Latent diffusion models, stable diffusion, sdxl, and control nets

Justin Barry

November 28, 2023

Intro

Main idea: Using diffusion models (DM) to generate an image given a text describing the image, dubbed "text-to-image".

Problem

Diffusion Models (DMs) for Image Synthesis

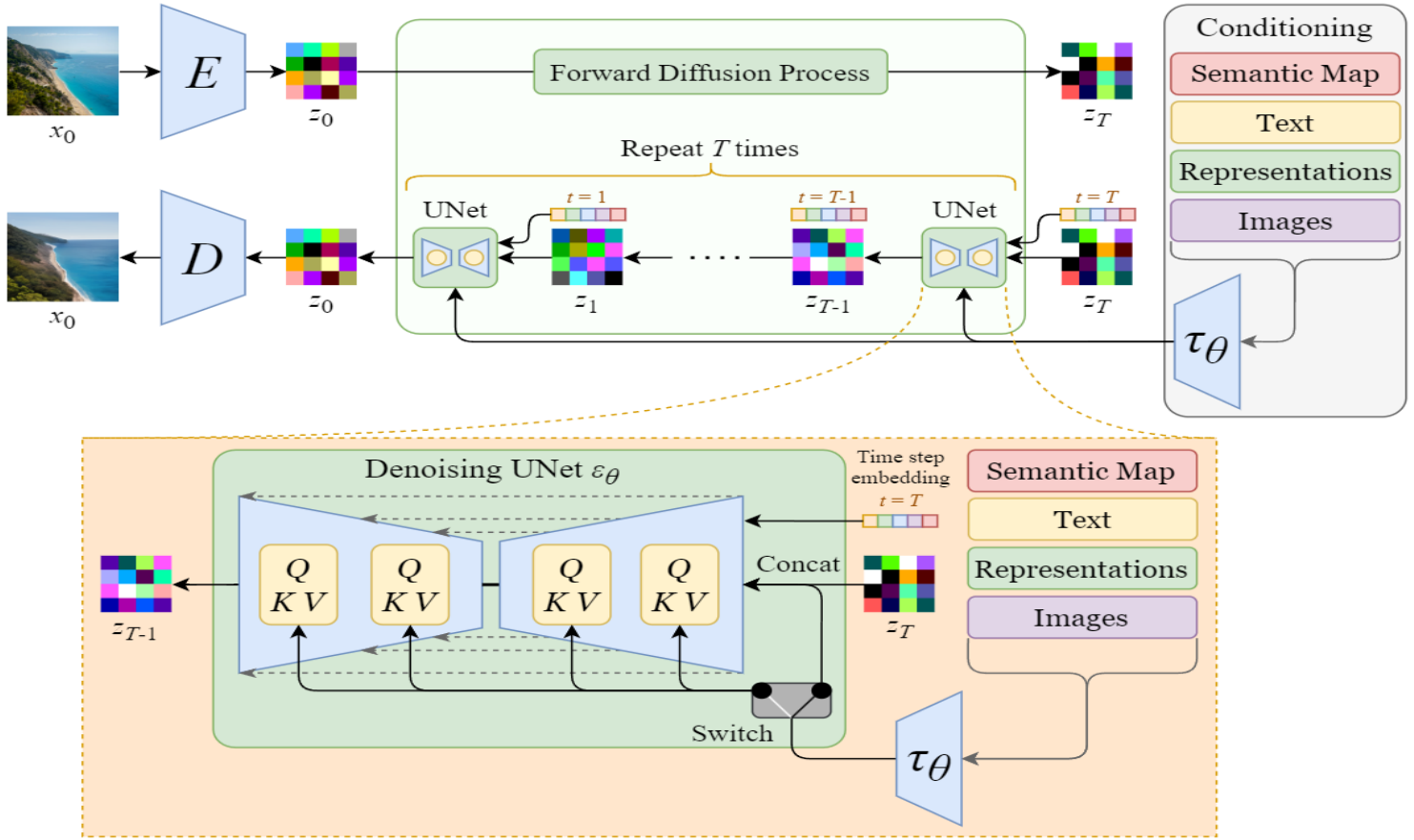
- DMs aim to learn an image distribution by gradually denoising a distribution of noisy images.

Novelties

Novel additions [6]

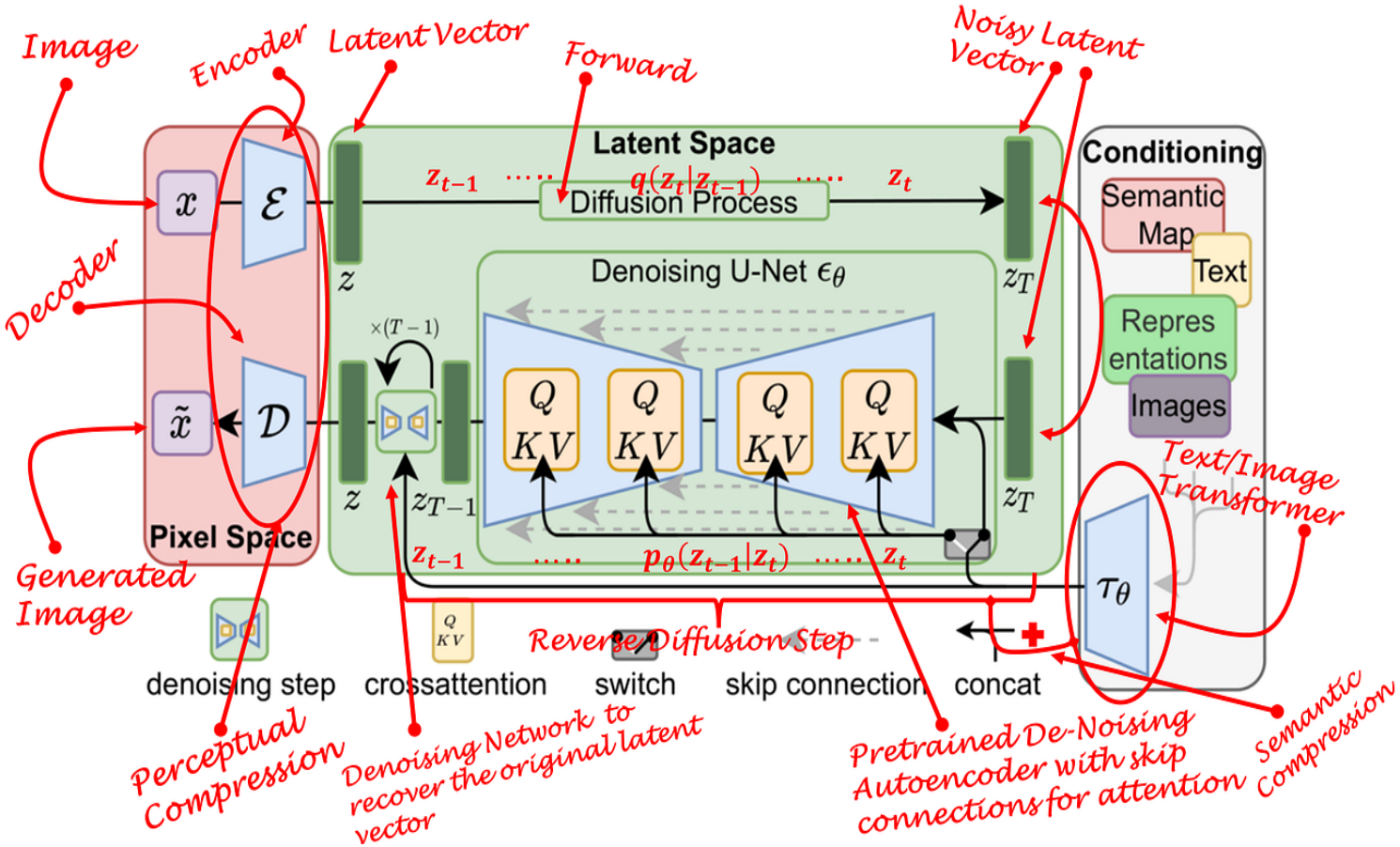
- Conditioning on text via cross-attention mechanism.
- Performing denoising objective in a compressed latent space as opposed to the original pixel space.
- Model that transforms conditioning inputs and yields a representation for both images and texts.
- 1.45B parameters

