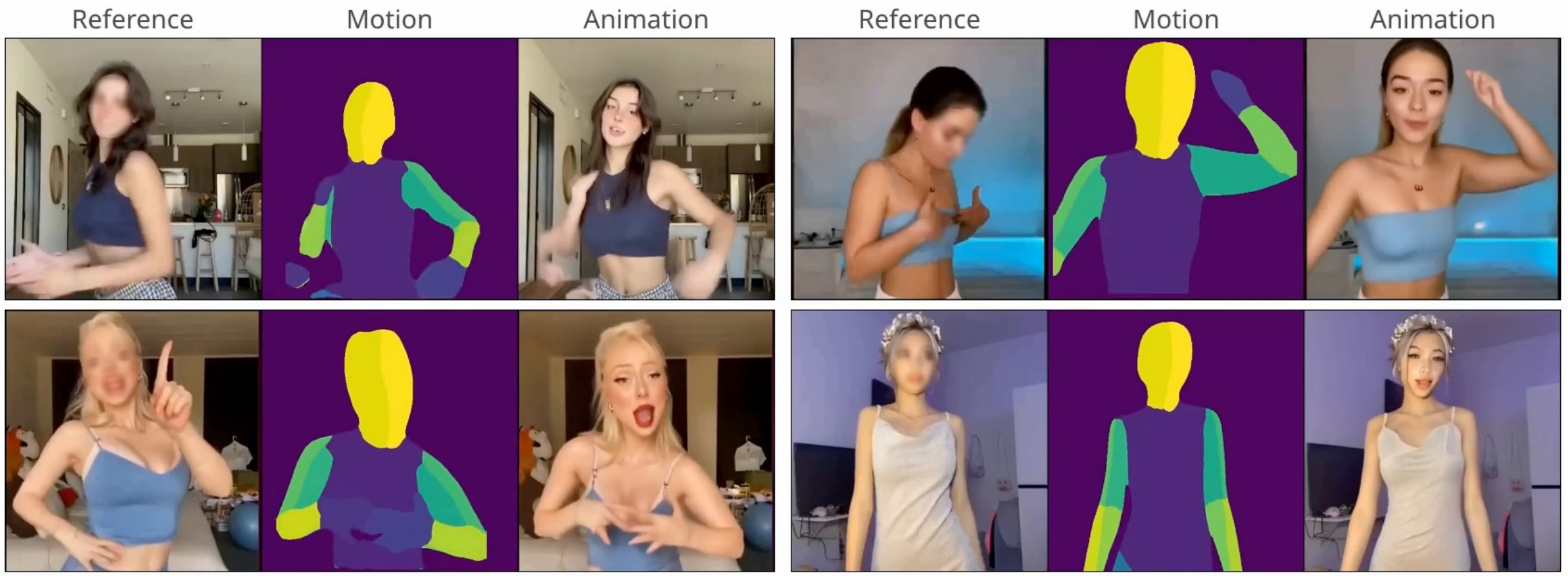
Conclusion

MagicAnimate



Conclusion

MagicAnimate



Conclusion

- Why ControlNet is the best paper at ICCV 2023
- How to fine-tune Stable Diffusion on lab-scale GPUs with small datasets
- The impact of ControlNet on new and fun projects.

