

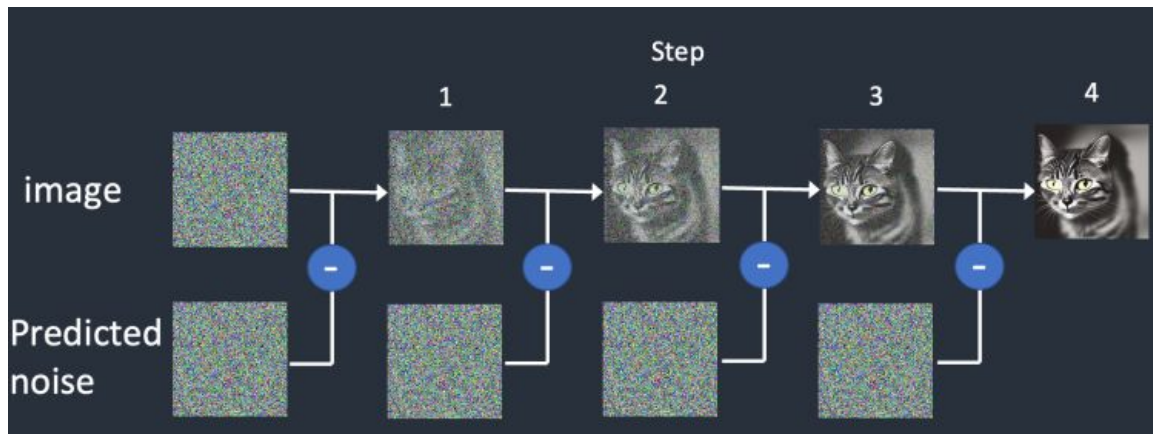
# Image Generative Model

**Stable Diffusion**

# What are Diffusion Models?

— — —

- It is generative deep learning model using noise reduction method
- It reverses the process of adding noise to an image
- Usually use for text-to-image, but also img-to-img and inpainting
- More stable than GAN (no mode collapse)



**Figure 1** The Process of Noise Reduction

# Stable diffusion

— — —

- Text-to-Image (Image-to-Image in some pipeline)
- Latent diffusion model + Text encoder
- Open access
- Trained on  $512 \times 512$  images from a subset of the LAION-5B database
- Uses a frozen CLIP ViT-L/14 text encoder

# Outline

— — —

- How to train & generate image
- Finetuning Techniques
- Code Demo

# How to Train & Generate Image

# Stable Diffusion Model Comprises

— — —

- 1) Tokenizer → tokenize prompt
- 2) Text Encoder → encode tokens
- 3) Variational Auto-Encoder (vae) → map image to latent space
- 4) Noise Scheduler → generate noise
- 5) UNET → predict noise

