Carnegie Mellon University

Intermediate Deep Learning

Spring 2025, Deep Learning for Engineers
March 25, 2025, 5th Session

Amir Barati Farimani
Associate Professor of Mechanical Engineering and Bio-Engineering
Carnegie Mellon University

Diffusion Models

Diffusion Model

"Diffusion Models Beat GANs on Image Synthesis" Dhariwal & Nichol, OpenAI, 2021

"Cascaded Diffusion Models for High Fidelity Image Generation" Ho et al., Google, 2021

Carnegie Mellon University

Diffusion Model

Figure: Generated images by Midjourney v4.

Carnegie Mellon University

Text To Image Generation

DALL. E2

“a teddy bear on a skateboard in times square”

“Hierarchical Text-Conditional Image Generation with CLIP Latents” Ramesh et al.,2022

Imagen

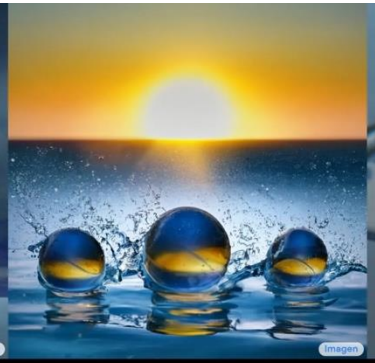
A group of teddy bears in suit in a corporate office celebrating the birthday of their friend. There is a pizza cake on the desk.

“Photorealistic Tex-to-Image Diffusion Models with Deep Language Understanding”, Saharia et al.,2022

Carnegie Mellon University

Imagen

An alien octopus floats through a portal reading a newspaper.



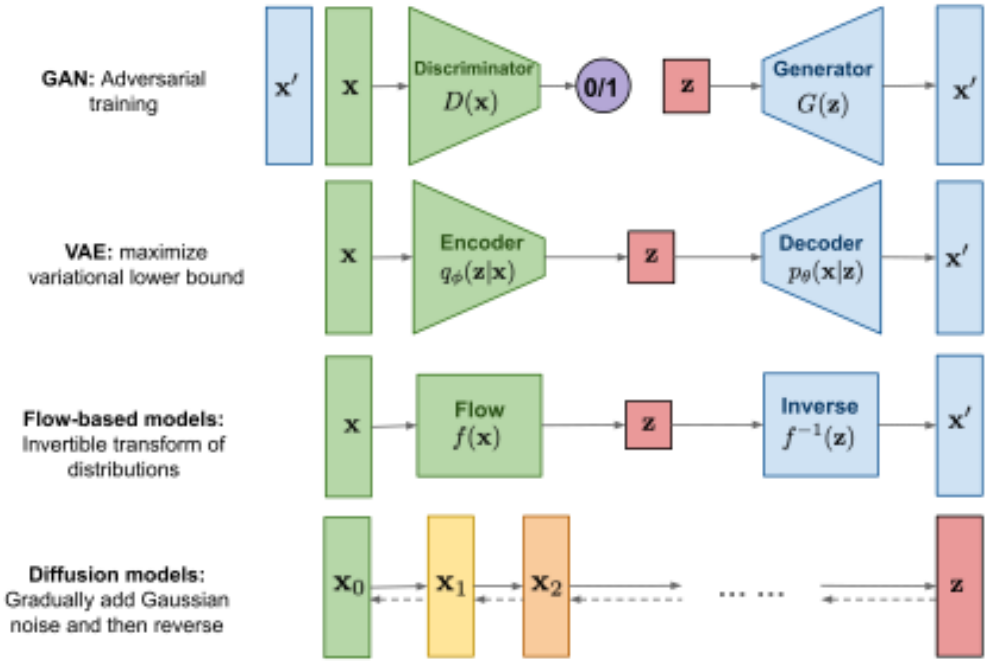
Three spheres made of glass falling into ocean. Water is splashing. Sun is setting.

A strawberry mug filled with white sesame seeds. The mug is floating in a dark chocolate sea.

A chrome-plated duck with a golden beak arguing with an angry turtle in a forest.