Q & A

Appendix

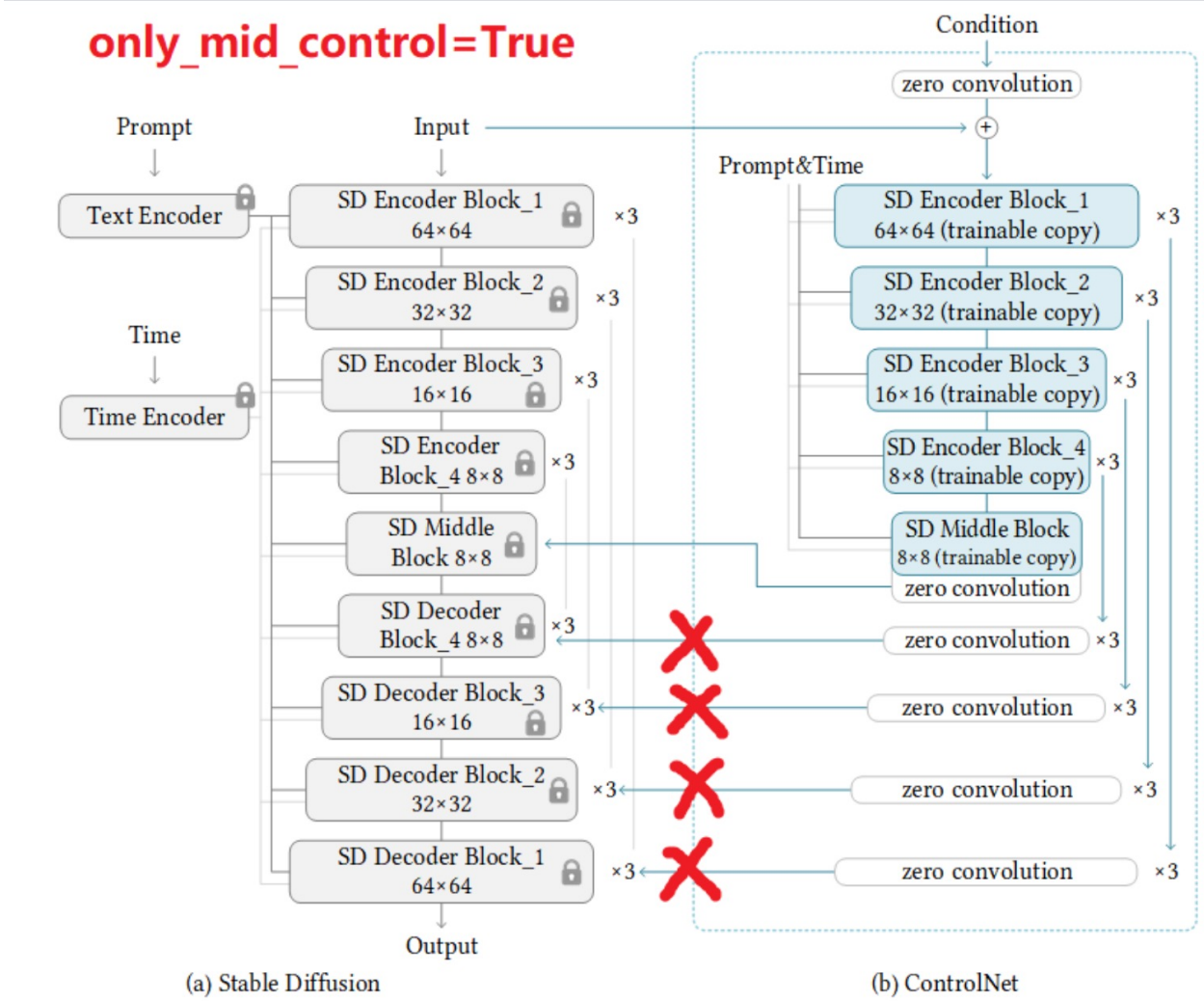
Training technic

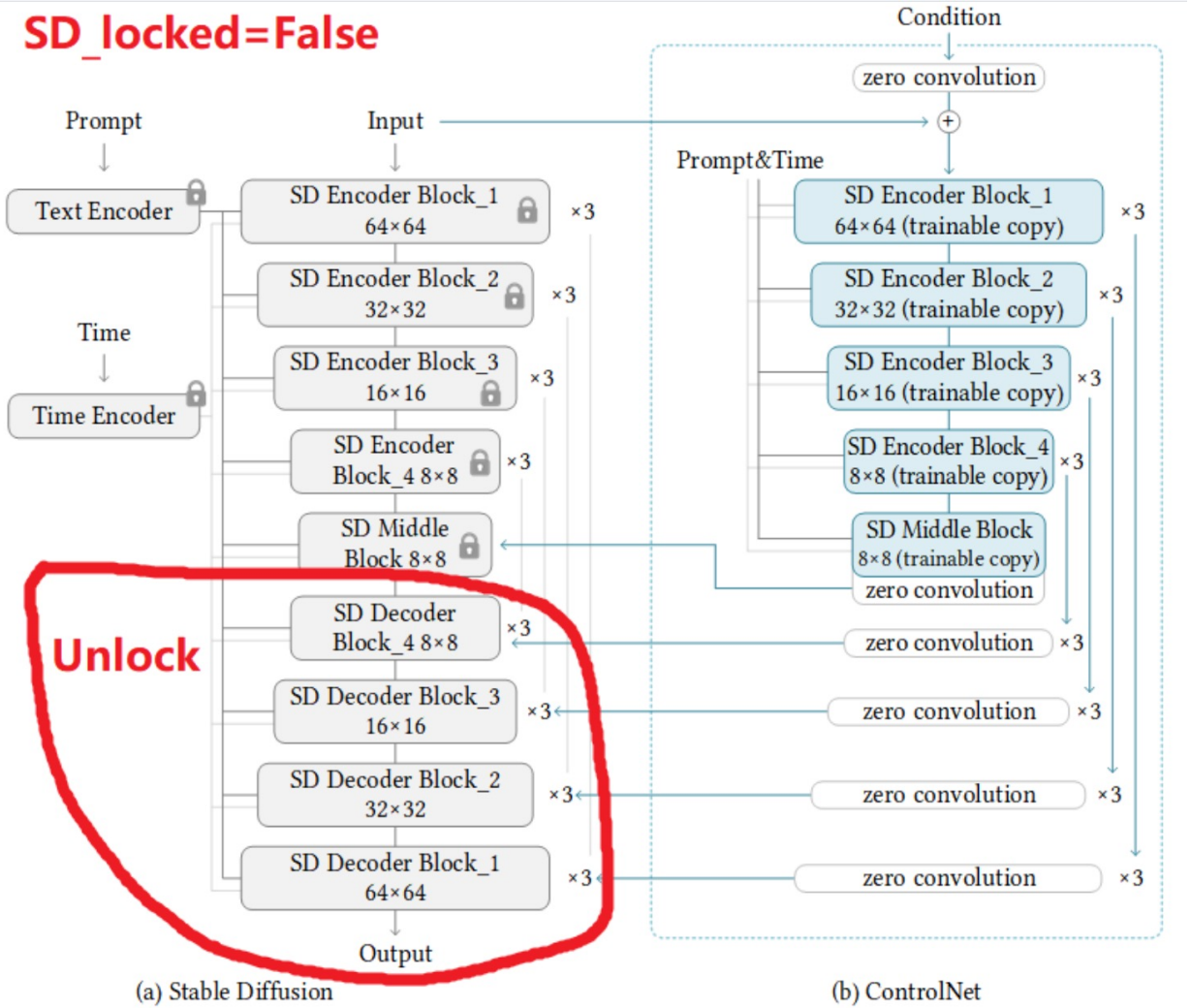
Randomly replace 50% text prompts with empty strings

This approach increases ControlNet's ability to directly recognize semantics in the input conditioning images (e.g., edges, poses, depth, etc.) as a replacement for the prompt.