To the point



[7]

Model annotated



[7]

Computational complexity

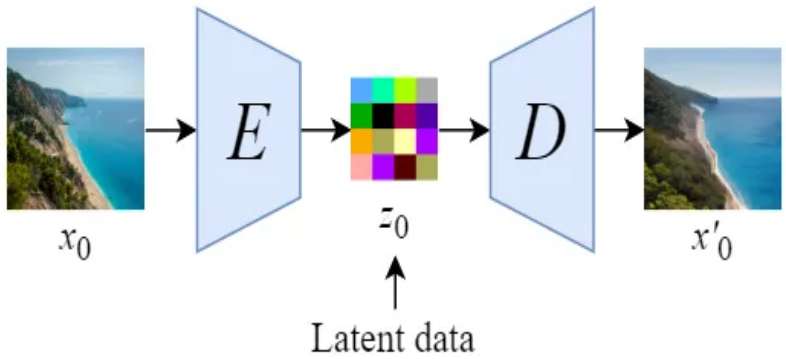
- Problem: high complexity computing the diffusion model over the original images (pixel space).
- Solution: The authors' approach is to apply DMs in the latent space yielded by autoencoders.
- Two components to stable diffusion:
  - The autoencoder compresses images to the latent space.
  - U-net to denoise latent images (our diffusion model).

Perceptual compression model

- Avoid function evaluation in the pixel space by leveraging image compression.
- A reconstructed image

$$\tilde{x} = \mathcal{D}(z) = \mathcal{D}(\mathcal{E}(x))$$

- $\mathcal{E}$ encoder
- $\mathcal{D}$ decoder
- x original image
- z latent (compressed) image



[7]

Perceptual compression loss

Autoencoder objective

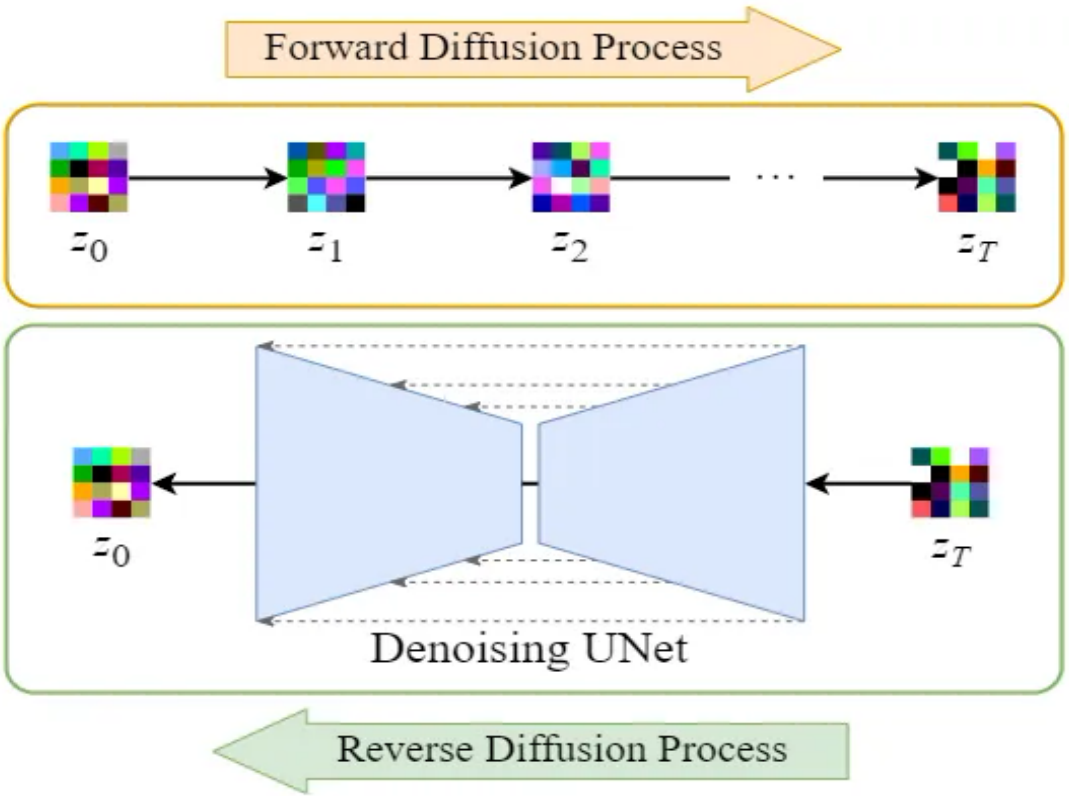
$$L_{autoencoder} = \min_{\mathcal{E},\mathcal{D}} \max_{\Psi} (L_{rec}(x, \mathcal{D}(\mathcal{E}(x))) - L_{adv}(\mathcal{D}(\mathcal{E}(x))) + \log \mathcal{D}_{\Psi}(x) + L_{reg}(x;\ \mathcal{E}, \mathcal{D}))$$

$$= L_{reconstruction} + L_{adversarial} + L_{regularization}$$

- $L_{reconstruction}$ computes the loss over the feature space, not the pixel space
- $L_{adversarial}$ Is the total loss from an adversarial game between the autoencoder and a discriminator
- $L_{regularization}$ ensures the latent space and the predictive image distribution don't stray from a normal distribution.

Image generation

- Diffusion: A function that adds noise images gradually.
- Problem: How to generate images.
- Solution: Learn the reverse diffusion process to generate images from noise.



[7]

Abilities & tasks

- Text-to-image
  - Class conditional image synthesis.
  - Text prompt describing the image.
- Image-to-image
  - Semantic synthesis: condition on semantic map.
  - Super resolution: condition on low-resolution image.
  - Inpainting: condition on masked regions of the image.

Notes on training and data

- Training data contains images paired with a condition (image, text) or (image, image)
- LAION-400M language prompts.
- Finetuning on COCO.
- Evaluation data MS-COCO validation set.
- Classifier-free guidance greatly boosts sample quality.

Goal

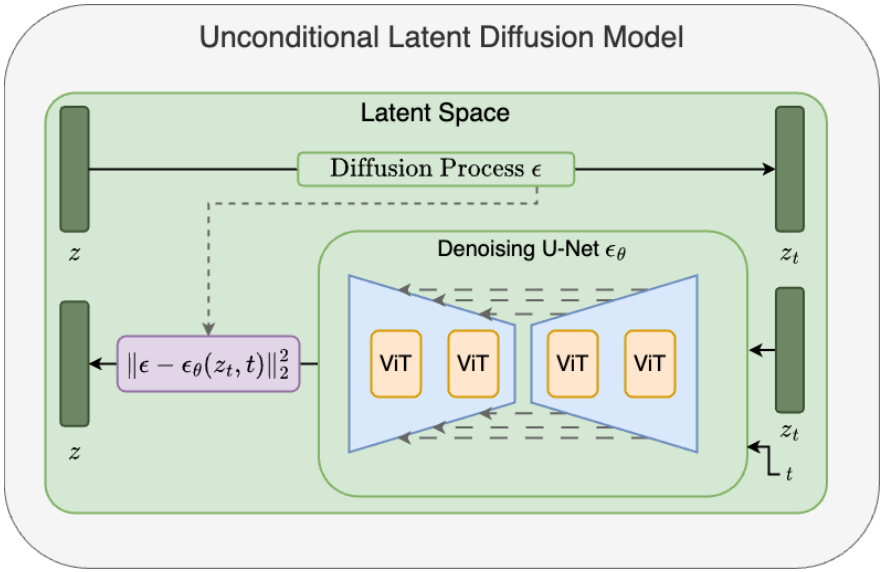
- We want a model that can generate a meaningful image from noise.
- A diffusion function describes how noise is gradually added to an image overtime.
- We can apply the diffusion function in reverse to generate meaningful images from noise.
- The authors leverage a mechanism to condition on text that guides image generation

Training objective

Unconditional training objective learn a latent image distrubution $p(z)$ by gradually denoising the latent image. Find θ that minimizes the following