Carnegie Mellon University

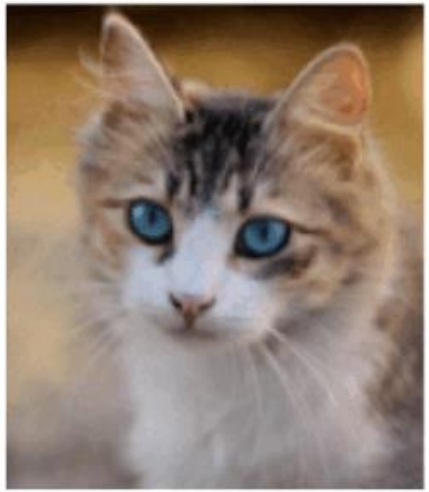
Generative Models



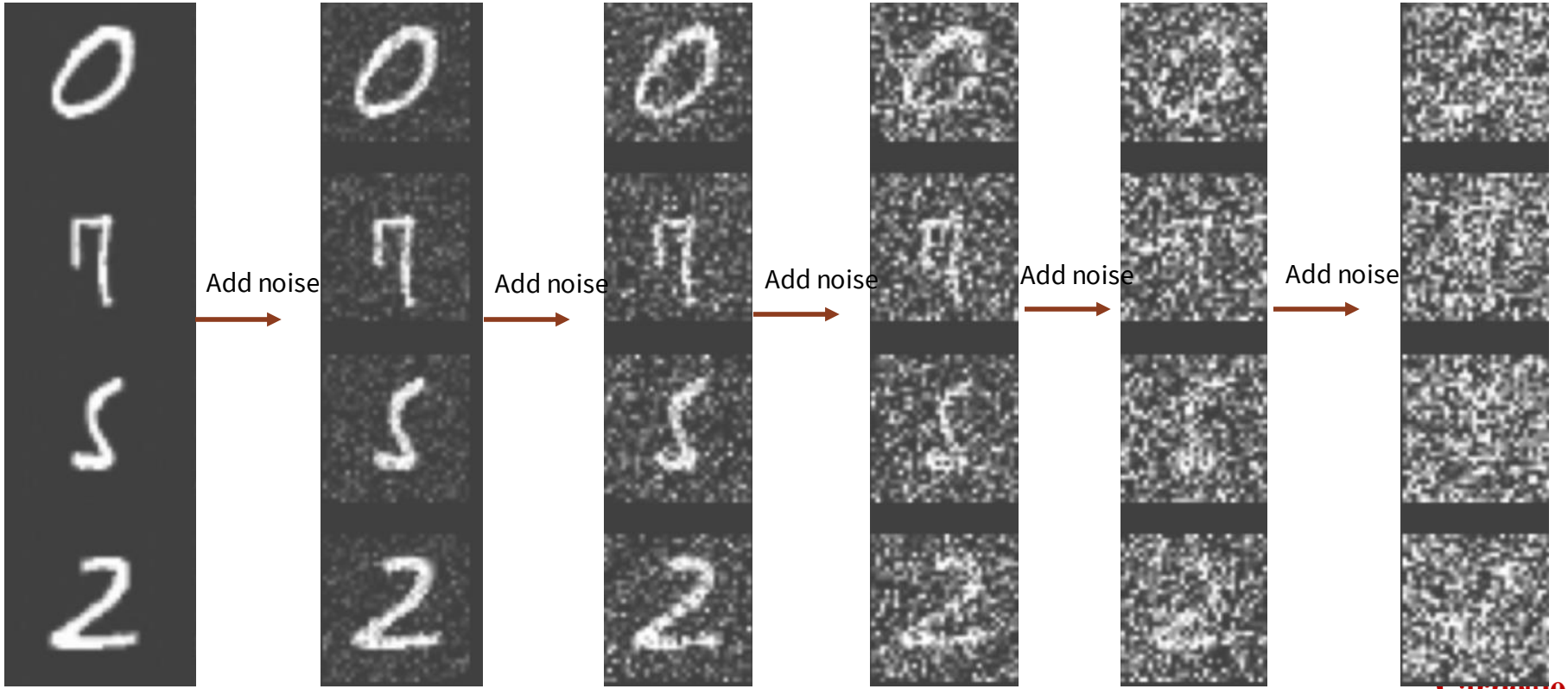
Carnegie Mellon University

Diffusion Models

Diffusion models:
Gradually add Gaussian noise and then reverse



Diffusion Models



Diffusion Models

X1

F → P(X1)=0.98

X2

F → P(X2)=0.78

X3

F → P(X3)=0.48

X4

F → P(X4)=0.2

Diffusion Models

X1

F → P(X1)=0.98

X2

F → P(X2)=0.78

X3

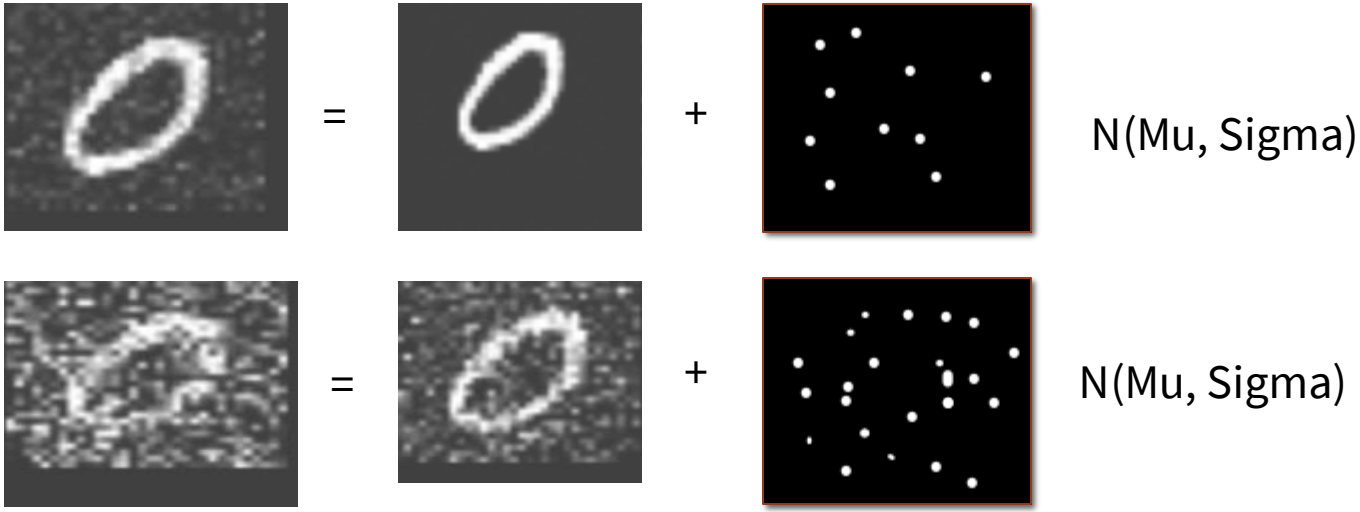
F → P(X3)=0.48

X4

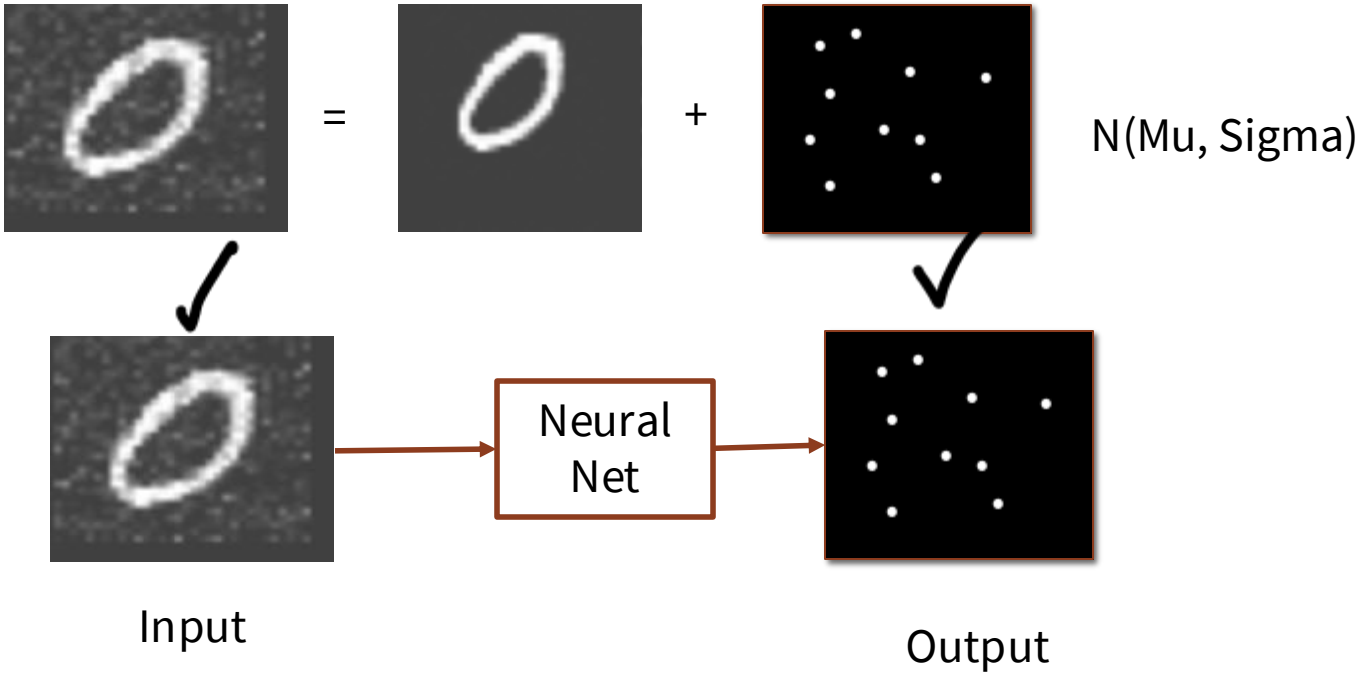
F → P(X4)=0.2

$$\frac{\text{Grad}(P(X1))}{\text{Grad}(X1)}$$

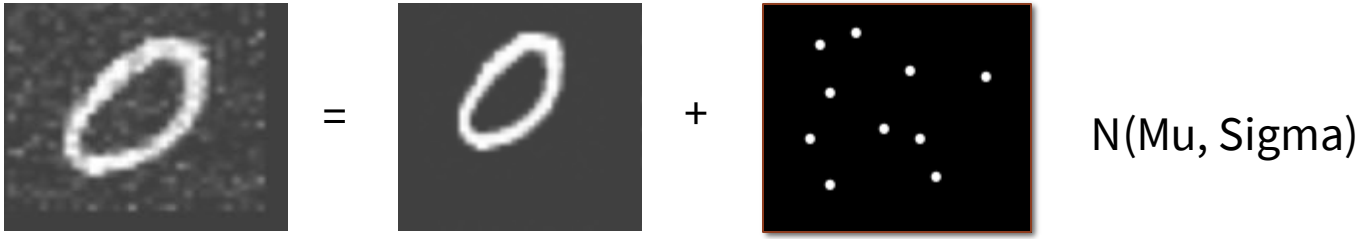
Diffusion Models

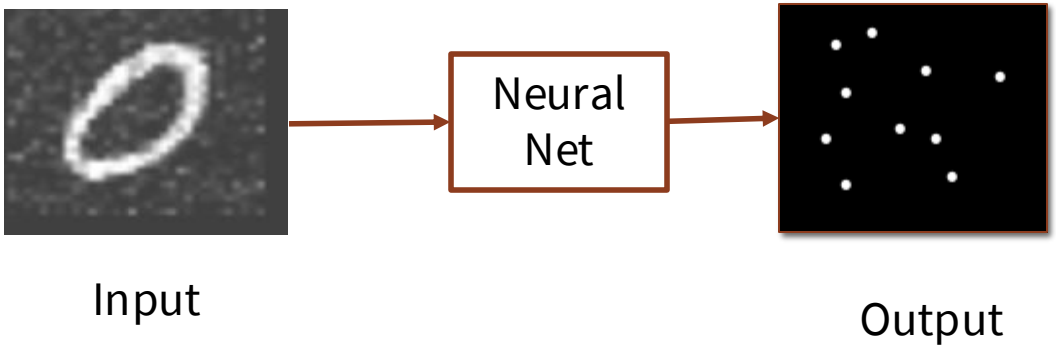


Diffusion Models

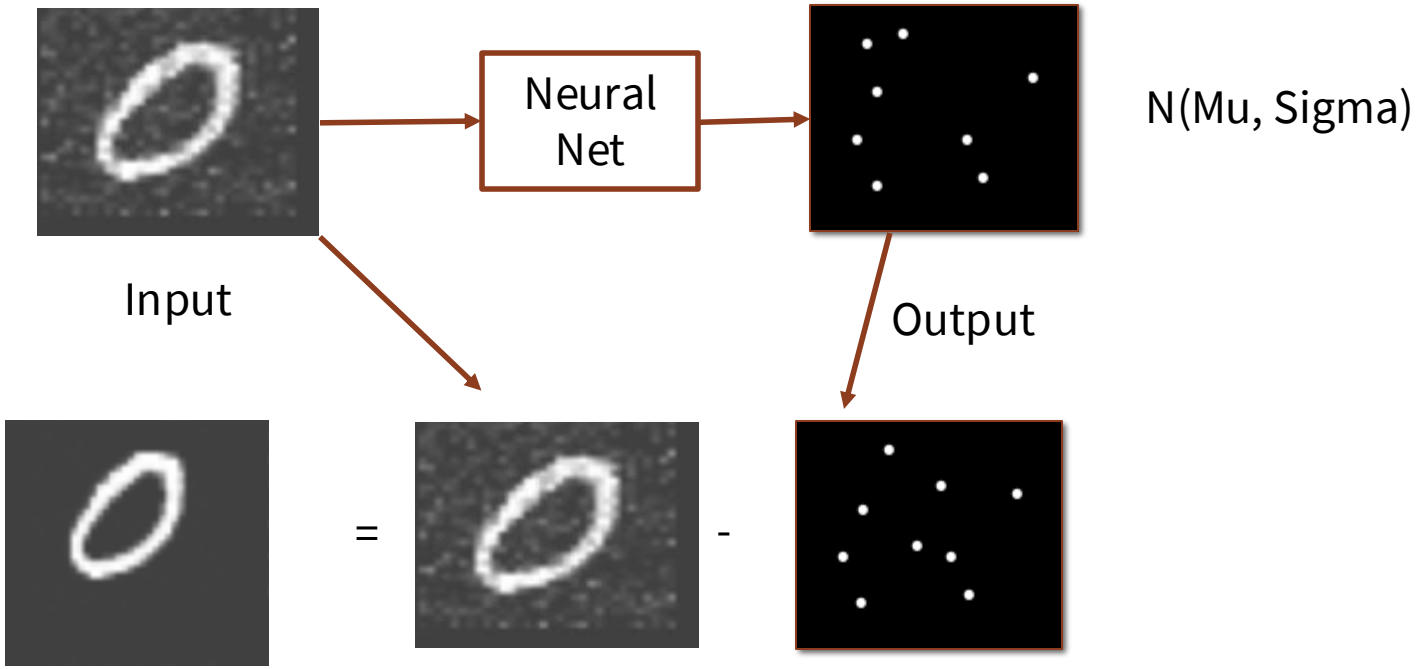


Diffusion Models





Diffusion Models



Denoising Diffusion Models
Learning to generate by denoising

Denoising diffusion models consist of two processes:

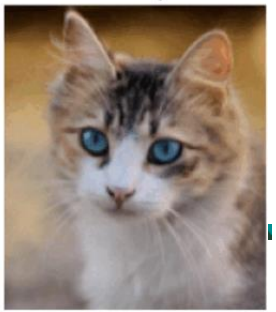
- Forward diffusion process that gradually adds noise to input
- Reverse denoising process that learns to generate data by denoising

Forward diffusion process (fixed)

Data

Noise

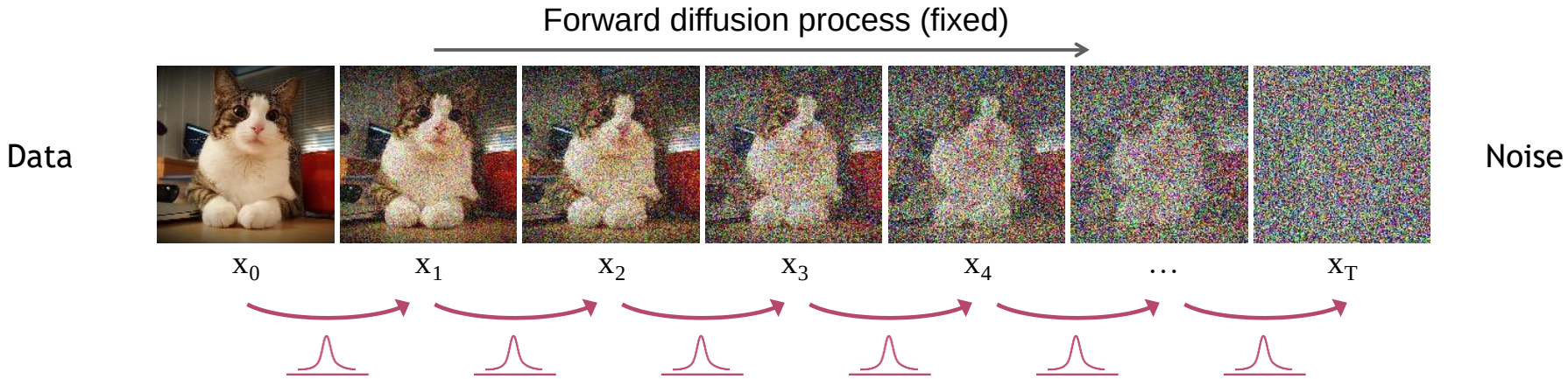
Reverse denoising process (generative)