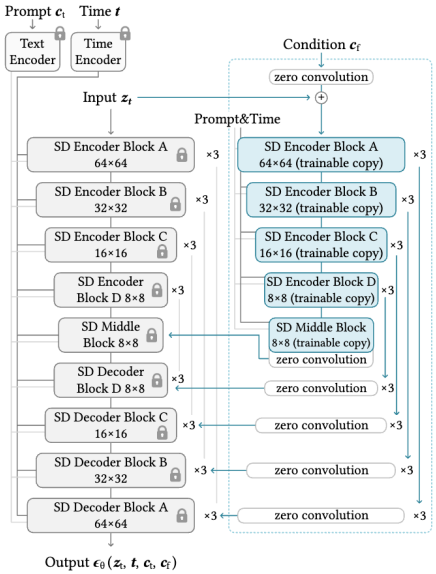
Appendix

Training technic

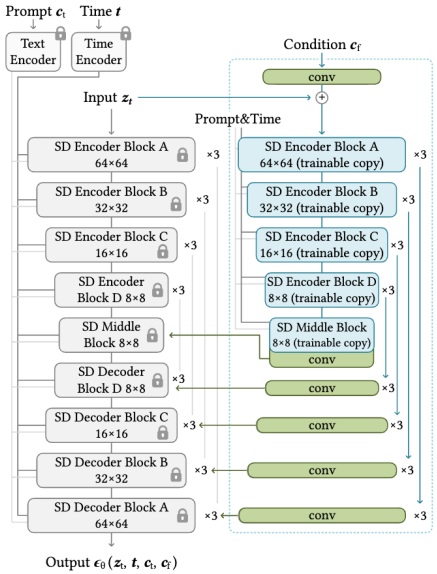




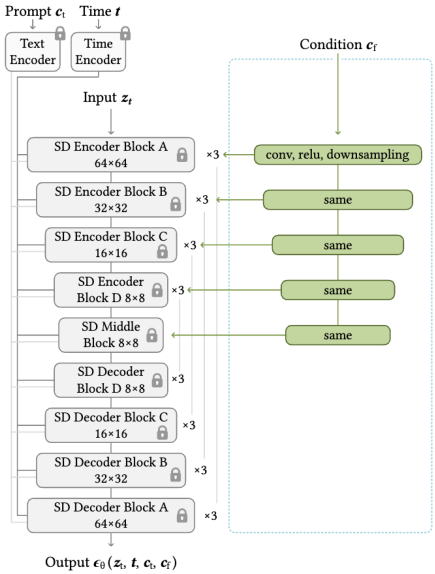
Appendix



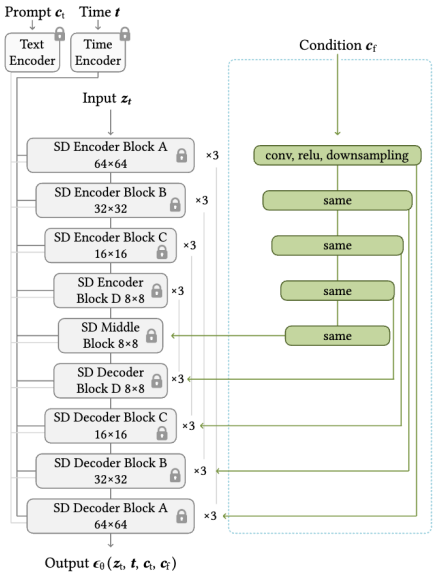
(a) Proposed



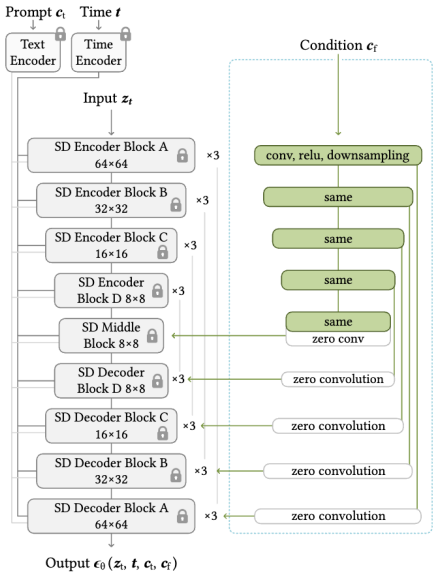
(b) w/o zero convolutions



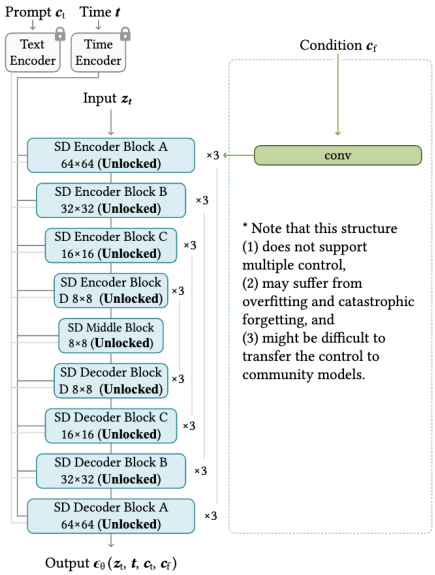
(c) w/o trainable copy, training lightweight layers from scratch (connecting encoder)



(d) w/o trainable copy, training lightweight layers from scratch (connecting decoder)