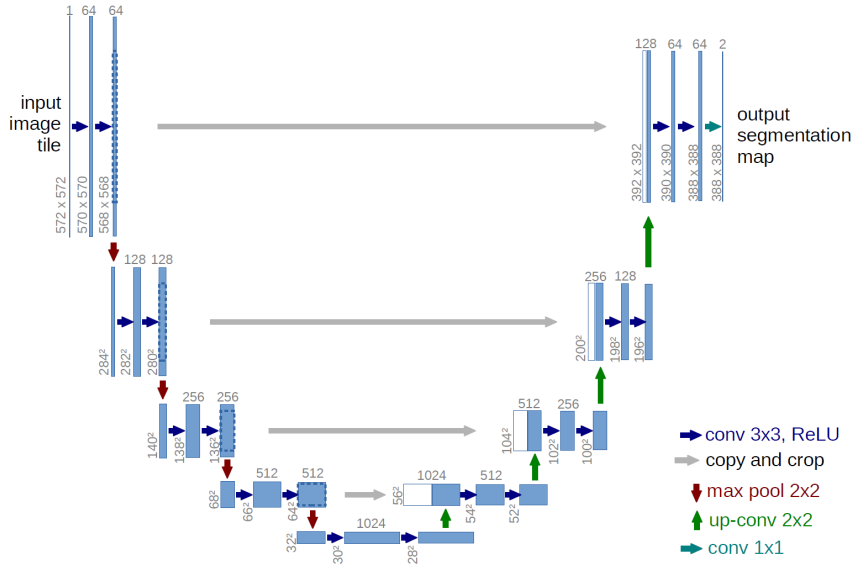
$$L_{LDM} = \mathbb{E}_{\mathcal{E}(x),\epsilon\sim N(0,1)}\|\epsilon - \epsilon_\theta(z_t, t)\|_2^2$$

- ϵ the added noise
- $\epsilon_\theta(z_t, t)$ The predicted noise added to the original image
- z_t is the noised image
- t is a timestep embedding of the same dimensions as z that acts as contextual guidance denoising U-Net model



U-net

U-net is an autoencoder with skip connections from the encoders to the decoders that preserve the loss of spatial information during the encoding.



[8]

Diffusion function

The diffusion function $q(z_t \mid z_0) = N(z_t \mid \alpha_t z_0, \sigma_t^2 I)$ determines the amount of noise we add to images.

- z_0 is the original latent image.
- z_t is the noised latent image at timestep t.
- α_t is the scaling factor at timestep t.
- σ_t^2 represents the variance of the noise at timestep t.
- N denotes a normal distribution.
- I represents the identity matrix, indicating that the noise is isotropic

Diffusion function (continued)

Noise schedule

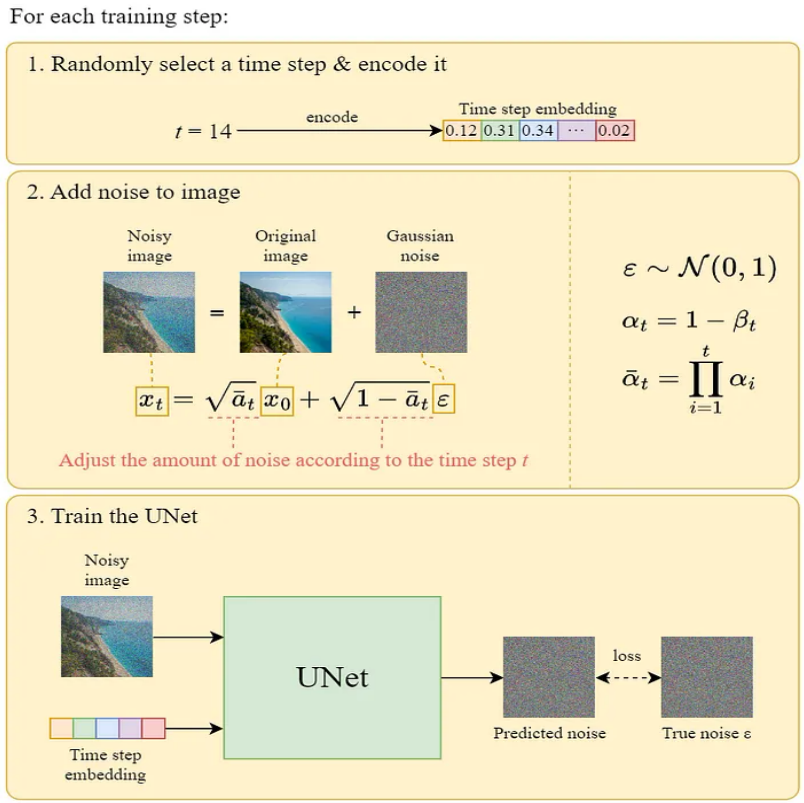
- Training iterations do NOT correspond to timesteps t
- Total number of timesteps $|T|$ and scaling factors α_t are preset hyperparamters.
- Each t will have a corresponding scaling factor α
- The noise "progression" is determined by randomly assigning t values to images at the beginning of each training iteration.

Learning the diffusion function

To learn the diffusion function (note about time step embeddings for learning reverse)

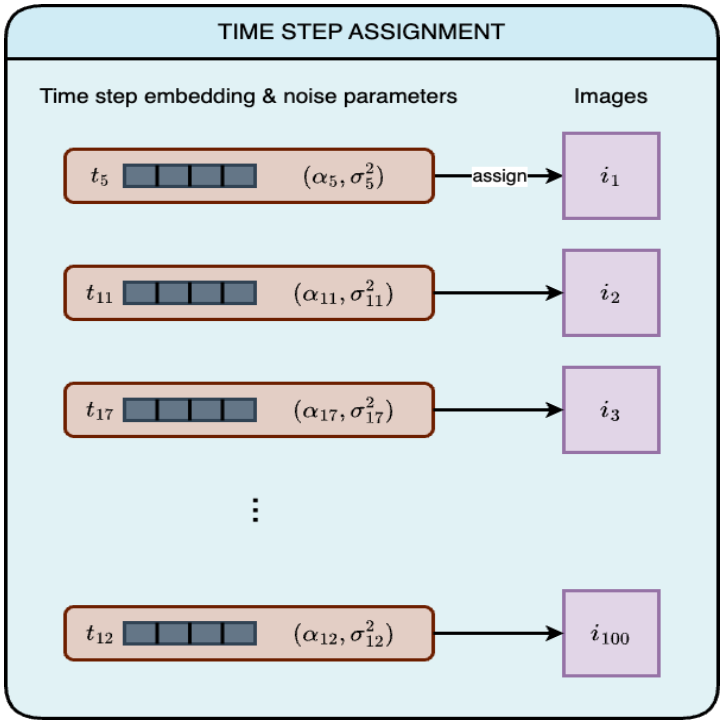
- At the beginning of each iteration, randomly assign a time step t to latent image z_0
- Add noise to the compressed image according to the schedule: $z_t = \alpha_t z_0 + \sigma_t^2 \epsilon$
- Predict the noise added to the latent image using our model $\epsilon_\theta(z_t, t)$
- Calculate and backpropagate the loss given by $\|\epsilon - \epsilon_\theta(z_t, t)\|$

[7]



Learning the diffusion function (continued)

- The model learns the diffusion function by randomly distributing the diffusion schedule across images.
- This is done by randomly assigning t's to z's (images).
- Computationally efficient: the model doesn't have to see each image at all stages of diffusion.