Sohl-Dickstein et al., Deep Unsupervised Learning using Nonequilibrium Thermodynamics, ICML 2015 Ho et al., Denoising Diffusion Probabilistic Models, NeurIPS 2020
Song et al., Score-Based Generative Modeling through Stochastic Differential Equations, ICLR 2021

Forward Diffusion Process

The formal definition of the forward process in T steps:



$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I}) \quad \Rightarrow \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \quad \text{(joint)}$$

Carnegie Mellon University

Forward Process

$$q(x_t|x_{t-1}) = N(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$$

$$q(x_t|x_0) = N(x_t; \sqrt{\bar{\alpha}_t}x_0, 1-\bar{\alpha}_t I)$$

- It admits sampling x_t at an arbitrary timestep t

x_0 x_{50} x_{100} x_{300} x_{500} x_{700} x_{900} x_{1000}

Variance schedule

β_t linear from 0.0001 ~ 0.02

Timestep

T = 1000

Notation

$$\alpha_t = 1 - \beta_t$$

$$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s \ , \ \bar{\alpha_T} \to 0$$

$$\epsilon \sim N(0,1)$$

Reparameterization

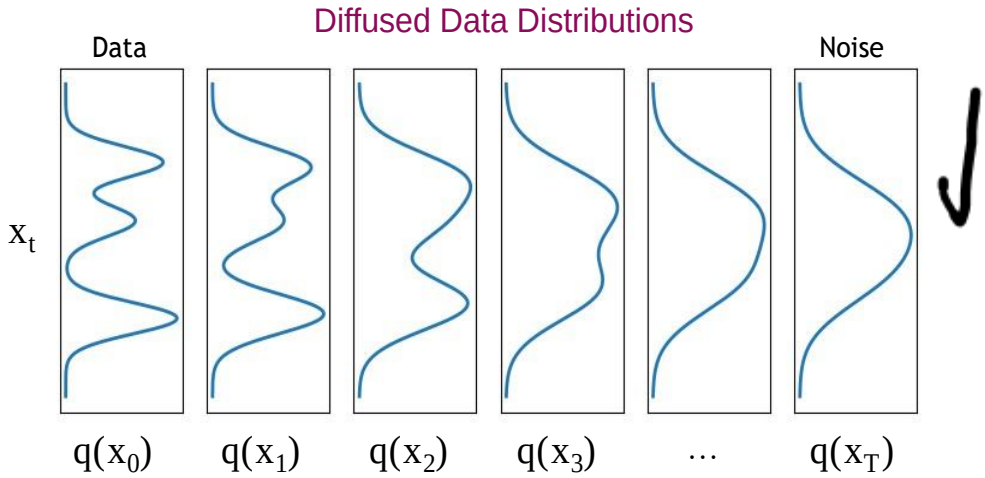
$$N(\mu, \sigma^2) = \mu + \sigma\epsilon$$

Carnegie Mellon University

Distribution in the forward diffusion

$$\underbrace{q(\mathbf{x}_t)}_{\text{Diffused data dist.}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\text{Joint dist.}} d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\text{Input data dist.}} \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{Diffusion kernel}} d\mathbf{x}_0$$

The diffusion kernel is Gaussian convolution.

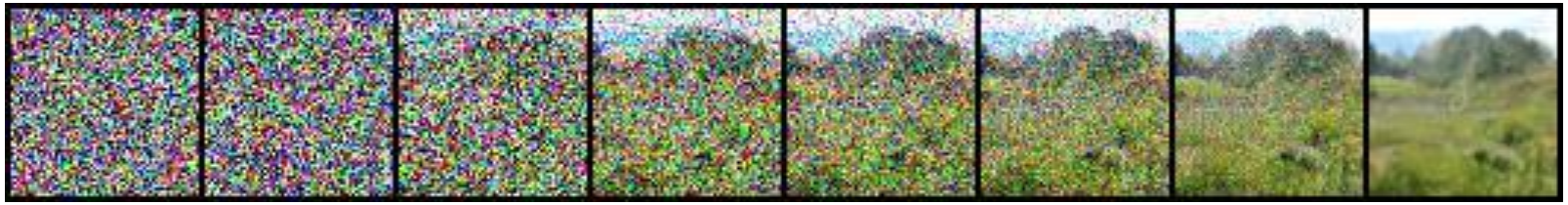


We can sample $\mathbf{x}_t \sim q(\mathbf{x}_t)$ by first sampling $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ and then sampling $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_0)$ (i.e., ancestral sampling).

Reverse Process

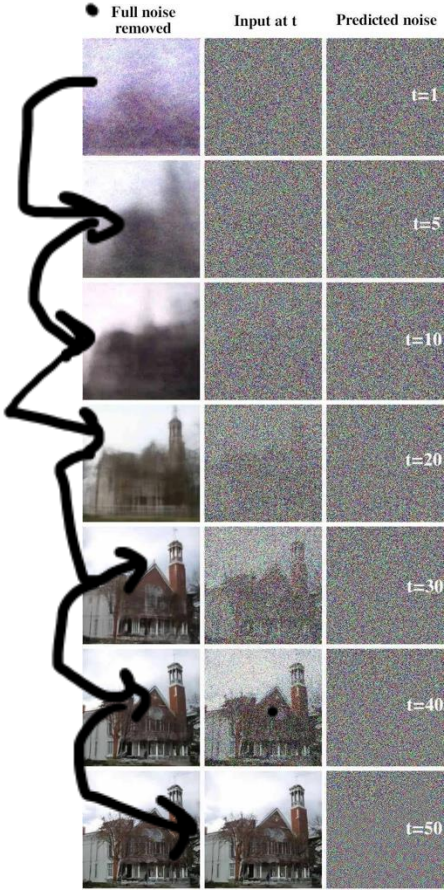
$$p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$
$$= N(x_{t-1}; \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)), \beta_t)$$

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)) + \sqrt{\beta_t}\epsilon$$



x_{1000} x_{800} x_{400} x_{200} x_{150} x_{100} x_{50} x_0

Reverse Process



$$p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

$$= N(x_{t-1}; \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)), \beta_t)$$

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)) + \sqrt{\beta_t}\epsilon$$

Carnegie Mellon University

Training Objective

$$\mathbb{E}[-\log p_\theta(\mathbf{x}_0)] \leq \mathbb{E}_q\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t>1} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L$$

$$L_{t-1} - C = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{1}{2\sigma_t^2}\left\|\tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}), \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}) - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon})\right) - \boldsymbol{\mu}_\theta(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}), t)\right\|^2\right]$$

$$= \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}\right) - \boldsymbol{\mu}_\theta(\mathbf{x}_t(\mathbf{x}_0,\boldsymbol{\epsilon}), t)\right\|^2\right]$$

$$\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

We can use our model to predict x_0 or μ_θ or ϵ_θ , and the author choose to predict ϵ_θ

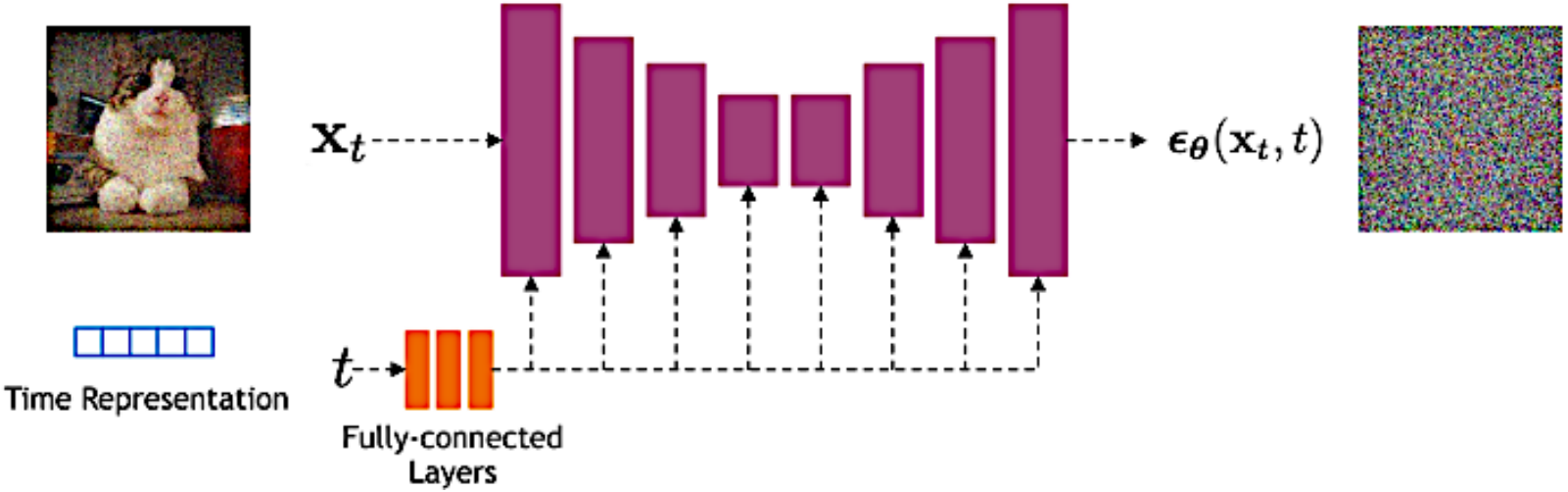
Note:

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t$$

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t))\right) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$$

Carnegie Mellon University

Network Architecture



- Unet with residual block and self-attention and add time embeding to conditioned with time step

Ref: https://cvpr2022-tutorial-diffusion-models.github.io/

Carnegie Mellon University

Training and Sampling

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

- Forward Process
- Loss Function

- Reverse Process

Carnegie Mellon University

Experiment

When conditioned on the same latent, CelebA-HQ 256×256 samples share high-level attributes. Bottom-right quadrants are $\mathbf{x}_t$, and other quadrants are samples from $p_\theta(\mathbf{x}_0|\mathbf{x}_t)$.