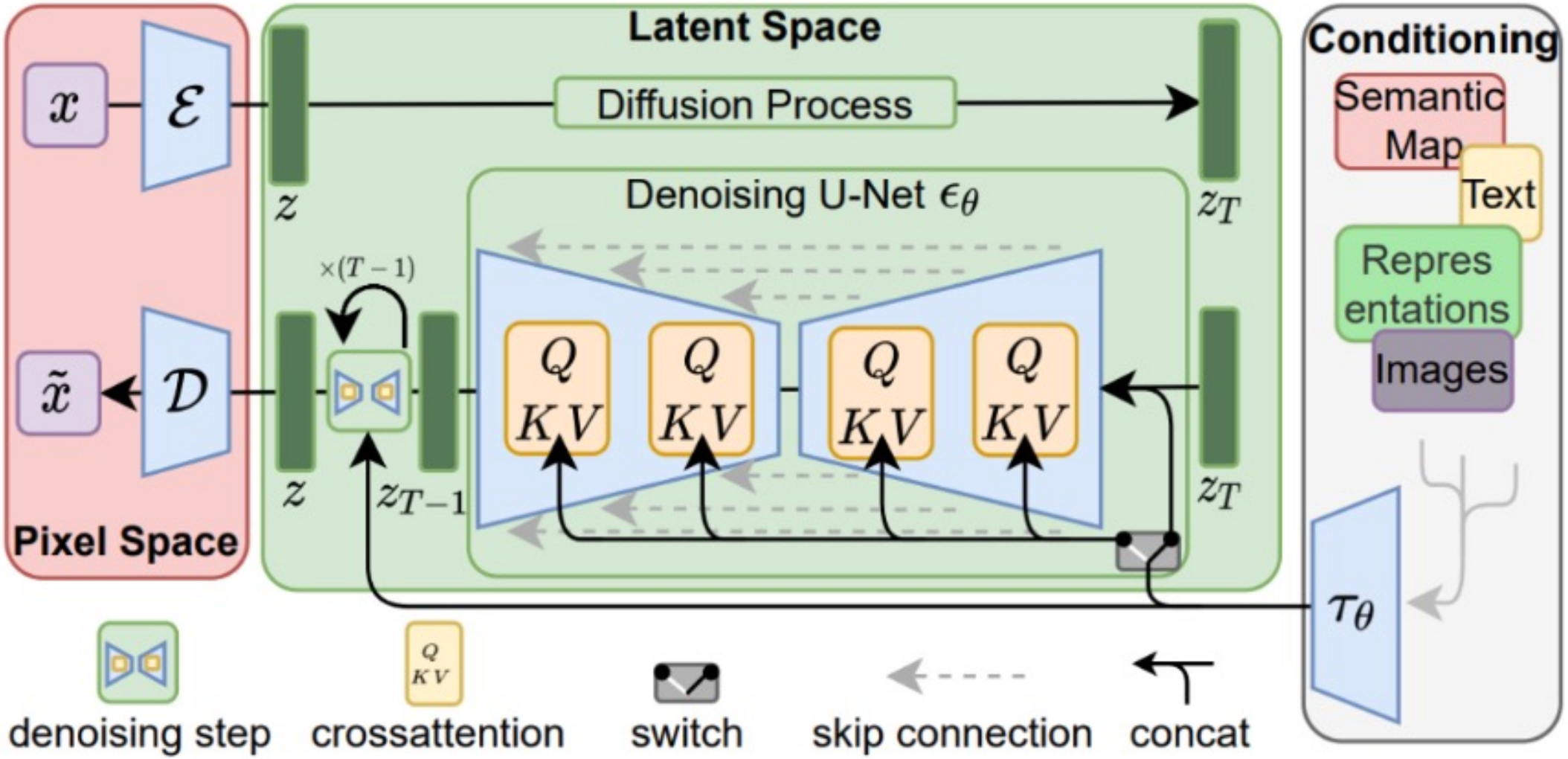


(e) w/o trainable copy, training lightweight layers from scratch (connecting decoder, using zero conv)



(f) directly train original model

Appendix



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

Appendix

Q: If the weight of a conv layer is zero, the gradient will also be zero, and the network will not learn anything. Why "zero convolution" works?

A: This is wrong. Let us consider a very simple

$$y = wx + b$$

and we have

$$\partial y/\partial w = x, \partial y/\partial x = w, \partial y/\partial b = 1$$

and if $w = 0$ and $x \neq 0$, then

$$\partial y/\partial w \neq 0, \partial y/\partial x = 0, \partial y/\partial b \neq 0$$

which means as long as $x \neq 0$, one gradient descent iteration will make w non-zero. Then

$$\partial y/\partial x \neq 0$$

so that the zero convolutions will progressively become a common conv layer with non-zero weights.

Appendix

Zero convolution layer

$$y = wx + b$$

$$\partial y/\partial w = x, \partial y/\partial x = w, \partial y/\partial b = 1$$

if $w = 0$ and $x \neq 0$, then

$$\partial y/\partial w \neq 0, \partial y/\partial x = 0, \partial y/\partial b \neq 0$$

$$\partial y/\partial x \neq 0$$

$\boldsymbol{I} \in \mathbb{R}^{h \times w \times c}$, the forward pass can be written as

$$\mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i} = \boldsymbol{B}_i + \sum_j^c \boldsymbol{I}_{p,j} \boldsymbol{W}_{i,j}. \tag{1}$$

Since a zero convolution layer in initialized with $\boldsymbol{W} = \boldsymbol{0}$ and $\boldsymbol{B} = \boldsymbol{0}$ (*i.e.*, before any optimization steps), anywhere that $\boldsymbol{I}_{p,i} \neq \boldsymbol{0}$ the gradients become

$$\begin{cases} \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{B}_i} = 1, \\ \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{I}_{p,i}} = \displaystyle\sum_j^c \boldsymbol{W}_{i,j} = 0, \\ \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{W}_{i,j}} = \boldsymbol{I}_{p,j} \neq \boldsymbol{0}. \end{cases} \tag{2}$$