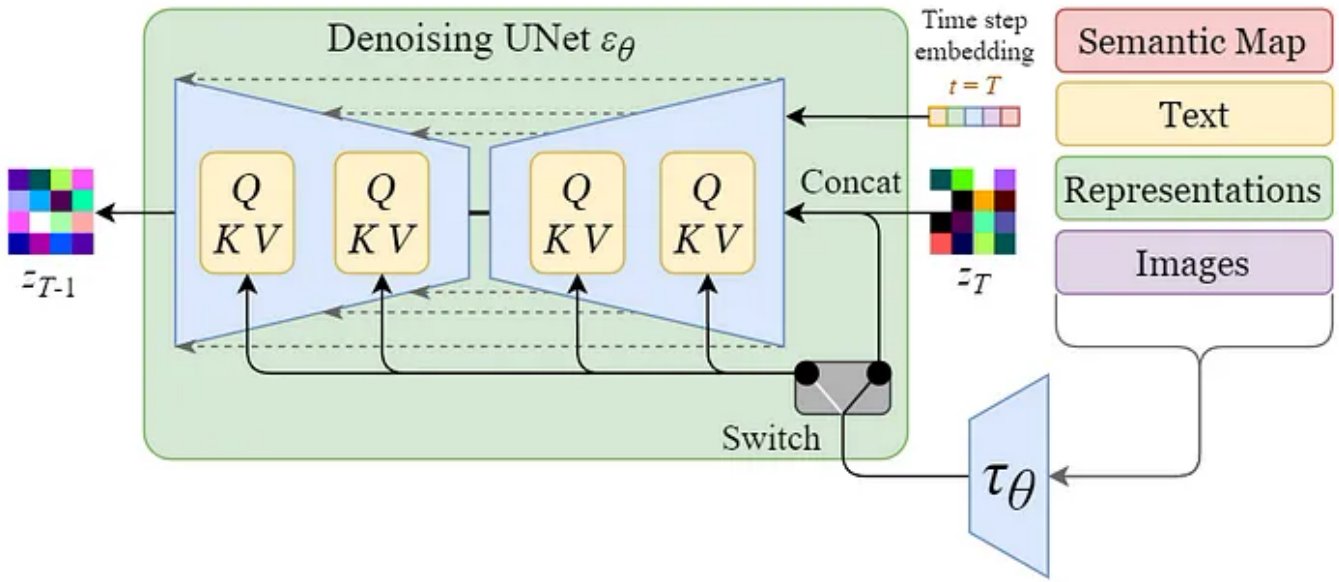


Conditioning the model

Stable diffusion can condition on both text and images.

- Conditional denoising autoencoder $\epsilon_\theta(z_t, t, y)$, where y is our condition (text or image).
- Transform y using CLIP to get an embedding: $\tau_\theta(y)$
- Reparameterize to get

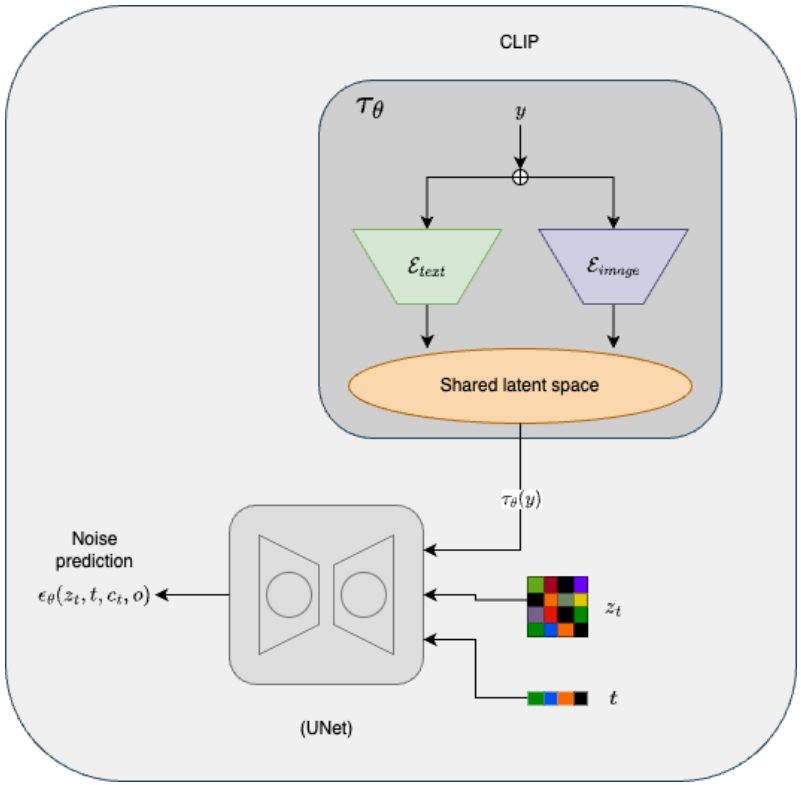
$$L_{LDM} = \mathbb{E}_{\mathcal{E}(x),\epsilon\sim N(0,1)}\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2$$



[7]

Text transformer

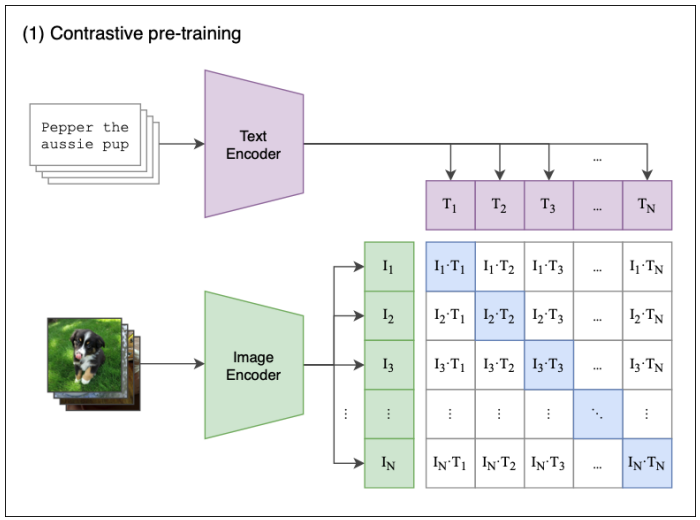
- SD leverages CLIP to implement τ_θ.
- CLIP uses a text encoder and an image encoder to learn a similarity score in a shared embeddings space.



CLIP

How does CLIP work?

- CLIP maximizes the normalized dot product of images and their text descriptions (labels).
- The image and text encoders are both implemented by a transformer.
- Objective: maximize the the dot product between image and text embeddings.



[4]

CLIP in SD

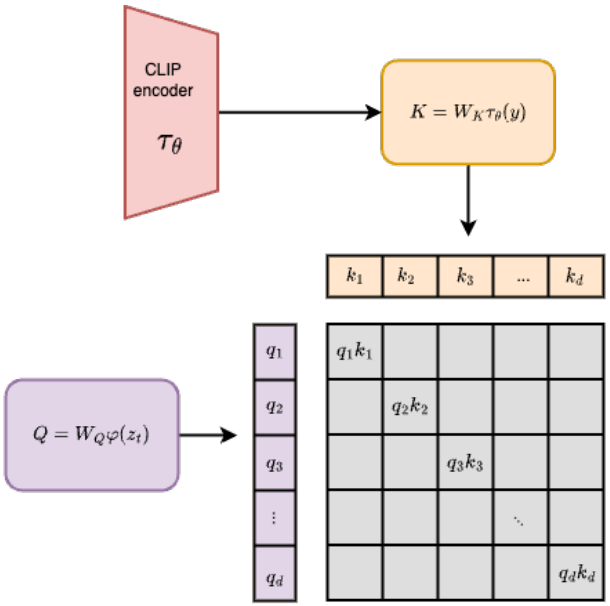
- $\tau_\theta(y)$ is a representation an image or text that is jointly learned in a shared embeddings space.

CLIP drawback

- From AssemblyAI "In particular, many users of Stable Diffusion 2 have claimed that it cannot represent celebrities or artistic styles as well as Stable Diffusion 1, despite the fact that the training data for Stable Diffusion 2 was not deliberately filtered to remove artists. This discrepancy stems from the fact that CLIP's training data had more celebrities and artists than the LAION dataset. Since CLIP's dataset is not available to the public, it is not possible to recover this same functionality using the LAION dataset alone. In other words, many of the canonical prompting methods for Stable Diffusion 1 are all but obsolete for Stable Diffusion 2." [1]

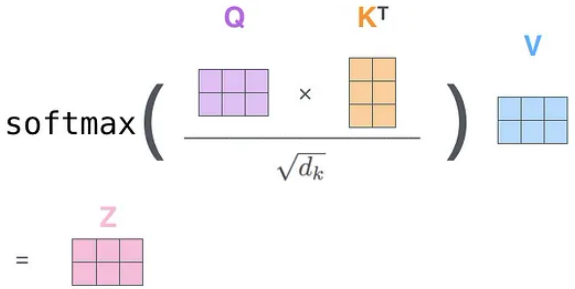
Cross attention in Stable Diffusion

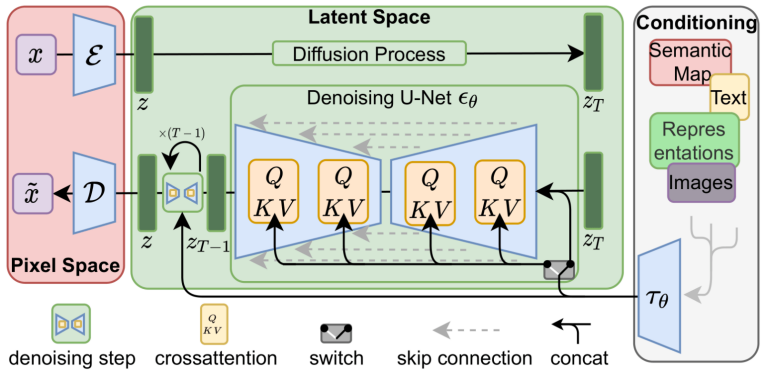
- The output of the text transformer $\tau_\theta(y)$ is passed to each block in U-net.
- $\tau_\theta(y)$ is the key K in the cross attention operation for each block it's passed to.