Carnegie Mellon University

Experiment

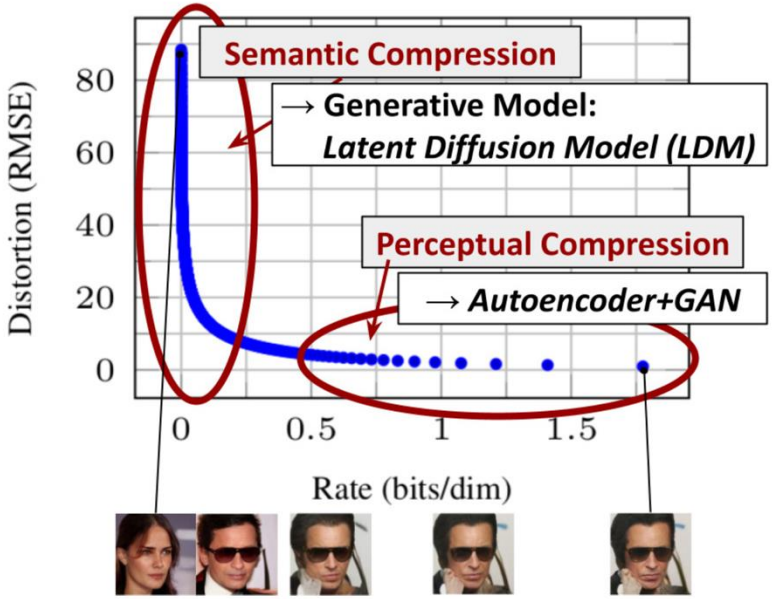
Unconditional CIFAR10 reverse process parameterization and training objective ablation. Blank entries were unstable to train and generated poor samples with out-of-range scores.

Objective	IS	FID
$\tilde{\mu}$ prediction (baseline)		
L, learned diagonal Σ	7.28 ± 0.10	23.69
L, fixed isotropic Σ	8.06 ± 0.09	13.22
$\|\tilde{\mu} - \tilde{\mu}_\theta\|^2$	–	–
ϵ prediction (ours)		
L, learned diagonal Σ	–	–
L, fixed isotropic Σ	7.67 ± 0.13	13.51
$\|\tilde{\epsilon} - \epsilon_\theta\|^2$ (L_{simple})	$\mathbf{9.46 \pm 0.11}$	**3.17**

Carnegie Mellon University

Latent Diffusion Model

Latent diffusion model (**LDM**; Rombach & Blattmann, et al. 2022) runs the diffusion process in the latent space instead of pixel space, making training cost lower and inference speed faster. It is motivated by the observation that most bits of an image contribute to perceptual details and the semantic and conceptual composition still remains after aggressive compression. LDM loosely decomposes the perceptual compression and semantic compression with generative modeling learning by first trimming off pixel-level redundancy with autoencoder and then manipulate/generate semantic concepts with diffusion process on learned latent



Carnegie Mellon University

Latent Diffusion Model (Stable Diffusion)

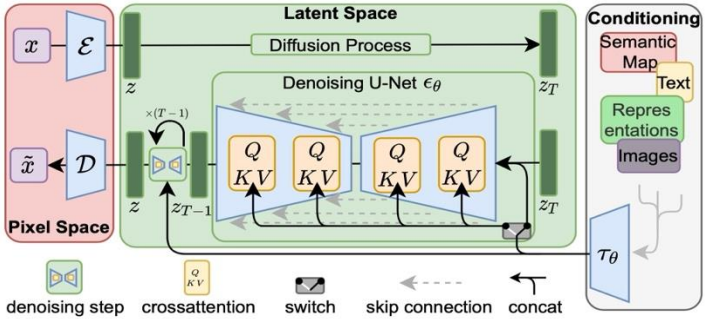
The perceptual compression process relies on an autoencoder model An encoder E compress the input image $\mathbf{x} \in \mathrm{R}^{H \times W \times 3}$ to a smaller $2D$ latent vector $\mathbf{z} = E(\mathbf{x}) \in \mathrm{R}^{h \times w \times c}$, where the downsampling rate $f = H/h = W/w = 2^m, m \in \mathrm{N}$

The downsampling rate $f = H/h = W/w = 2^m, m \in \mathrm{N}$. Then an decoder D reconstructs the images from the latent vector, $\mathbf{x}\tilde{} = D(\mathbf{z})$

Two types of regularization in autoencoder training are used to avoid arbitrarily high-variance in the latent spaces

KL-reg: A small KL penalty towards a standard normal distribution over the learned latent, similar to VAE.

VQ-reg uses a vector quantization layer within the decoder, like VQVAE but the quantization layer is absorbed by the decoder



Latent diffusion model (**LDM**; Rombach & Blattmann, et al. 2022)

Carnegie Mellon University

Latent Diffusion Model (Stable Diffusion)

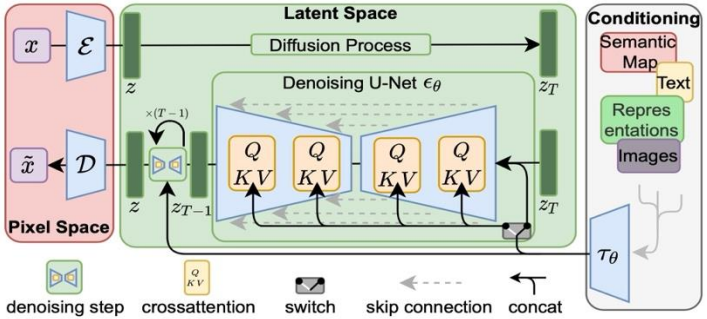
The perceptual compression process relies on an autoencoder model An encoder E compress the input image $\mathbf{x} \in \mathrm{R}^{H \times W \times 3}$ to a smaller $2D$ latent vector $\mathbf{z} = E(\mathbf{x}) \in \mathrm{R}^{h \times w \times c}$, where the downsampling rate $f = H/h = W/w = 2^m, m \in \mathrm{N}$

The downsampling rate $f = H/h = W/w = 2^m, m \in \mathrm{N}$. Then an decoder D reconstructs the images from the latent vector, $\mathbf{x}\tilde{} = D(\mathbf{z})$

Two types of regularization in autoencoder training are used to avoid arbitrarily high-variance in the latent spaces

KL-reg: A small KL penalty towards a standard normal distribution over the learned latent, similar to VAE.

VQ-reg uses a vector quantization layer within the decoder, like VQVAE but the quantization layer is absorbed by the decoder



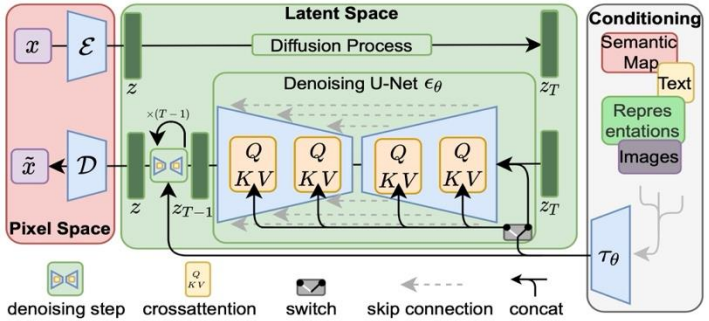
Latent diffusion model (**LDM**; Rombach & Blattmann, et al. 2022)

Carnegie Mellon University

Latent Diffusion Model (Stable Diffusion)

The diffusion and denoising processes happen on the latent vector $\mathbf{z}$ The denoising model is a time-conditioned U-Net, augmented with the cross-attention to handle flexible conditioning information(e.g. class labels, semantic maps, blurred variants of an image).)

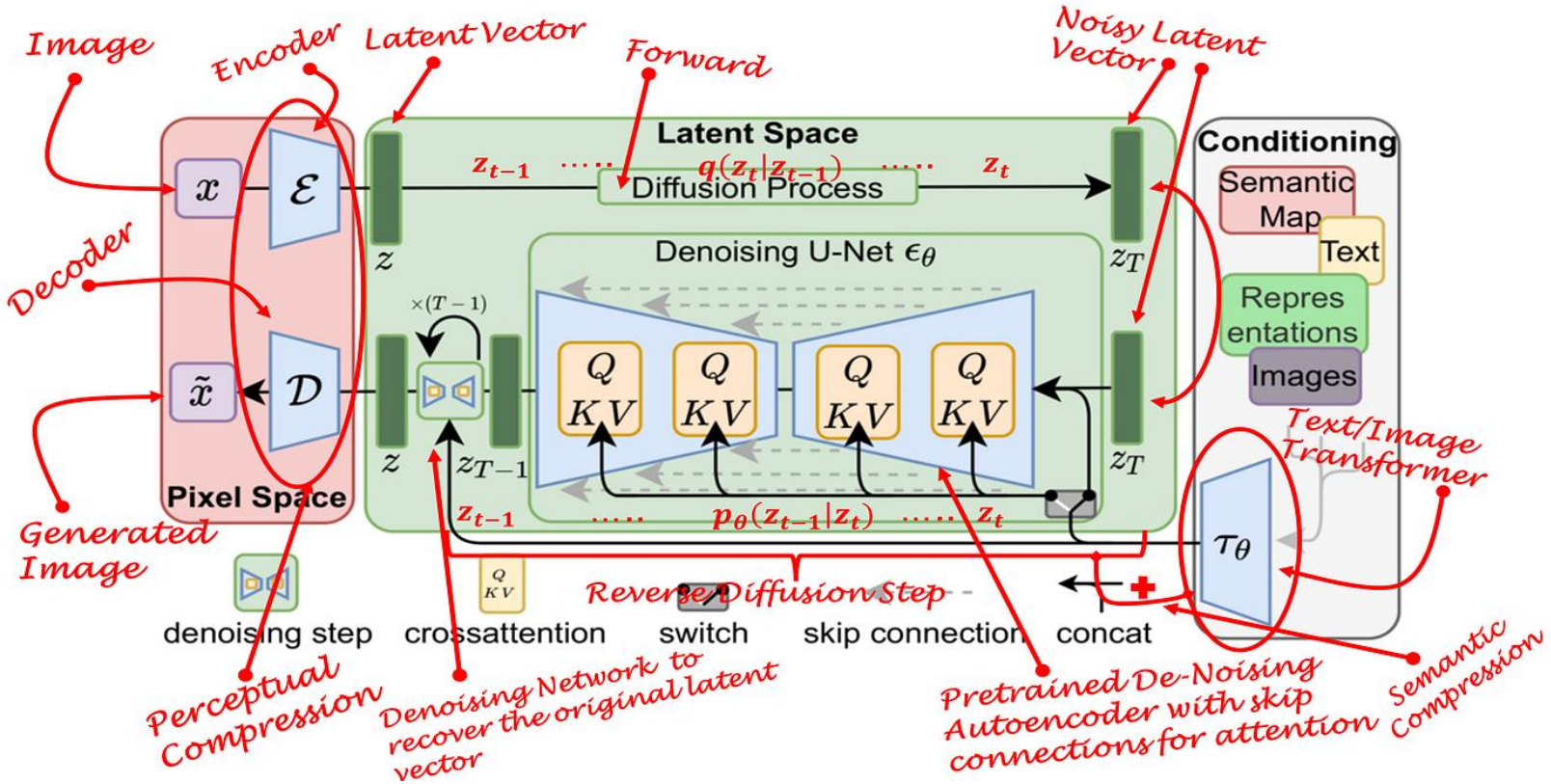
Each type of condition is paired with a domain-specific encoder τ_θ to project the conditioning input y to an intermediate representation that can be mapped into cross-attention component, $\tau_\theta(y) \in \mathbb{R}^{M \times d_\tau}$.