## “Training”

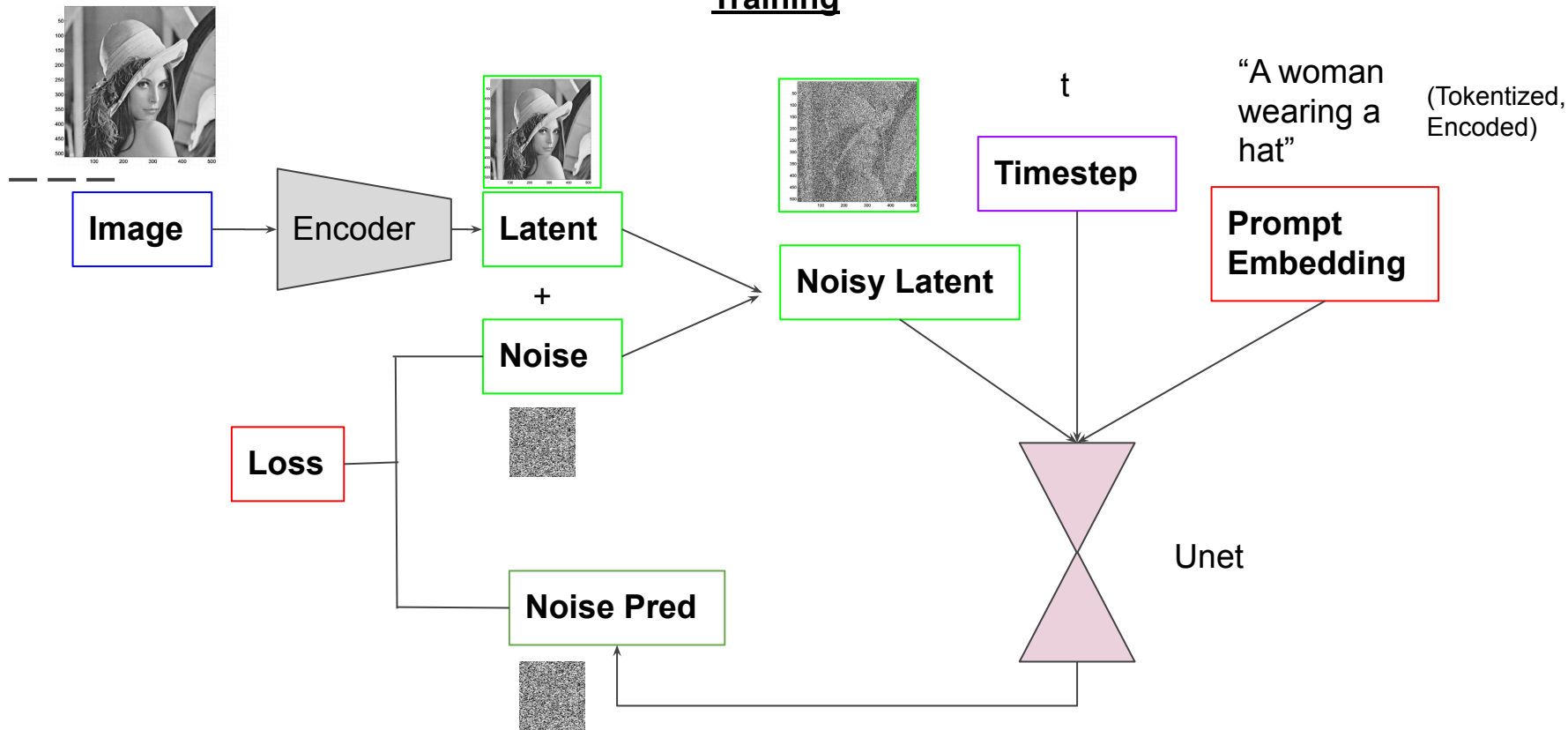