Each index of K and Q are vectors:

$k_i = q_i =$ 





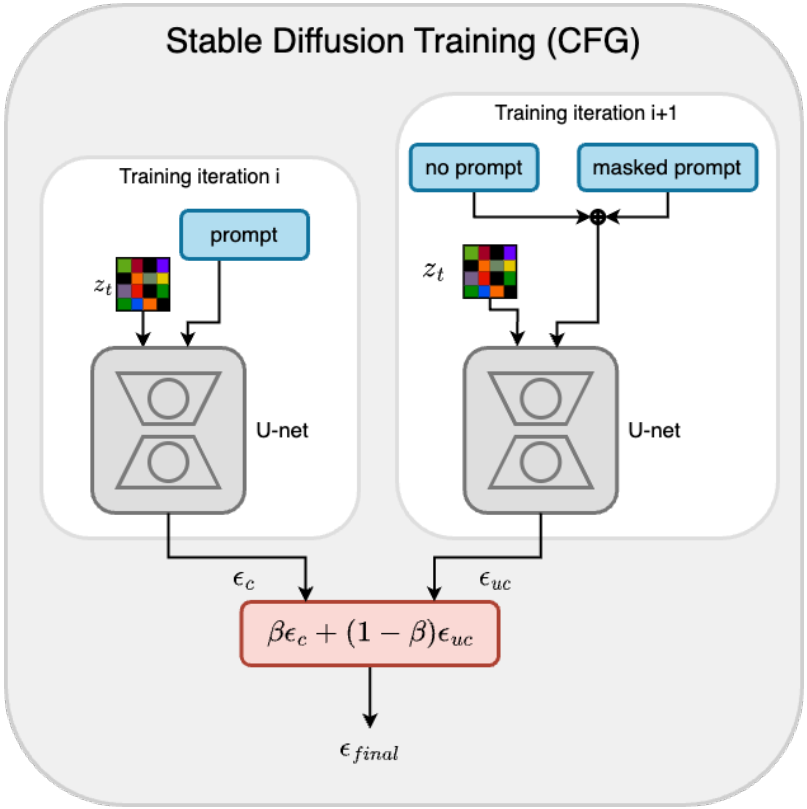
[6]

Classifier-free guidance

Classifier-free guidance (CFG) [3] guides sd to place emphasis on inputs with prompts over inputs without prompts.

$$\begin{aligned}\epsilon_{final} &= \epsilon_{uc} + \beta(\epsilon_c - \epsilon_{uc}) \\ &= \beta\epsilon_c + (1-\beta)\epsilon_{uc}\end{aligned}$$

- ϵ_c conditional image: has a prompt.
- ϵ_{uc} no prompt or masked prompt.
- A user defined weight β
- This effectively removes the influence of the unconditional image from the final prediction.



CFG

The effect of the linear combination is the same as having two models, one for unconditional images and one for conditional images, each parameterized separately, but trained jointly

Negative prompts

Negative prompts allow us to use prompts that describe what we DON'T want in an image to modify the output.

- Using the equation from [3] we get

$$\epsilon_{final} = (1 + \beta)\epsilon_c - \beta\epsilon_n$$

- ϵ_n a negative prompt.
- This modifies the latent image by removing the aspects captured by the negative prompt.

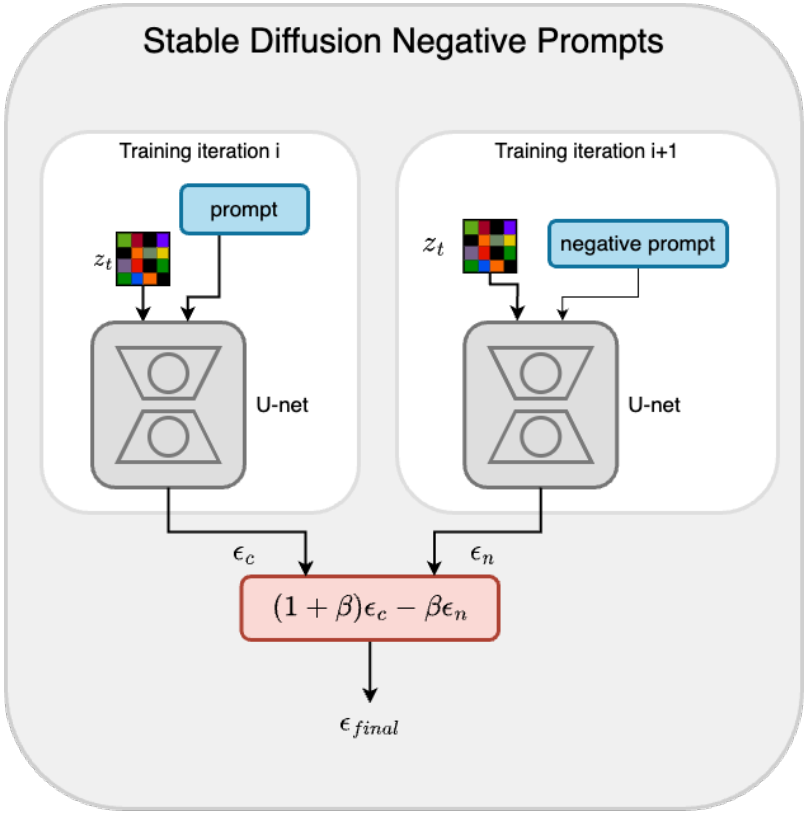
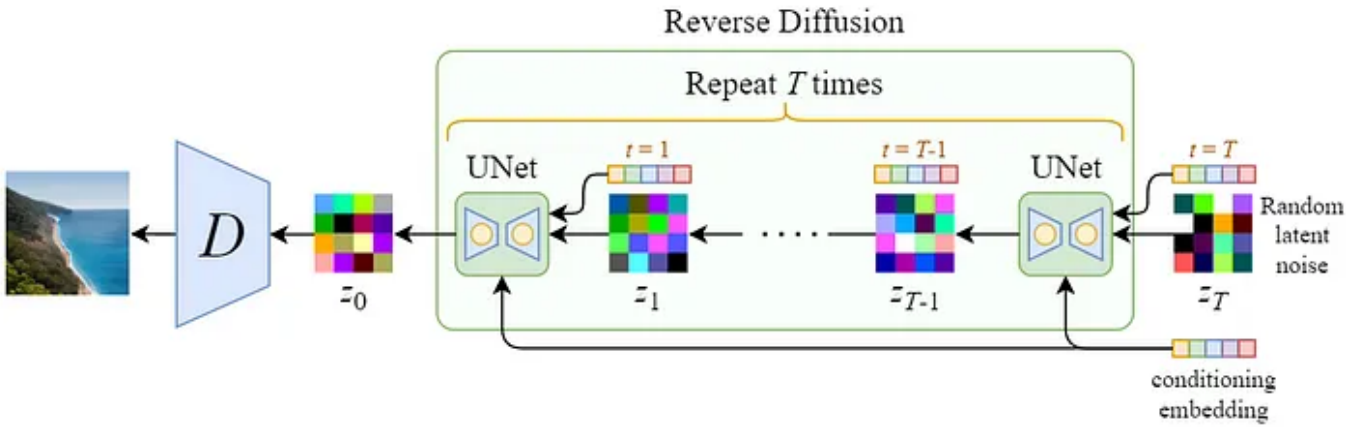