Latent diffusion model (**LDM**; Rombach & Blattmann, et al. 2022)

Carnegie Mellon University

Latent Diffusion Model (Stable Diffusion)



Latent diffusion model (**LDM**; Rombach & Blattmann, et al. 2022)

Carnegie Mellon University

How to accelerate sampling process?

- During test time, one needs to run the diffusion model > 1000 steps, which is slow!
- If we naively sample less steps, the quality is worse.
- Can we somehow sample less steps and have the quality?
- DDIM sampling shows how this can be done! (Most recent papers use this trick)
- Note that: During training, we only train for t-1 to t and randomly sample t between [0,T]. We do no need to run the whole generation process

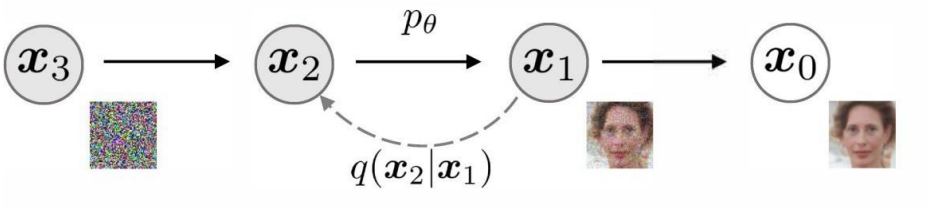
DDIM sampling overview

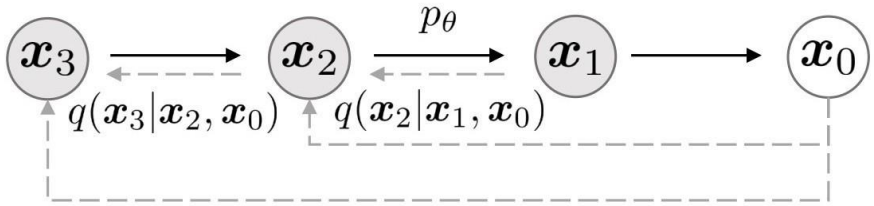
Key Idea:

- A regular diffusion model is Markovian process -> Generation at time t-1 only depends on time t, and independent of all other time stamps.
- Diffusion Markov process has optimal solution when we minimize this loss function:

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

- We want to use the trained Markovian Diffusion process at test time, but for faster sampling we define a non-Markovian process whose optimal solution is obtained by minimizing the same loss function as Markovian one!





DDIM sampling overview

Markovian Diffusion (often called DDPM sampling – Denoising Diffusion Probabilistic Model):

- Given noisy image X_t -> predict noise map $\epsilon_\theta(\mathbf{x}_t, t)$ -> subtract noise map from X_t (with some weighting) to obtain X_{t-1}

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$
$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$$

Non-Markovian Diffusion (DDIM sampling):

- Given noisy image X_t -> predict noise map $\epsilon_\theta(\mathbf{x}_t, t)$ -> generate clean image X'_0 -> add the predicted noise map to X'_0 to obtain X_{t-1}.

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}{\sqrt{\alpha_t}}\right)}_{\text{“predicted } \boldsymbol{x}_0\text{”}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}_{\text{“direction pointing to } \boldsymbol{x}_t\text{”}} + \underbrace{\sigma_t\epsilon_t}_{\text{random noise}}$$

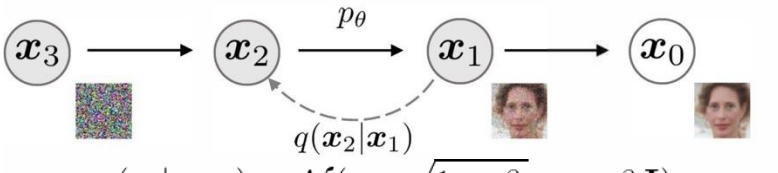
- Different choice of σ results in different generative process without re-training the model

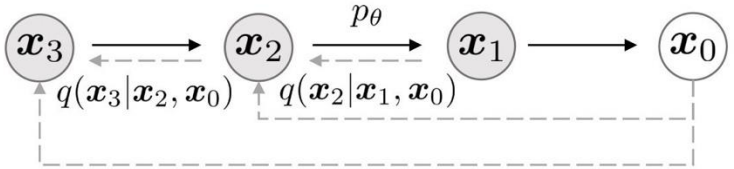
- When $\sigma = 0$ for all t, we have a deterministic generative process, with randomness from only t=T (the last step).

Carnegie Mellon University

Denoising Diffusion Implicit Models (DDIM)

Why can we do this?





$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I})$$
$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I})$$

Regular diffusion model

Define a family of forward processes that meets the above requirement:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}-\tilde{\sigma}_t^2}\cdot\frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0}{\sqrt{1-\bar{\alpha}_t}}, \tilde{\sigma}_t^2\mathbf{I}\right)$$

The corresponding reverse process is

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_0 + \sqrt{1-\bar{\alpha}_{t-1}-\tilde{\sigma}_t^2}\cdot\frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0}{\sqrt{1-\bar{\alpha}_t}}, \tilde{\sigma}_t^2\mathbf{I}\right)$$

$$:= (\boldsymbol{x}_t - \sqrt{1-\alpha_t}\cdot\epsilon_\theta^{(t)}(\boldsymbol{x}_t))/\sqrt{\alpha_t}.$$

This has same loss function as the regular/Markovian diffusion model. Thus we can use this strategy without re-training.

Outline for today's class

- THEORY:
  - How do we condition diffusion model?
  - How do people sample in practice? – DDIM sampling
  - How do we generate high-resolution images?

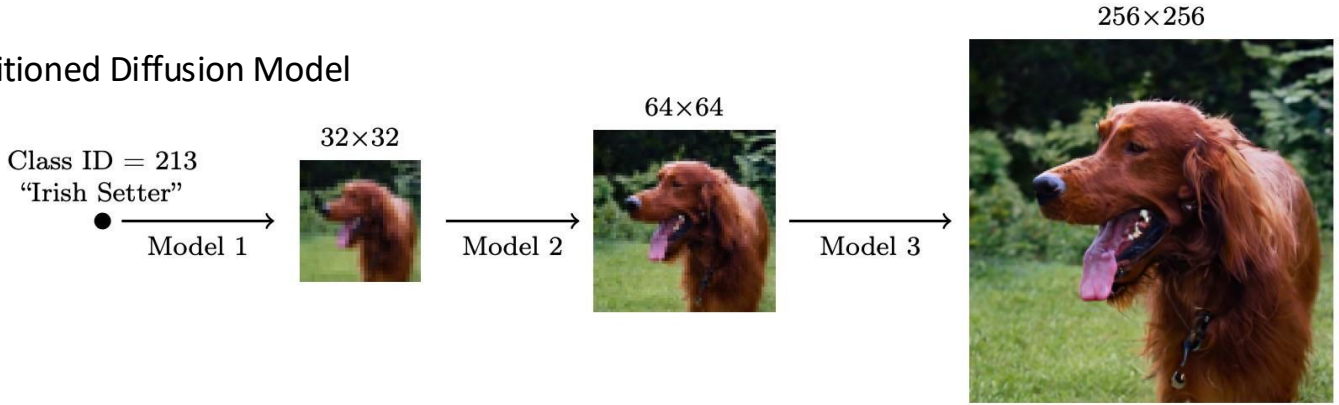
- APPLICATION:
  - Text to Image Generation
  - Image to Image Translation
  - Video Generation

Cascaded generation

Pipeline

Super-Resolution Diffusion Models

Class Conditioned Diffusion Model



Similar cascaded / multi-resolution image generation also exist in GAN (Big-GAN & StyleGAN)

Cascaded Diffusion Models outperform Big-GAN in FID and IS and VQ-VAE2 in Classification Accuracy Score.

Ho et al., "Cascaded Diffusion Models for High Fidelity Image Generation", 2021.

Noise conditioning augmentation

Reduce compounding error

Problem:

- During training super-resolution models are trained on original low-res images from the dataset.
- During inference, these low-res images are generated by class conditioned diffusion model, which has artifacts and poor quality than original low-res images used for training.

} Mismatch issue

Solution: Noise conditioning augmentation.

- During training, add varying amounts of Gaussian noise (or blurring by Gaussian kernel) to the low-res images.
- During inference, sweep over the optimal amount of noise added to the low-res images.
- BSR-degradation process: applies JPEG compressions noise, camera sensor noise, different image interpolations for downsampling, Gaussian blur kernels and Gaussian noise in a random order to an image.

Ho et al., "Cascaded Diffusion Models for High Fidelity Image Generation", 2021.
Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

Outline for today's class

- THEORY:
  - How do we condition diffusion model?
  - How do people sample in practice? – DDIM sampling
  - How do we generate high-resolution images?

- APPLICATION:
  - Text to Image Generation
  - Image to Image Translation
  - Video Generation

GLIDE
OpenAI

- A 64x64 base model + a 64x64 → 256x256 super-resolution model.
- Tried classifier-free and CLIP guidance. Classifier-free guidance works better than CLIP guidance.

"a hedgehog using a calculator"

"a corgi wearing a red bowtie and a purple party hat"

"robots meditating in a vipassana retreat"

"a fall landscape with a small cottage next to a lake"

Samples generated with classifier-free guidance (256x256)

Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

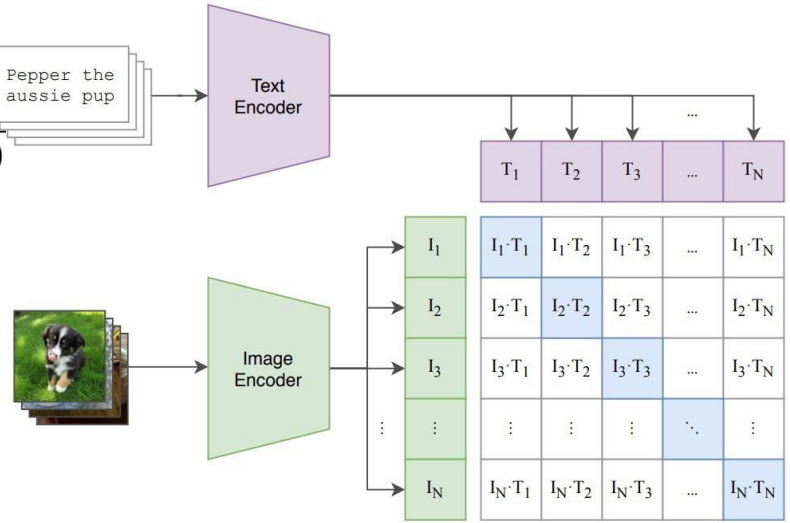
Carnegie Mellon University

CLIP guidance

What is a CLIP model?