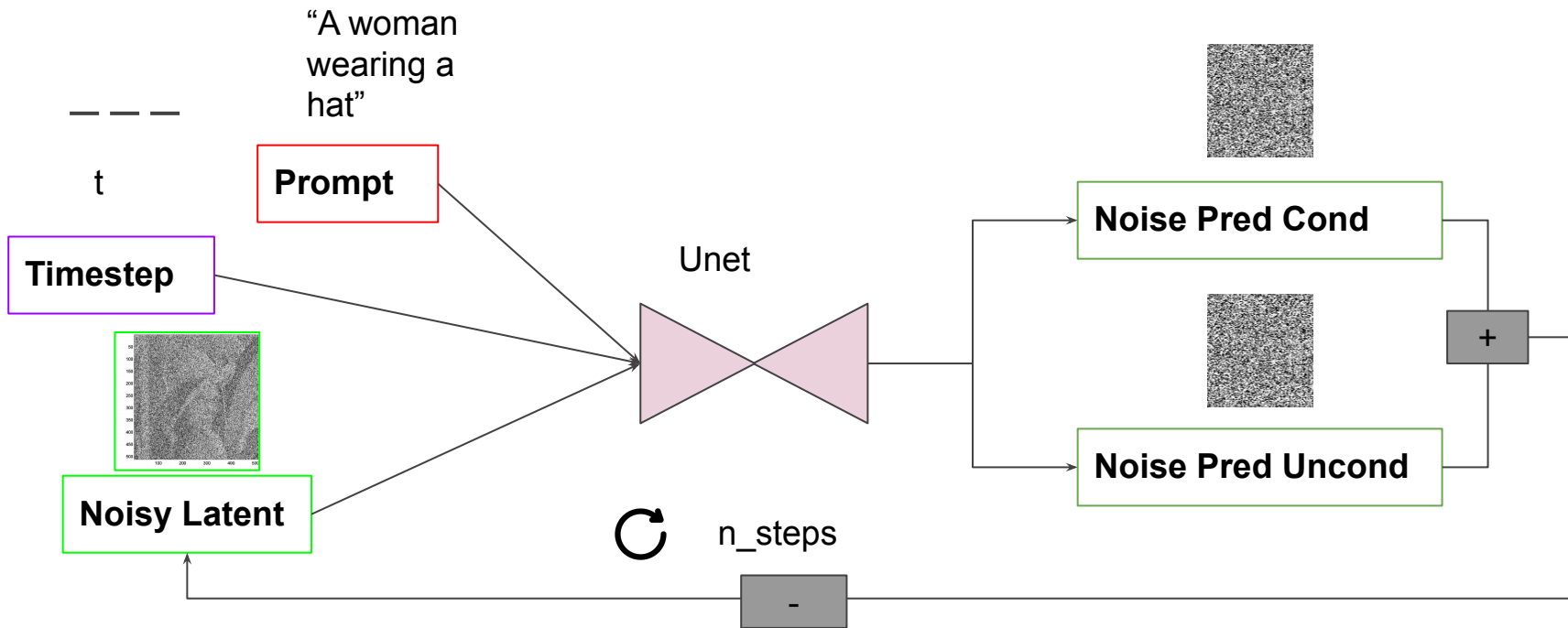