Image generation



Algorithm 1: Image generation

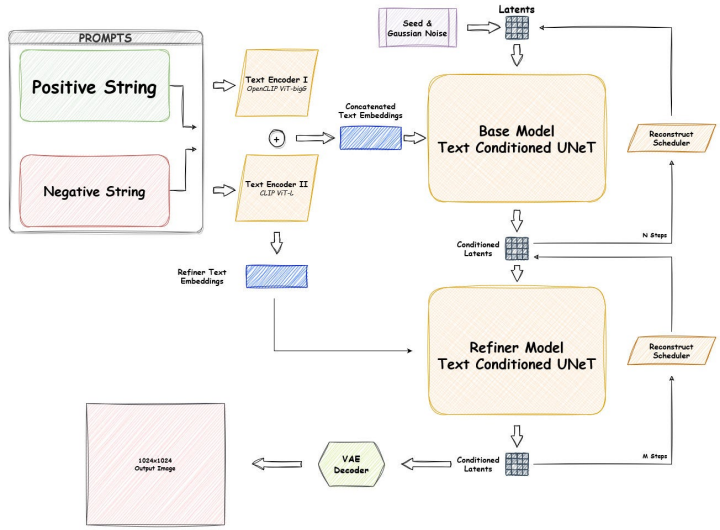
Data: $i = 0$
Result: z
$z =$ uniformly random noise;
$t_j =$ the learned, time step emebdding;
$\tau_\theta(y)$ The image description embedding;
while $i \leq T$ **do**
  $z = \epsilon_\theta(z, t_i, \tau_\theta(y))$;
end

[7]

Stable Diffusion XL (SDXL)

SDXL [5] improves SD to produce higher quality images

- Additions:
  - A 3x larger unet-backbone.
  - Shift bulk of transformer computation to lower levels of U-net
  - Uses two text-image encoders (OpenCLIP ViT-bigG, CLIP ViT-L)
  - Additional text conditioning on pooled text embedding from OpenCLIP.
  - Conditioning (image size, aspect ratio, cropping coordinates)
  - Improved image compression variational autoencoder.
  - A refiner model to improve the visual quality of samples.
  - Uses openclip, a public model, instead of the private clip model from openai.



[2]

SD image size minimum

Loss of training data.

- The original SD model has a minimum threshold for image resolution.
- Trivial solution
  - (1) Discard or (2) upscale image samples that below the threshold.
  - (1) Leads to loss of $\tilde{3}9\%$ of data, degrading model performance.
  - (2) Can introduce noise.

Conditioning on image size

To solve for the problems introduced by the image size minimum, the author's condition SDXL on image size

- The original width and height of the image are added as conditioning features.
- Embed width and height individually using fourier feature encoding
- Concatenate embeddings and add to timestep embedding as input

During inference, we can now change the generated image size based on the the conditioning paramters.

Cropped objects in SD

Problem: cropping leaking into generated images

- Different image sizes in a batch
- Crop images in a batch to match the target size

solution

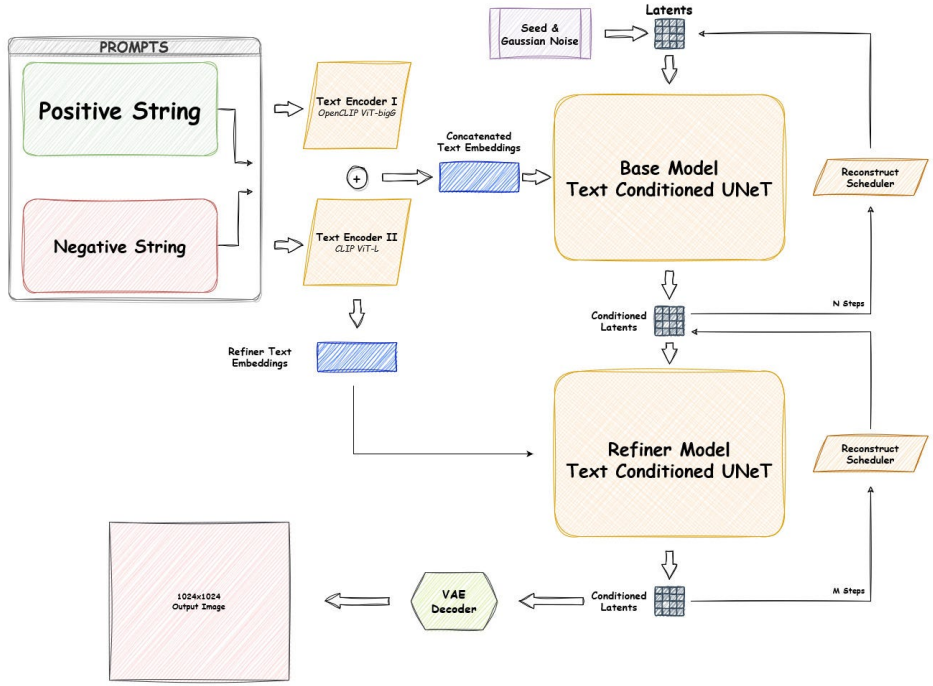
- Sample crop coordinates unfiormly at random.
- Transform the coordinates into Fourier feature embeddings.
- Concatenate the coordinate and size embeddings along the channel axis
- Feed as input to condition the model.

Figure 4: Comparison of the output of *SDXL* with previous versions of *Stable Diffusion*. For each prompt, we show 3 random samples of the respective model for 50 steps of the DDIM sampler [46] and cfg-scale 8.0 [13]. Additional samples in Fig. 14.

[5]

Image Refinement

- Problem: noisy patches in images
- Solution
  - Train model SD_{edit} only on super high-resolution images.
  - Diffuse the latent representation of the generated image with noisy patches z_t.
  - Denoise the diffused z_t with SD_{edit} using same prompt used to generate the image.
  - SD_{edit} specialized on the first 200 discrete noise scales.



[2]

ControlNet

ControlNet modifies the SD architecture to allow for conditioning on additional input images.

[9]

ControlNet

- Freezes U-net encoder weights.
- Create a **trainable copy** of the U-net encoder weights.
- Conditional input image transformed with simple convolution.
- Outputs passed to skip connections of U-nets decoder through a zero-convolution

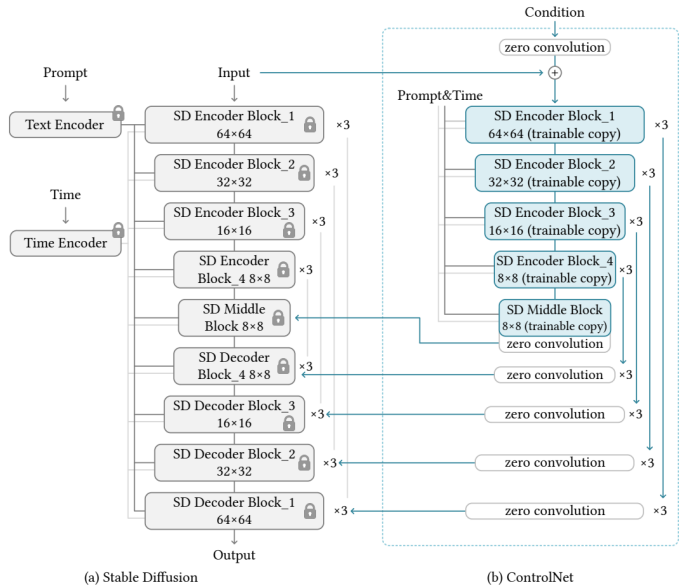
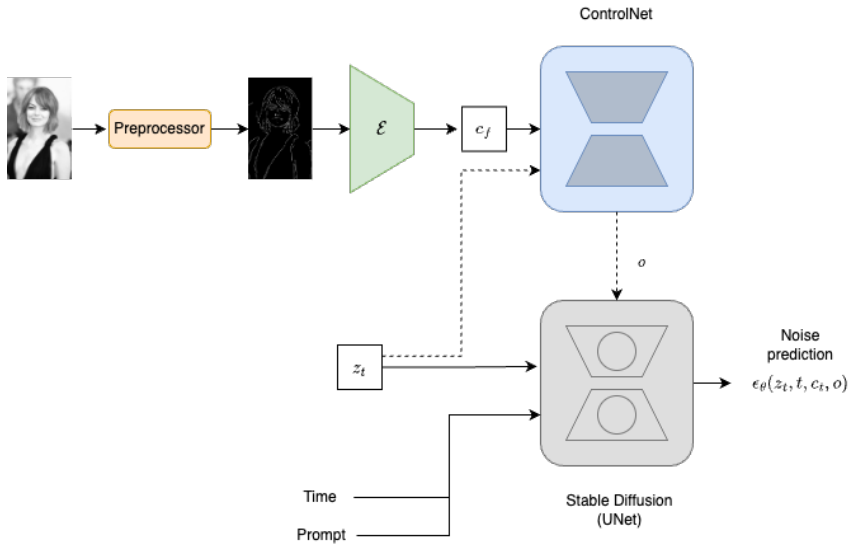