- Trained by contrastive cross-entropy loss:

$$-\log \frac{\exp(f(\mathbf{x}_i)\cdot g(\mathbf{c}_j)/\tau)}{\sum_k \exp(f(\mathbf{x}_i)\cdot g(\mathbf{c}_k)/\tau)} - \log \frac{\exp(f(\mathbf{x}_i)\cdot g(\mathbf{c}_j)/\tau)}{\sum_k \exp(f(\mathbf{x}_k)\cdot g(\mathbf{c}_j)/\tau)}$$



- The optimal value of $f(\mathbf{x})\cdot g(\mathbf{c})$ is

$$\log \frac{p(\mathbf{x},\mathbf{c})}{p(\mathbf{x})p(\mathbf{c})} = \log p(\mathbf{c}|\mathbf{x}) - \log p(\mathbf{c})$$

Carnegie Mellon University

Radford et al., "Learning Transferable Visual Models From Natural Language Supervision", 2021.
Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

CLIP guidance

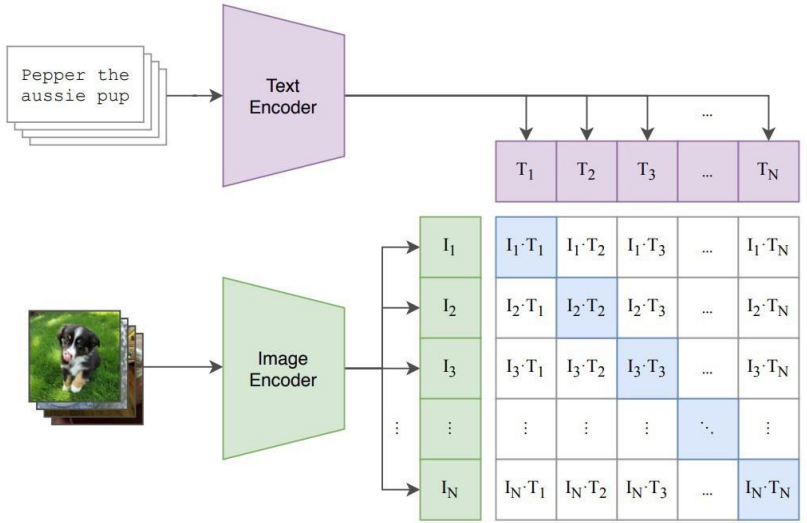
Replace the classifier in classifier guidance with a CLIP model

- Sample with a modified score:

$$\nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega \log p(\mathbf{c}|\mathbf{x}_t)]$$

$$= \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega(\underbrace{\log p(\mathbf{c}|\mathbf{x}_t) - \log p(\mathbf{c})}_{\text{CLIP model}})]$$

$$= \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega(f(\mathbf{x}_t) \cdot g(\mathbf{c}))]$$



Radford et al., "Learning Transferable Visual Models From Natural Language Supervision", 2021.
Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

Carnegie Mellon University

GLIDE

OpenAI

- Fine-tune the model especially for inpainting: feed randomly occluded images with an additional mask channel as the input.

"an old car in a snowy forest"

"a man wearing a white hat"

Text-conditional image inpainting examples

Nichol et al., "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models", 2021.

Carnegie Mellon University

DALL·E 2

OpenAI

a shiba inu wearing a beret and black turtleneck

a close up of a handpalm with leaves growing from it

1kx1k Text-to-image generation.

Outperform DALL-E (autoregressive transformer).

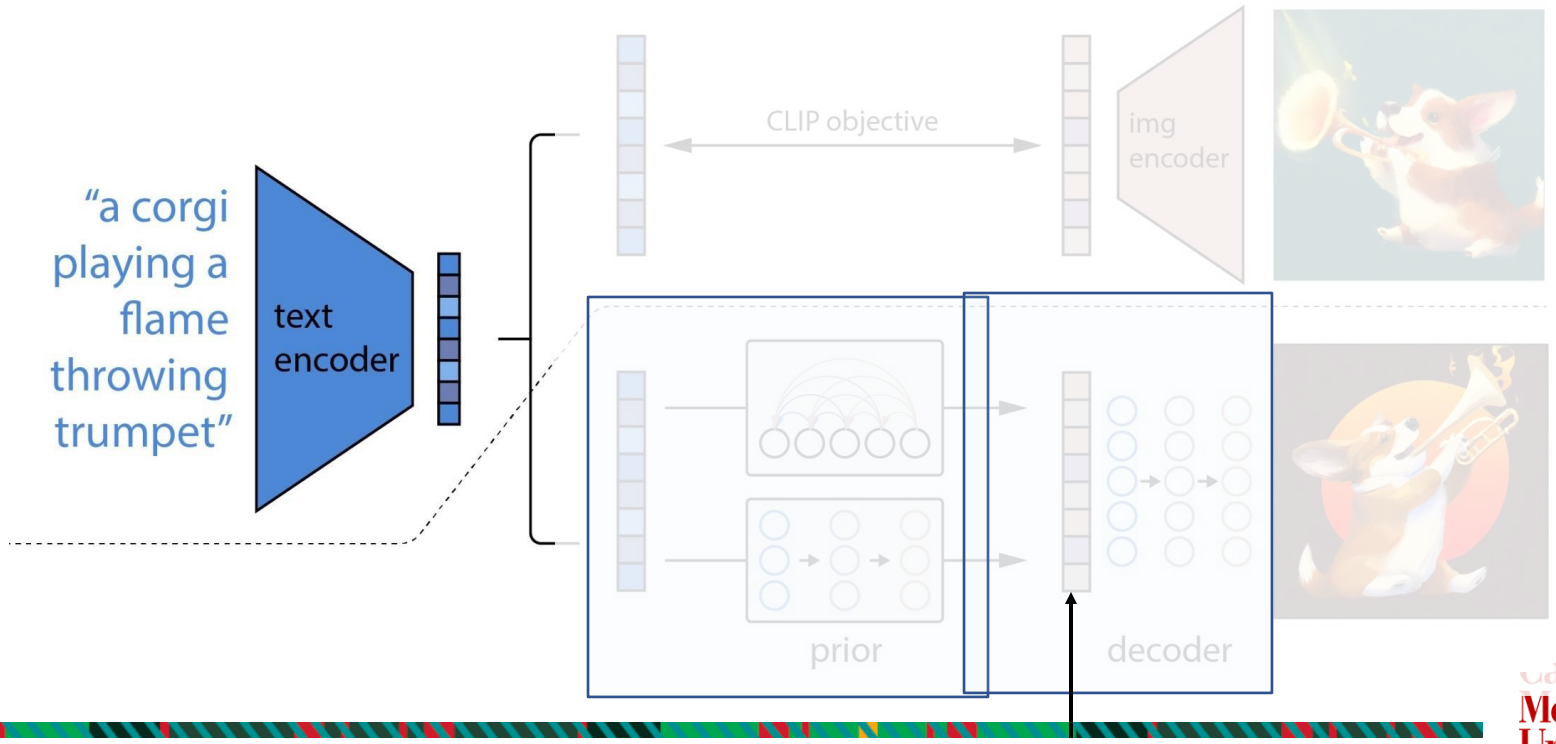
Ramesh et al., “Hierarchical Text-Conditional Image Generation with CLIP Latents”, arXiv 2022.

Carnegie Mellon University

DALL·E 2

Model components

Prior: produces CLIP image embeddings conditioned on the caption.
Decoder: produces images conditioned on CLIP image embeddings and text.

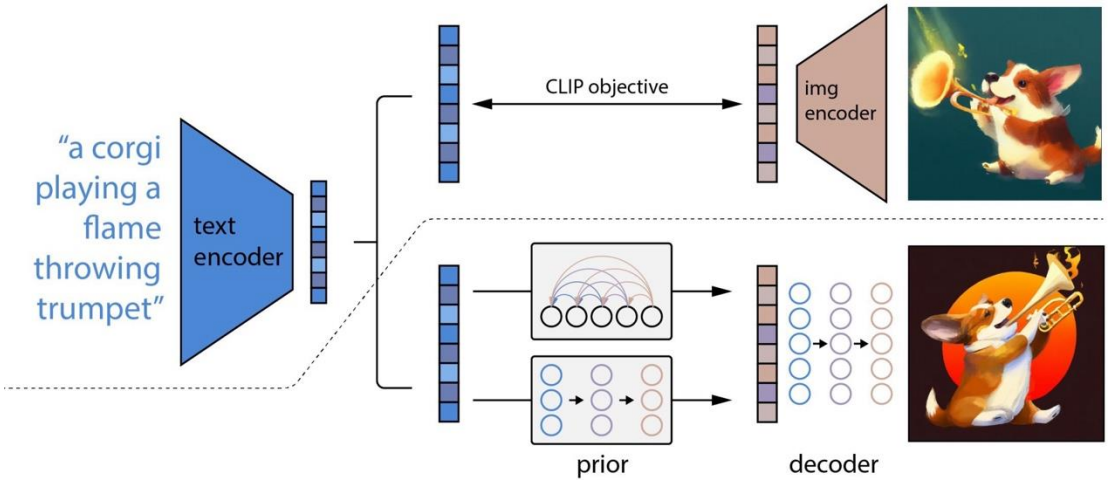


Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

Carnegie Mellon University

DALL·E 2

Model components



Why conditional on CLIP image embeddings?

CLIP image embeddings capture high-level semantic meaning.

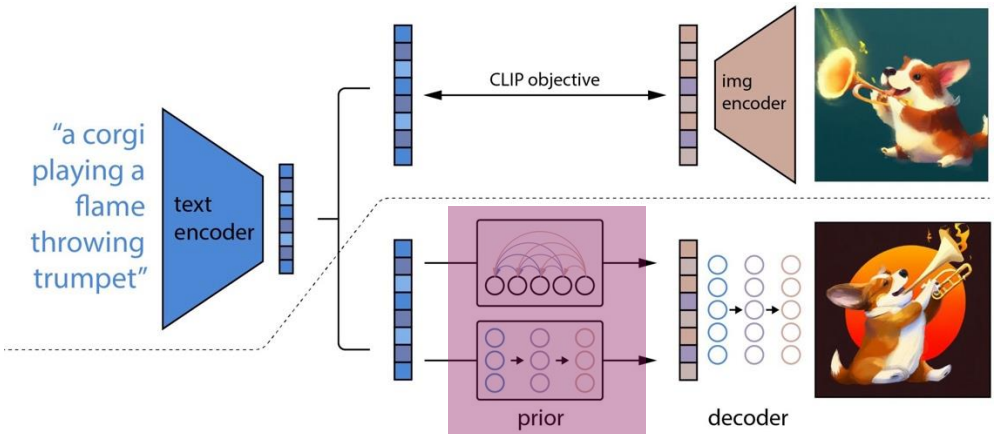
Latents in the decoder model take care of the rest.