Figure 2 Diffusion During Training

## “Image Generation”



**Figure 3** Diffusion Model During Inference



# Finetuning Techniques

# Finetuning Techniques

---

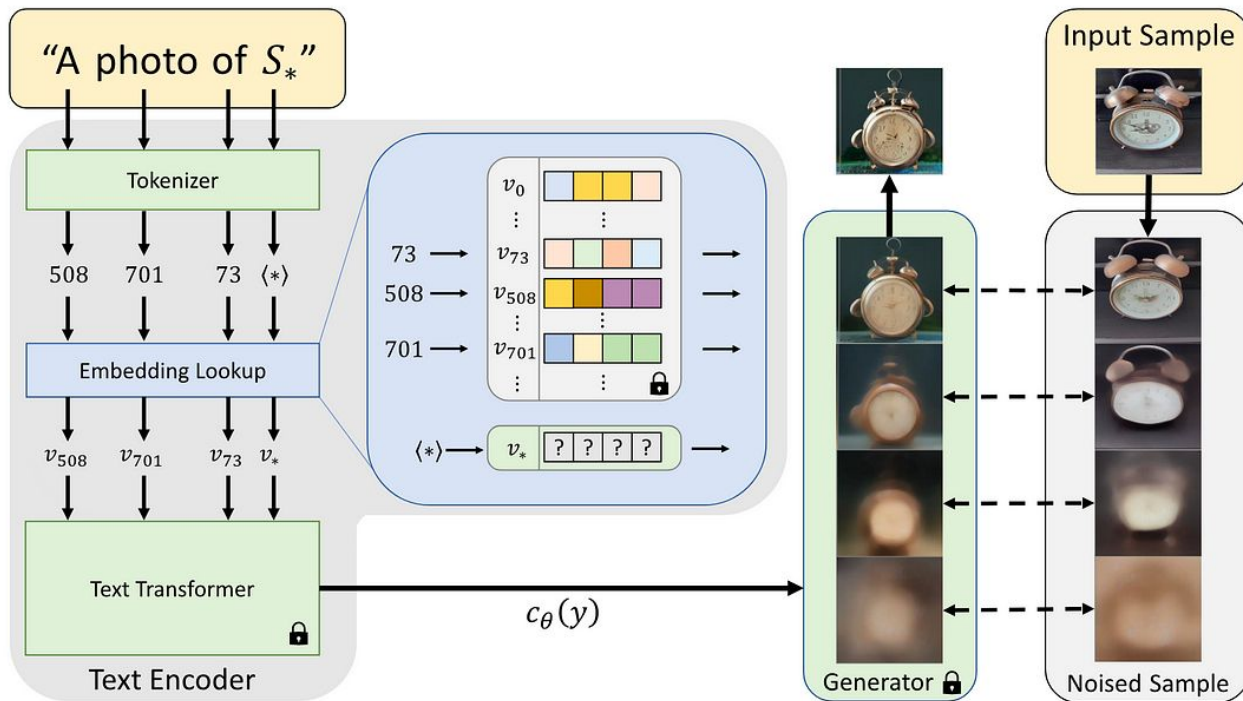
- Textual Inversion
- DreamBooth
- ControlNet