We see that although a zero convolution can cause the gradient on the feature term $\boldsymbol{I}$ to become zero, the gradients for the weight and bias are not influenced. As long as the feature $\boldsymbol{I}$ is non-zero, the weight $\boldsymbol{W}$ will be optimized into a non-zero matrix in the first gradient descent iteration. Notably, in our case, the feature term is input data or condition vectors sampled from datasets, which naturally ensures non-zero $\boldsymbol{I}$.

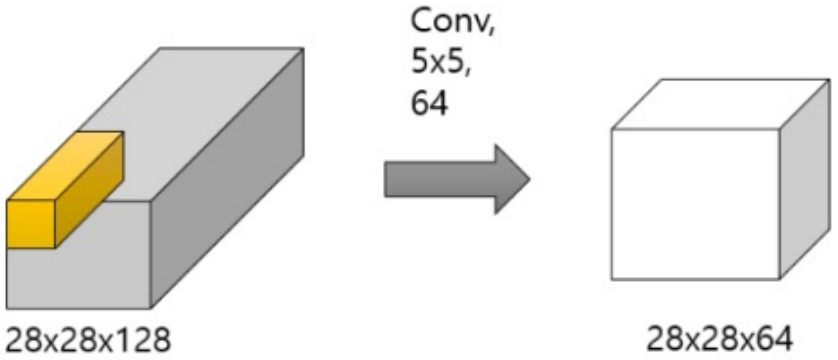
For example, consider classic gradient descent with an overall loss function $\mathcal{L}$ and a learning rate $\beta_{\text{lr}} \neq 0$, if the "outside" gradient $\partial \mathcal{L}/\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})$ is not zero, we have

$$\boldsymbol{W}^* = \boldsymbol{W} - \beta_{\text{lr}} \cdot \frac{\partial \mathcal{L}}{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})} \odot \frac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})}{\partial \boldsymbol{W}} \neq \boldsymbol{0}, \tag{3}$$

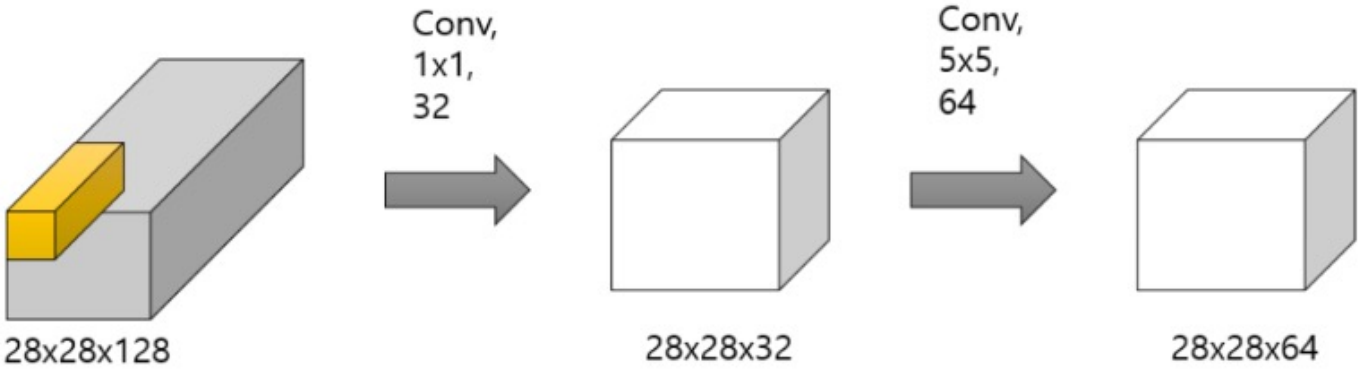
where $\boldsymbol{W}^*$ is the weight after one gradient descent step and $\odot$ is Hadamard product. After this step, we have

$$\frac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}^*, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{I}_{p,j}} = \sum_j^c \boldsymbol{W}^*_{i,j} \neq \boldsymbol{0}, \tag{4}$$

Appendix



#params = 28x28x64x5x5x128 = 160M



#params = 28x28x32x128x1x1
= 4.8M

#params = 28x28x64x5x5x32
= 40M

#total = 44.8M