The bipartite latent representation enables several text-guided image manipulation tasks.

Carnegie Mellon University

DALL·E 2

Model components (1/2): prior model



Prior: produces CLIP image embeddings conditioned on the caption.

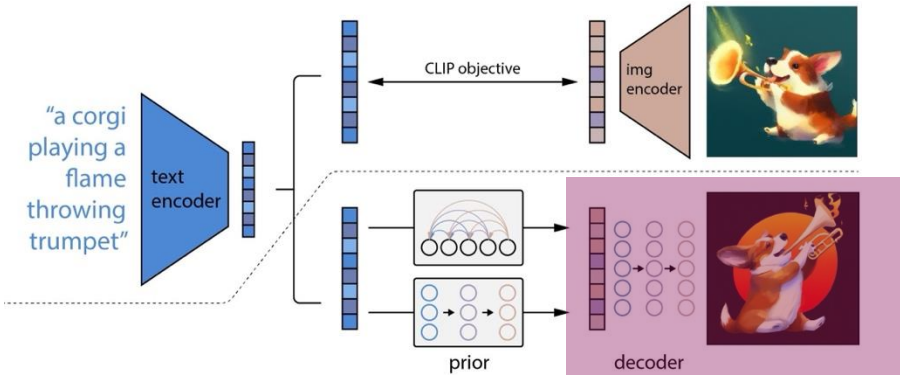
- Option 1. autoregressive prior: quantize image embedding to a seq. of discrete codes and predict them autoregressively.
- Option 2. diffusion prior: model the continuous image embedding by diffusion models conditioned on caption.

Carnegie Mellon University

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

DALL·E 2

Model components (2/2): decoder model



Decoder: produces images conditioned on CLIP image embeddings (and text).

- Cascaded diffusion models: 1 base model (64x64), 2 super-resolution models (64x64 → 256x256, 256x256 → 1024x1024).
- Largest super-resolution model is trained on patches and takes full-res inputs at inference time.
- Classifier-free guidance & noise conditioning augmentation are important.

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

Carnegie Mellon University

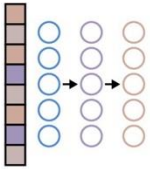
DALL·E 2

Bipartite latent representations

Bipartite latent representations $(\mathbf{z}, \mathbf{x}_T)$

$\mathbf{z}$: CLIP image embeddings

$\mathbf{x}_T$: inversion of DDIM sampler
(latents in the decoder model)



decoder

Near exact
reconstruction

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

Carnegie Mellon University

DALL·E 2

Image variations

Fix the CLIP embedding $\mathbf{z}$.

Decode using different decoder latents $\mathbf{x}_T$

DALL·E 2

Image interpolation

Interpolate image CLIP embeddings $\mathbf{z}_i$.

Use different $\mathbf{x}_T$ to get different interpolation trajectories.

Carnegie Mellon University

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", arXiv 2022.

DALL·E 2

Text Diffs

a photo of a cat → an anime drawing of a super saiyan cat, artstation

a photo of a victorian house → a photo of a modern house

a photo of an adult lion → a photo of lion cub

Change the image CLIP embedding towards the difference of the text CLIP embeddings of two prompts.

Decoder latent is kept as a constant.

Carnegie Mellon University

Imagen

Google Research, Brain team

Input: text; Output: 1kx1k images

- An unprecedented degree of photorealism
  - SOTA automatic scores & human ratings
- A deep level of language understanding
- Extremely simple
  - no latent space, no quantization

A brain riding a rocketship heading towards the moon.

Carnegie Mellon University

Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

Imagen

Google Research, Brain team

A photo of a Shiba Inu dog with a backpack riding a bike. It is wearing sunglasses and a beach hat.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Carnegie Mellon University

Imagen

Google Research, Brain team

A dragon fruit wearing karate belt in the snow.

Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

Carnegie Mellon University

Imagen

Google Research, Brain team

A relaxed garlic with a blindfold reading a newspaper while floating in a pool of tomato soup.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Carnegie Mellon University

Imagen

Google Research, Brain team

A cute hand-knitted koala wearing a sweater with 'CVPR' written on it.

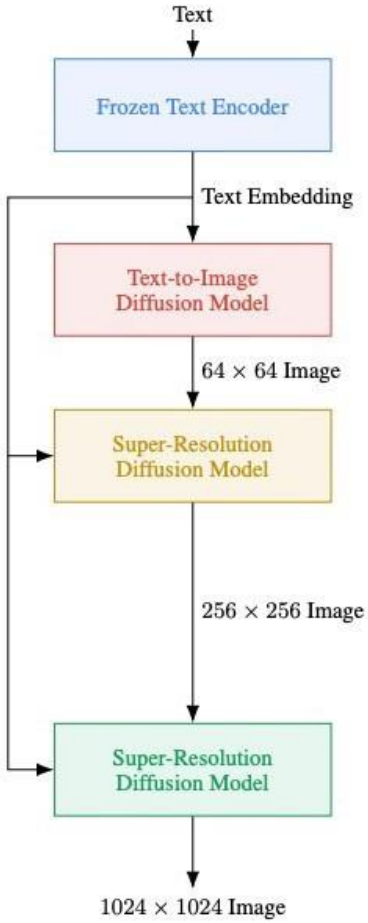
Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

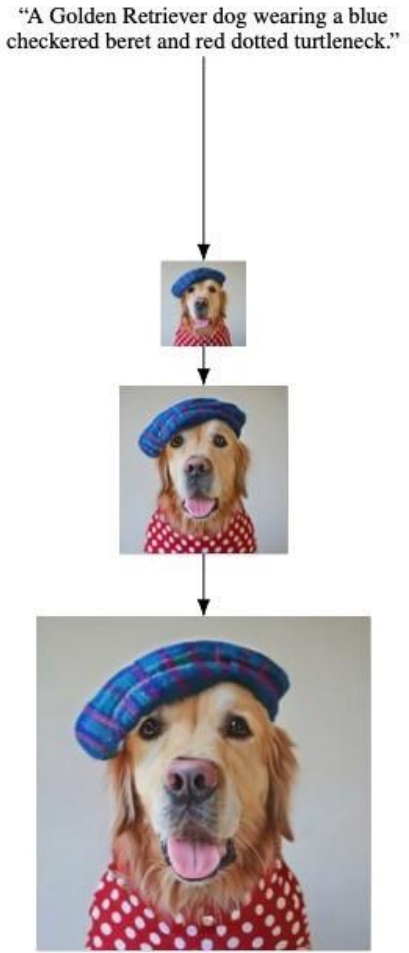
Carnegie Mellon University

Imagen

Key modeling components:

- Cascaded diffusion models
- Classifier-free guidance and dynamic thresholding.
- Frozen large pretrained language models as text encoders. (T5-XXL)



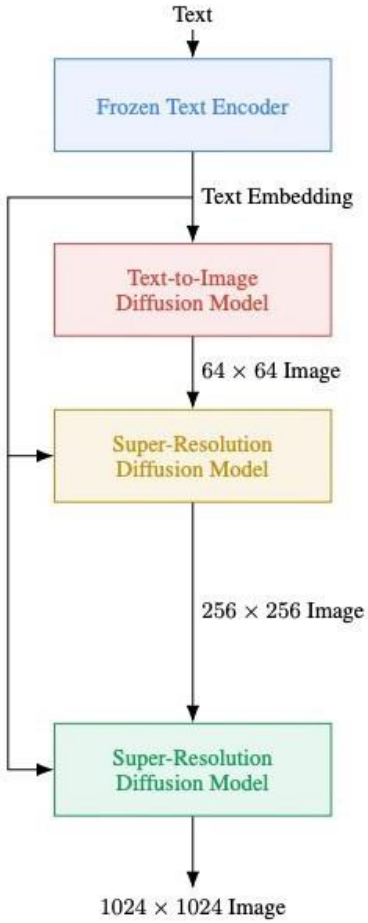


Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen

Key observations:

- Beneficial to use text conditioning for all super-res models.
  - Noise conditioning augmentation weakens information from low-res models, thus needs text conditioning as extra information input.
- Scaling text encoder is extremely efficient.
  - More important than scaling diffusion model size.
- Human raters prefer T5-XXL as the text encoder over CLIP encoder on DrawBench.

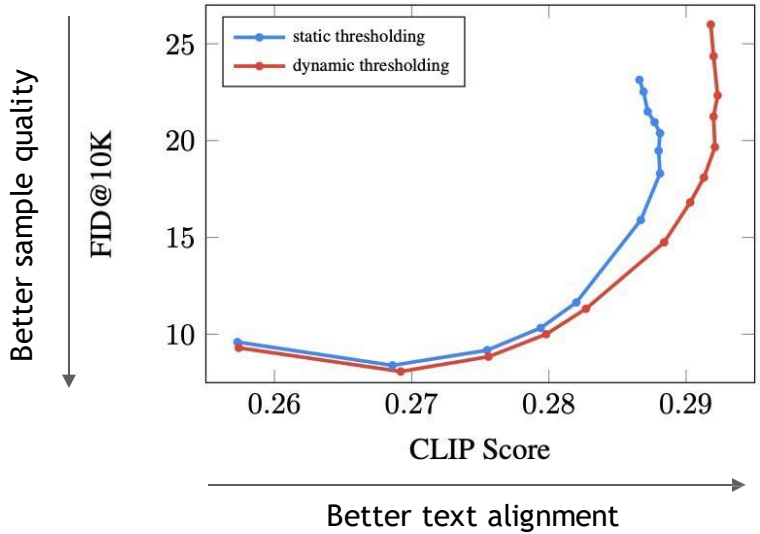


Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Imagen

Dynamic thresholding

- Large classifier-free guidance weights → better text alignment, worse image quality



Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Carnegie Mellon University

Imagen

Dynamic thresholding

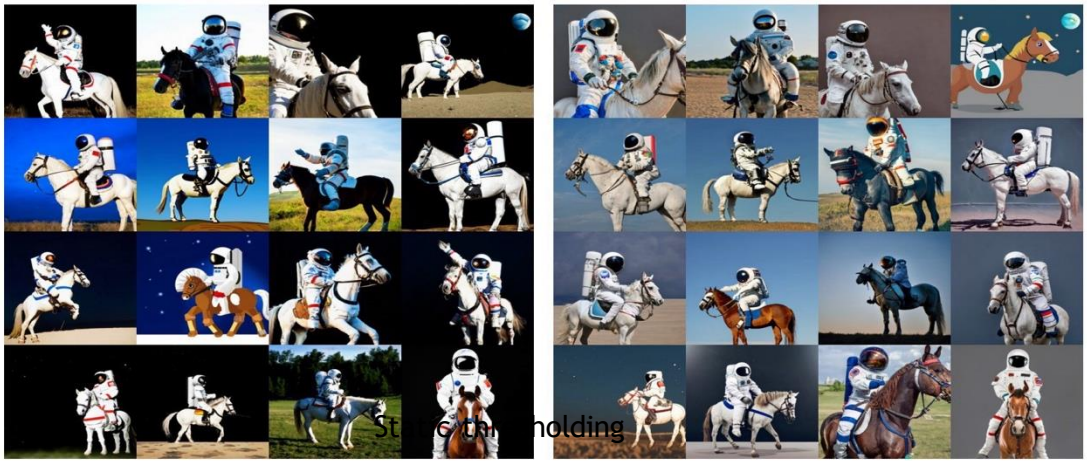
- Large classifier-free guidance weights → better text alignment, worse image quality
- Hypothesis : at large guidance weight, the generated images are saturated due to the very large gradient updates during sampling
- Solution – dynamic thresholding: adjusts the pixel values of samples at each sampling step to be within a dynamic range computed over the statistics of the current samples.