# Textual Inversion (paper)

— — —

- Use a small set of images (typically 3-5)
- The image depicts our target concept across multiple settings
- e.g. varied background or poses
- “We intervene in the embedding process and replace the vector associated with the tokenized string with a new, learned embedding”
- “In essence “injecting” the concept in to our vocabulary”
- Cons of Textual Inversion
  - This method only trains text encoder (Tokenizer)
  - It’s good when you want to give your “concept” a new style

# Textual Inversion (paper)



**Figure 4** Outline for text embeddings and inversion process

# DreamBooth (paper)

— — —

- trains unet (and text encoder if you want)
- only need 3-5 image per subject
- May overfit to training data
- Some subjects are easier to learn than others

# DreamBooth (paper)

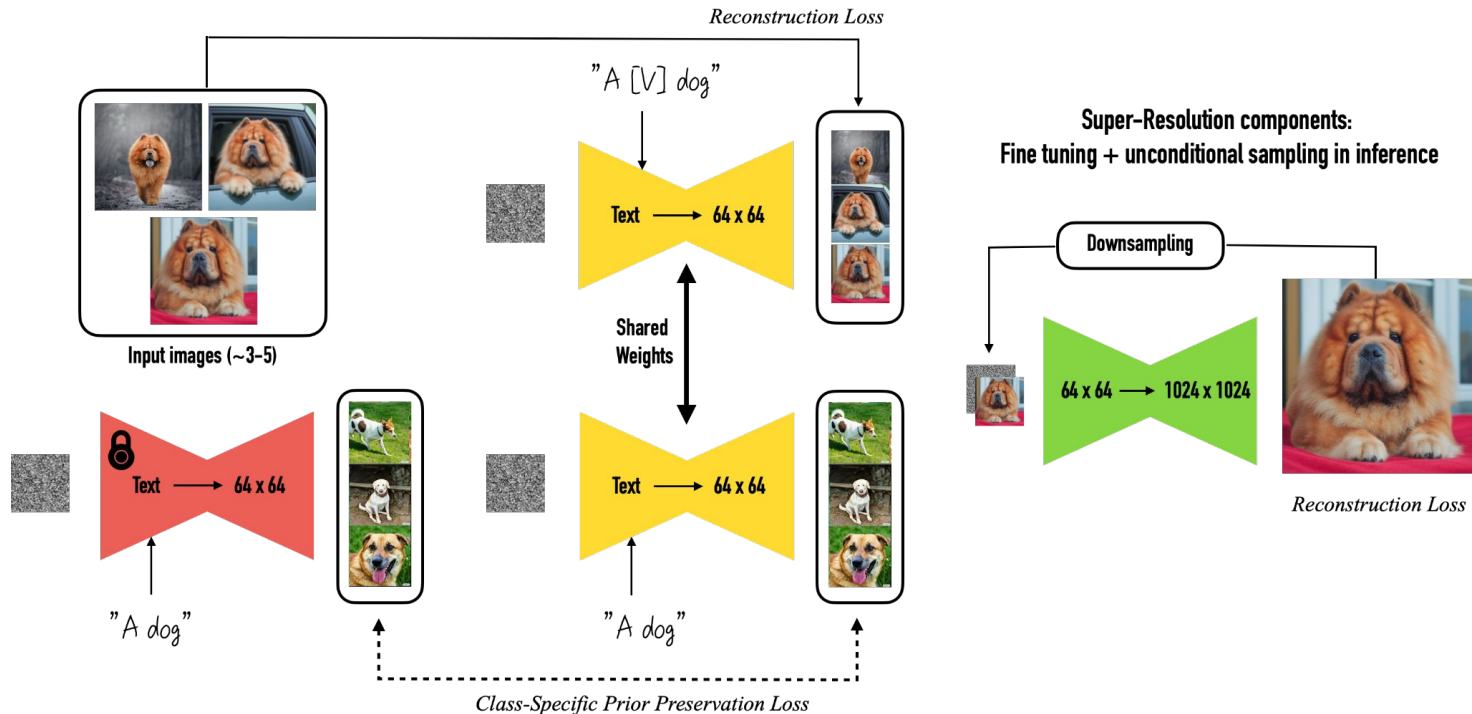


Figure 5 Architecture overview from the DreamBooth

# ControlNet (paper)

— — —

- ControlNet is model that **add conditioning controls** pretrained diffusion model
- Condition input to guide the content of the generated image usually is edge maps, pose key points, depth maps, segmentation maps, normal maps, etc



Canny edge (input)

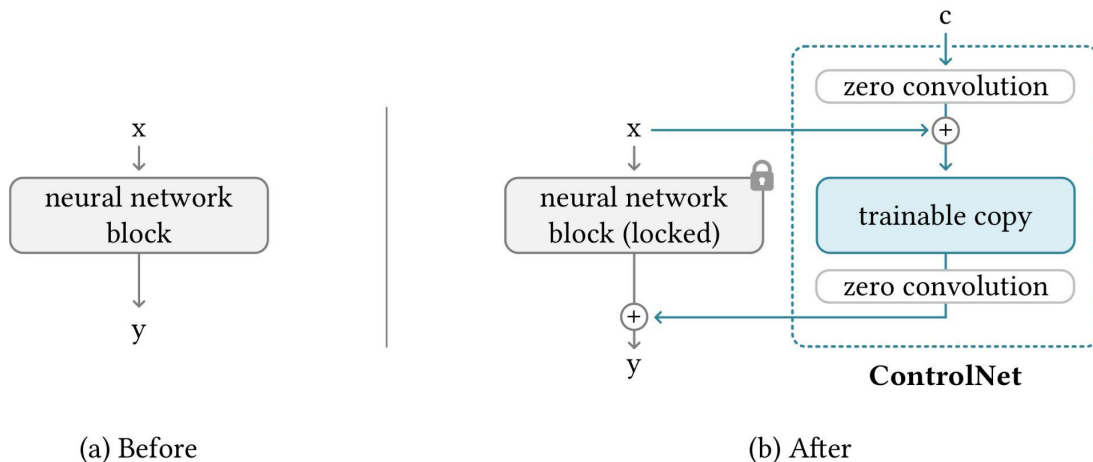


Generated images (output)