Figure 3: ControlNet in Stable Diffusion. The gray blocks are the structure of Stable Diffusion 1.5 (or SD V2.1, since they use the same U-Net architecture), while the blue blocks are ControlNet.

Training Flow

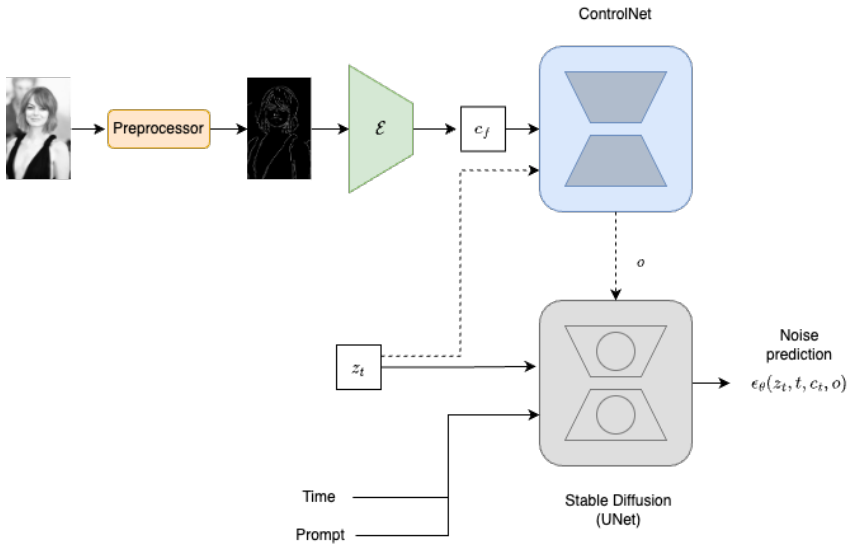
- Choose one conditional input image type (ie canny-edge)
- Use a preprocessor to transform your training images into canny-edge versions of the image.
- Input is prompt (50% no prompt), noisy image, and canny-edge image.
- Sudden convergence phenomenon: a pre-trained latent diffusion model "abruptly succeeds in following the input conidtioning image; usually in less than 10k optimization steps." [9]



ControlNet training & objective

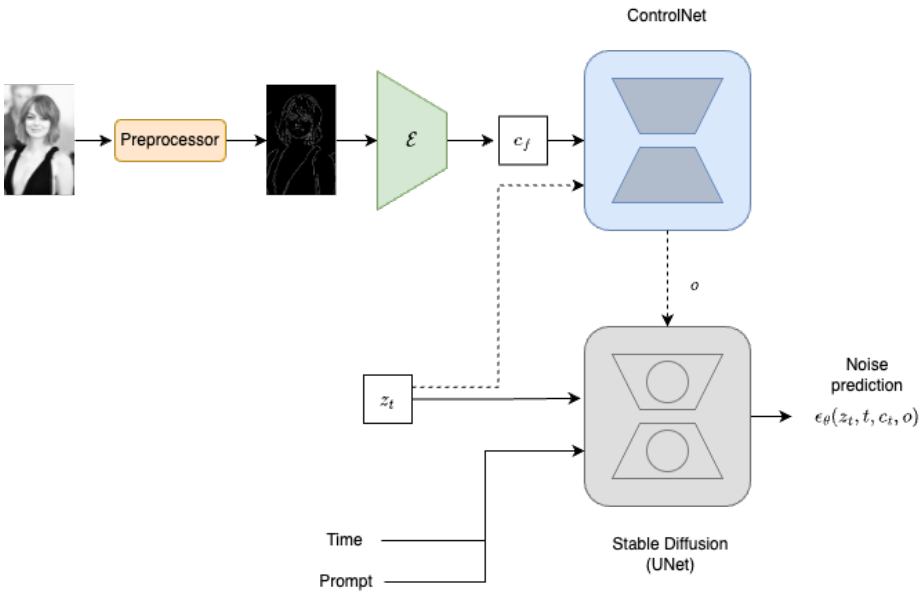
$$l_{ldm} = z_{\mathbf{0}}, t, c_t, c_f, \epsilon \sim \setminus(\mathbf{0}, \mathbf{1}) |\epsilon - \epsilon_\theta(z_t, t, c_t, c_f)|_{\mathbf{2}}^{\mathbf{2}}$$

- z_0 input image.
- t time step embedding.
- c_t text conditioning embedding with 50% of strings replaced with blanks to place embphasis on control image.
- c_f control image embedding.



Training

- During training, 50% of text prompts are replaced with empty strings at random, placing emphasis on the conditional image.
- One ControlNet model trained per conditional input image type (canny edge, hough line, etc)



Prompt

Controlnet performs well without prompt guidance.

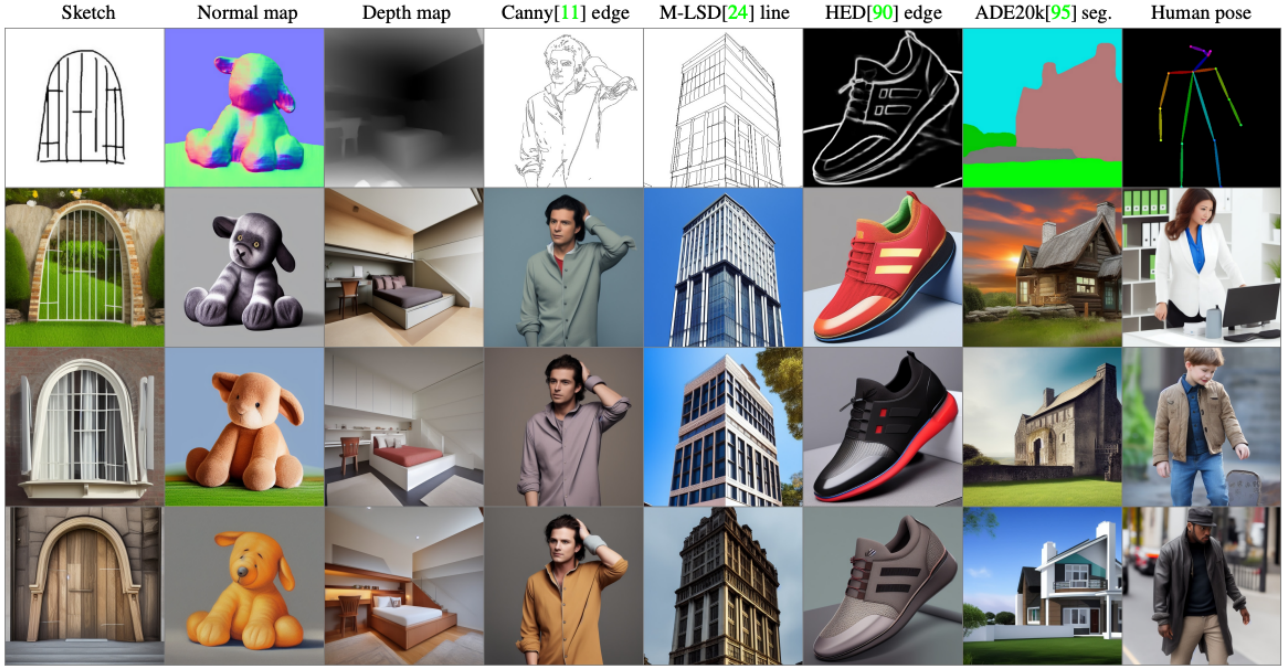


Figure 7: Controlling Stable Diffusion with various conditions **without prompts**. The top row is input conditions, while all other rows are outputs. We use the empty string as input prompts. All models are trained with general-domain data. The model has to recognize semantic contents in the input condition images to generate images.