**Figure 6:** Control Stable Diffusion with Canny edge map

# ControlNet Architecture

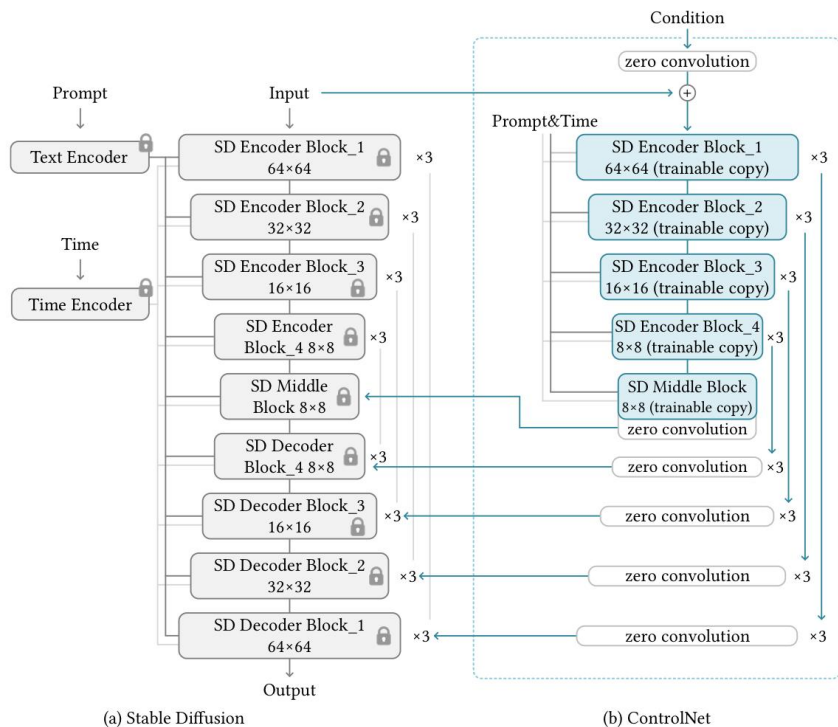
- ControlNet architecture is based on copying the structure and weights from the decoder part of the U-net architecture.
- The “zero convolution” is an  $1 \times 1$  convolution layer with both weight and bias initialized as zeros.



**Figure 7:** 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  $1 \times 1$  convolution layer with both weight and bias initialized as zeros.



# Training a ControlNet for Diffusion



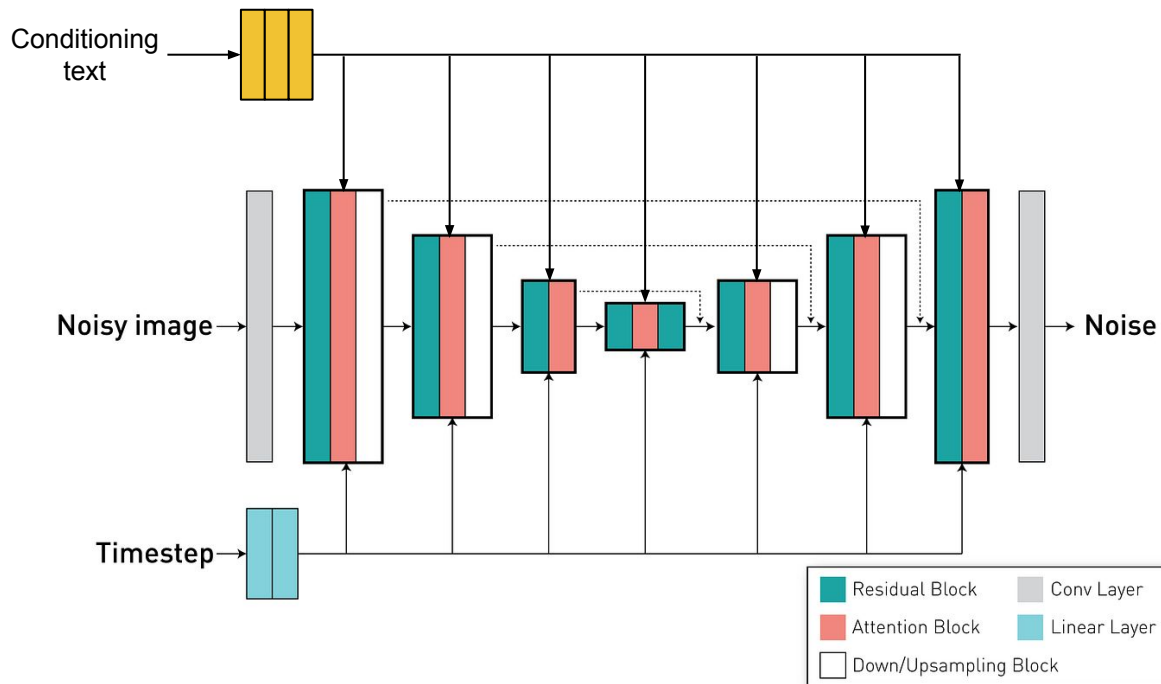
- In the training step, all parameters in the Diffusion part are locked, but only the ControlNet part is trained.

**Figure 8:** ControlNet in Stable Diffusion.

# Code Demo

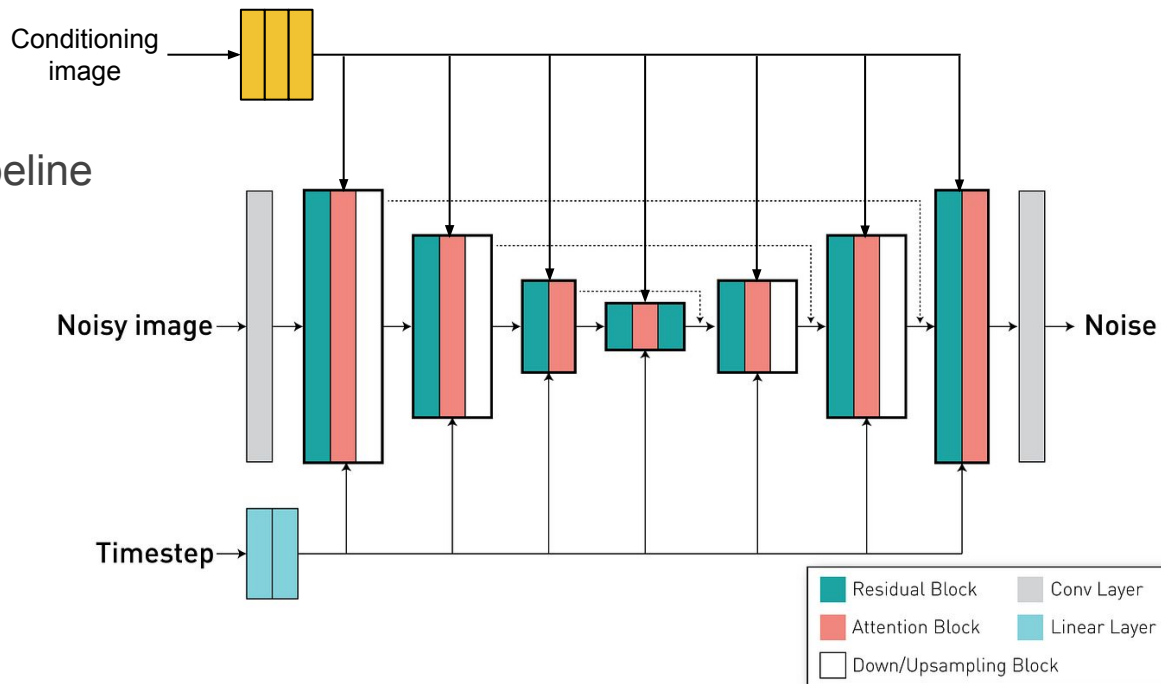
# Text-to-Image

## StableDiffusionPipeline



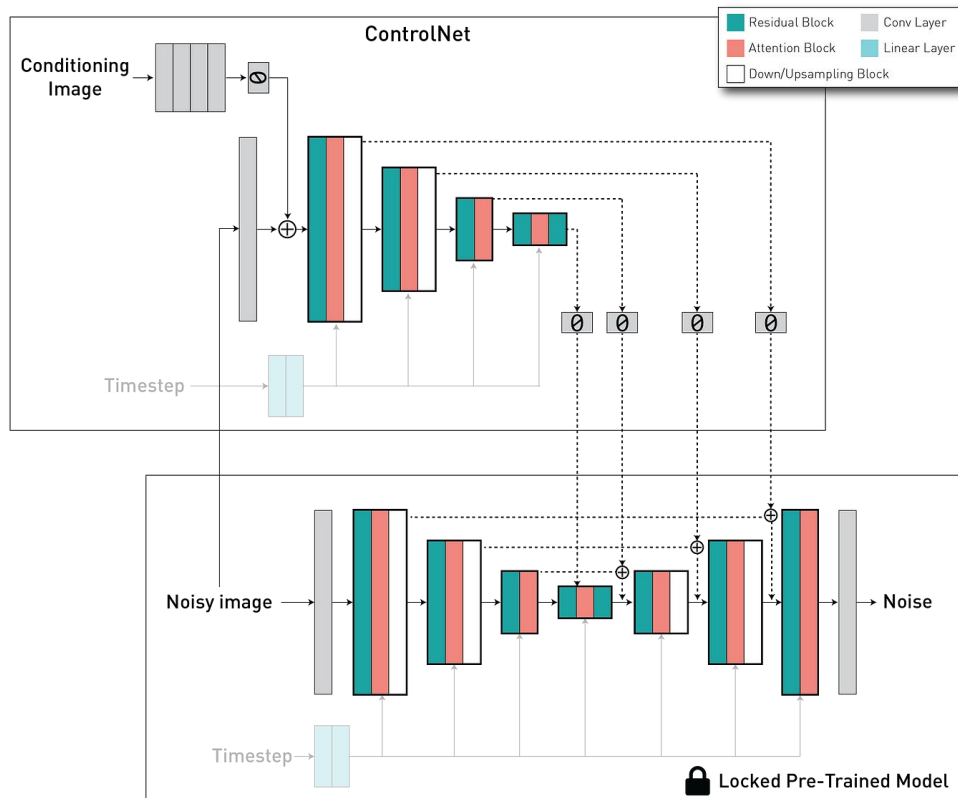
# Image-to-Image

## StableDiffusionImg2ImgPipeline