Ablative Study

The conditional input image is powerful.

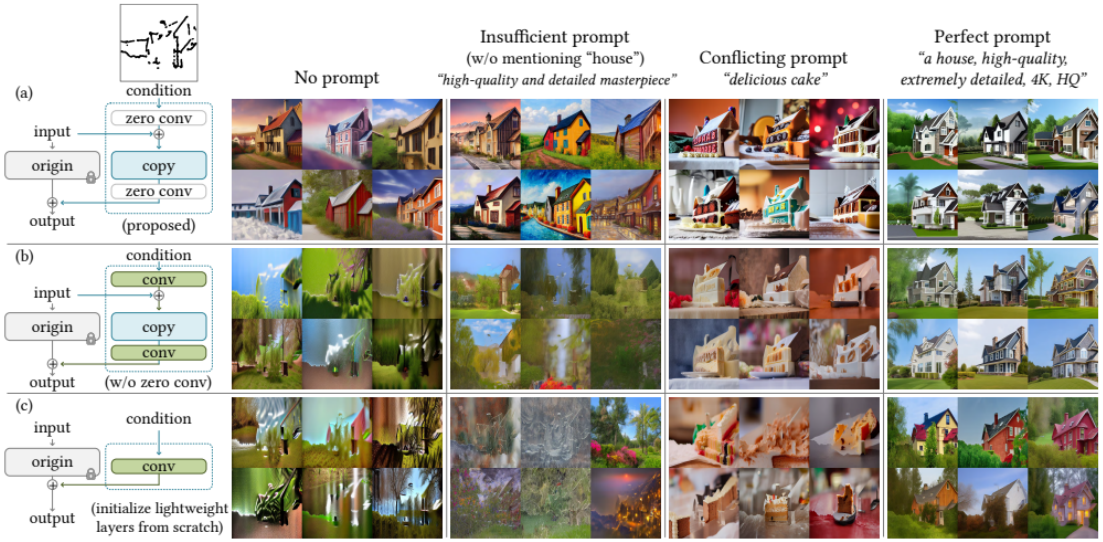


Figure 8: Ablative study of different architectures on a sketch condition and different prompt settings. For each setting, we show a random batch of 6 samples without cherry-picking. Images are at 512 × 512 and best viewed when zoomed in. The green "conv" blocks on the left are standard convolution layers initialized with Gaussian weights.

Composability

Controlnet is composable. do we train several ControlNets on one SD model?

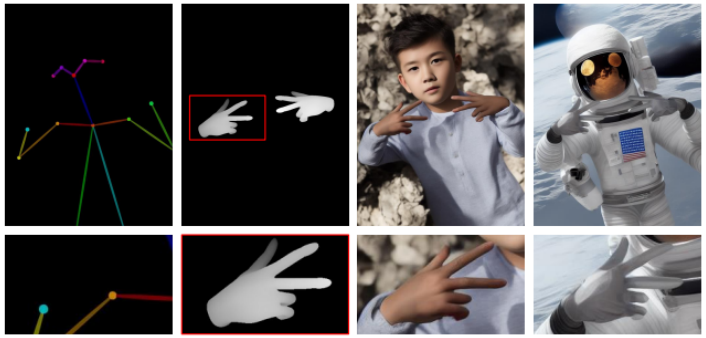


Figure 6: Composition of multiple conditions. We present the application to use depth and pose simultaneously.

Classifier-free guidance resolution weighting

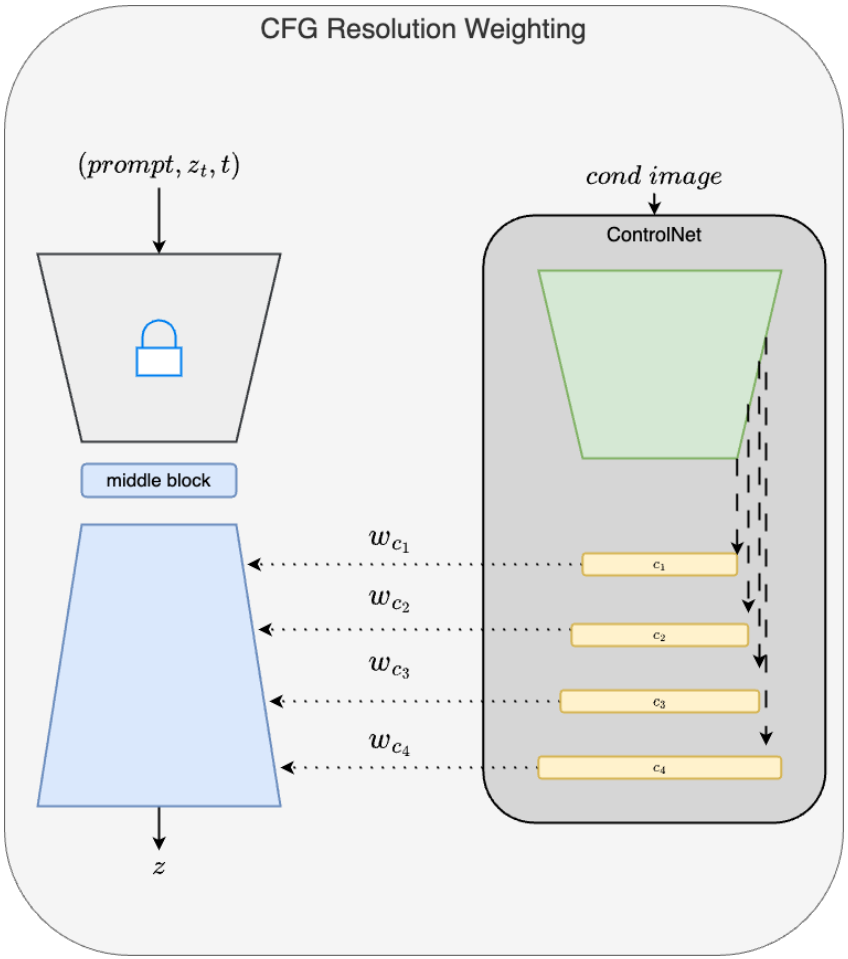
CFG resolution weighting for controlnet

- If left unmodfied, controlnet will add the conditional image to both ϵ_{uc} and ϵ_c.
- The conditional guidance on ϵ_{uc} will nullify the masked prompt, removing cfg.
- If added only to ϵ_c, the model will place to much emphasis on the conditional image.

Classifier-free guidance resolution weighting

Solution: Leverage the scaling properties inherent to the architecture to weight CFG.

- Add the conditional image to ϵ_c
- Mulitply each connection between stable diffusion and controlnet by a weight that reduces the strength of the conditional image according to the resolution of each block



Bibliography I

[1] AssemblyAI. In: (). URL: https://www.assemblyai.com/blog/stable-diffusion-1-vs-2-what-you-need-to-know/.

[2] demir. In: (). URL: https://towardsdatascience.com/the-arrival-of-sdxl-1-0-4e739d5cc6c7.

[3] Jonathan Ho and Tim Salimans. *Classifier-Free Diffusion Guidance*. 2022. arXiv: 2207.12598 [cs.LG].

[4] OpenAI. In: (). URL: https://openai.com/research/clip.

[5] Dustin Podell et al. *SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis*. 2023. arXiv: 2307.01952 [cs.CV].

[6] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. arXiv: 2112.10752 [cs.CV].

[7] steins. "Stable Diffusion Clearly Explained!" In: (2022). URL: https://medium.com/@steinsfu/stable-diffusion-clearly-explained-ed008044e07e.

[8] UNI. In: (). URL: https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/.

Bibliography II

[9] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. *Adding Conditional Control to Text-to-Image Diffusion Models*. 2023. arXiv: 2302.05543 [cs.CV].