# Image-to-Image(ControlNet)

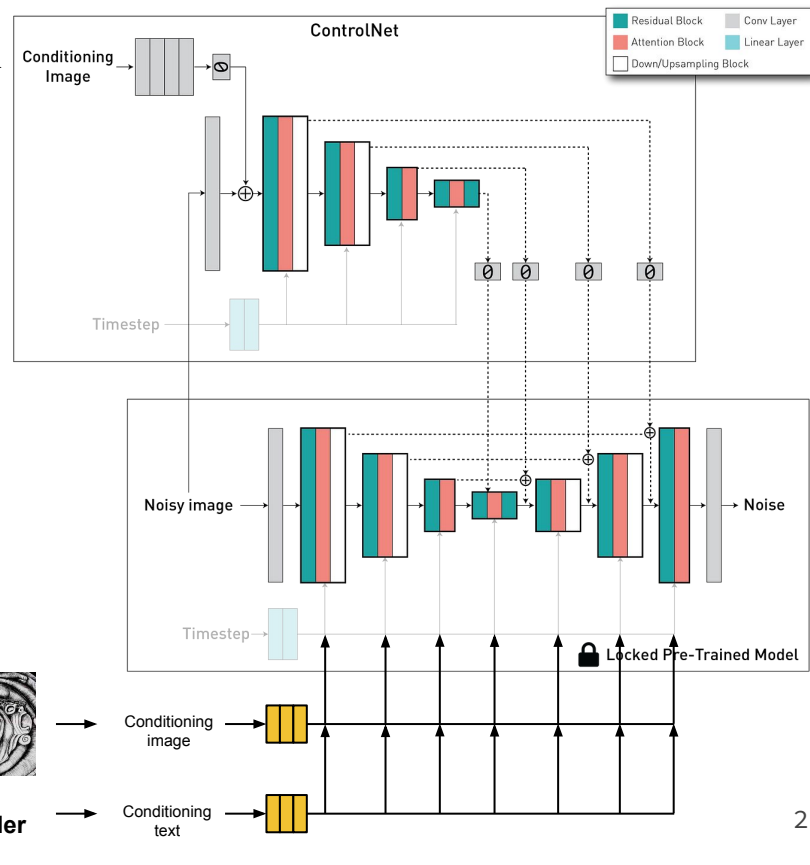
StableDiffusionControlNetPipeline



# Image-to-Image(ControlNet)



StableDiffusionControlNetImg2ImgPipeline



# Finetuning Techniques

— — —

- Textual Inversion
- DreamBooth