Imagen

Dynamic thresholding

- Large classifier-free guidance weights → better text alignment, worse image quality
- Hypothesis: at high guidance weight, the generated images are saturated due to the very large gradient updates during sampling

- Solution – dynamitic thresholding: adjusts the pixel values of samples at each sampling step to be within a dynamic range computed over the statistics of the current samples.



Static thresholding

Dynamic thresholding

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Carnegie Mellon University

Imagen

DrawBench: new benchmark for text-to-image evaluations

- A set of 200 prompts to evaluate text-to-image models across multiple dimensions.
    - E.g., the ability to faithfully render different colors, numbers of objects, spatial relations, text in the scene, unusual interactions between objects.
    - Contains complex prompts, e.g, long and intricate descriptions, rare words, misspelled prompts.

Saharia et al., “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, arXiv 2022.

Carnegie Mellon University

Imagen

DrawBench: new benchmark for text-to-image evaluations

to evaluate text-to-image models across multiple dimensions.

ability to faithfully render different colors, numbers of objects, spatial relations, text in the s
ns between objects.


A brown bird and a blue bear.


One cat and two dogs sitting on the grass.

A sign that says 'NeurIPS'.

ompts.


A small blue book sitting on a large red book.


A blue coloured pizza.


A wine glass on top of a dog.


A pear cut into seven pieces arranged in a ring.


A photo of a confused grizzly bear in calculus class.


A small vessel propelled on water by oars, sails, or an engine.

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Carnegie Mellon University

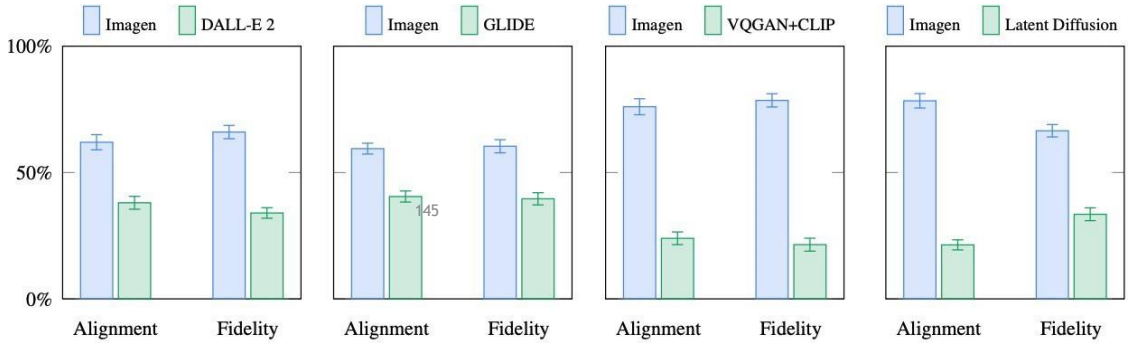
Imagen

Evaluations

Imagen got SOTA automatic evaluation scores on COCO dataset

Model	FID-30K	Zero-shot FID-30K
AttnGAN [76]	35.49	
DM-GAN [83]	32.64	
DF-GAN [69]	21.42	
DM-GAN + CL [78]	20.79	
XMC-GAN [81]	9.33	
LAFITE [82]	8.12	
Make-A-Scene [22]	7.55	
DALL-E [53]		17.89
LAFITE [82]		26.94
GLIDE [41]		12.24
DALL-E 2 [54]		10.39
Imagen (Our Work)		**7.27**

Imagen is preferred over recent work by human raters in sample quality & image-text alignment on DrawBench.



Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", arXiv 2022.

Carnegie Mellon University

Stable Diffusion

Latest & Publicly available text-to-image generation

To be discussed in detail in paper presentation

High-Resolution Image Synthesis with Latent Diffusion Models

Robin Rombach*, Andreas Blattmann*, Dominik Lorenz, Patrick Esser, Björn Ommer

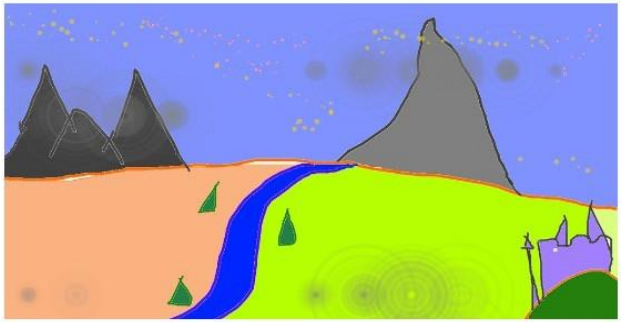
CVPR '22 Oral | GitHub | arXiv | Project page

Stable Diffusion is a latent text-to-image diffusion model. Thanks to a generous compute donation from Stability AI and support from LAION, we were able to train a Latent Diffusion Model on 512x512 images from a subset of the LAION-5B database. Similar to Google's Imagen, this model uses a frozen CLIP ViT-L/14 text encoder to condition the model on text prompts. With its 860M UNet and 123M text encoder, the model is relatively lightweight and runs on a GPU with at least 10GB VRAM. See this section below and the model card.

Stable Diffusion

Latest & Publicly available text-to-image generation

Input

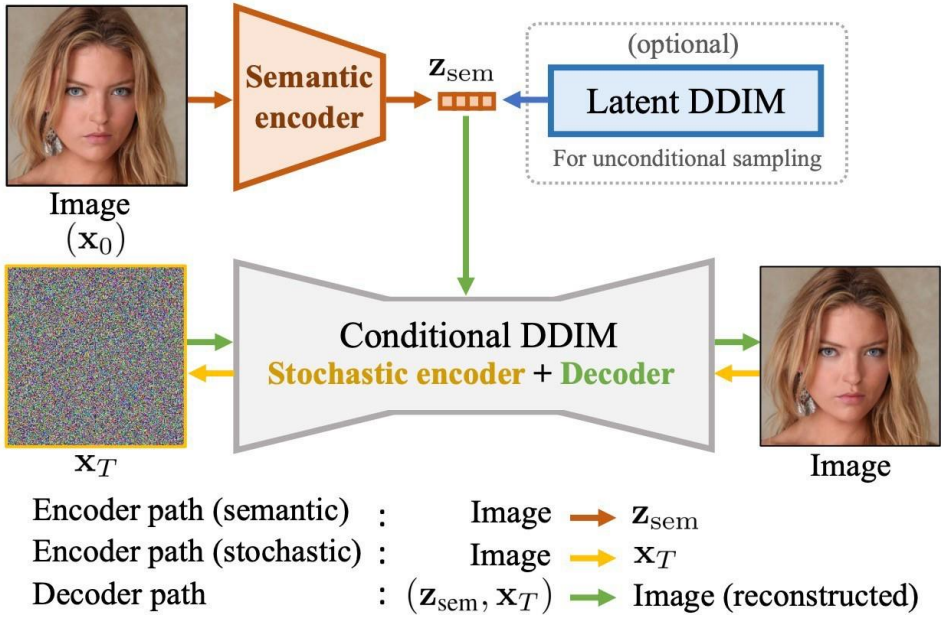


Outputs

Carnegie Mellon University

Diffusion Autoencoders

Learning semantic meaningful latent representations in diffusion models

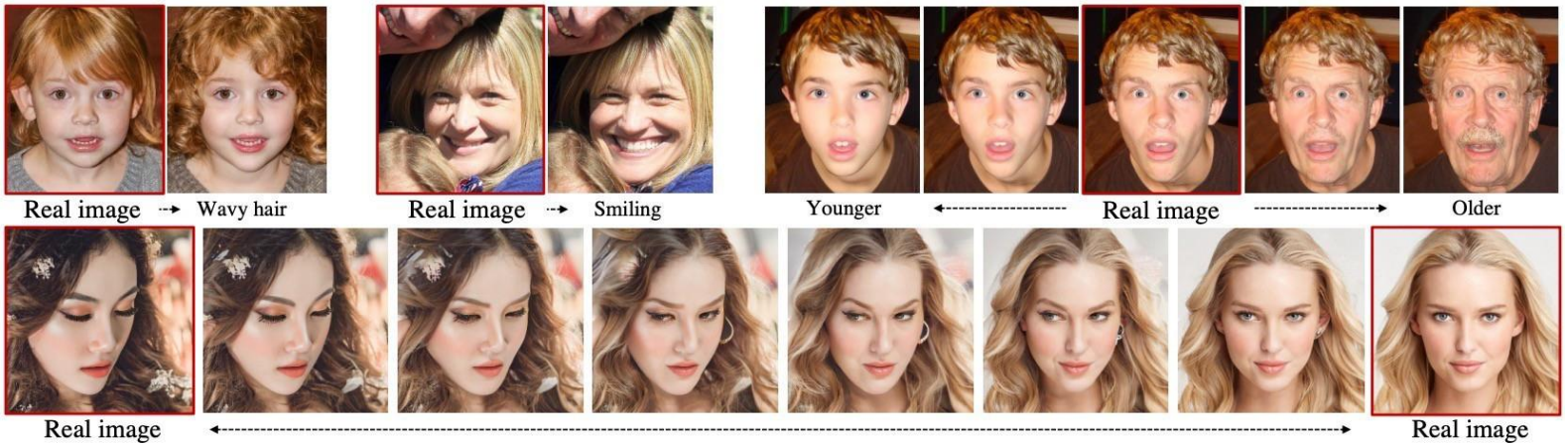


Preechakul et al., "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation", CVPR 2022.

Carnegie Mellon University

Diffusion Autoencoders

Learning semantic meaningful latent representations in diffusion models



Changing the semantic latent $\mathbf{z}_{\text{sem}}$

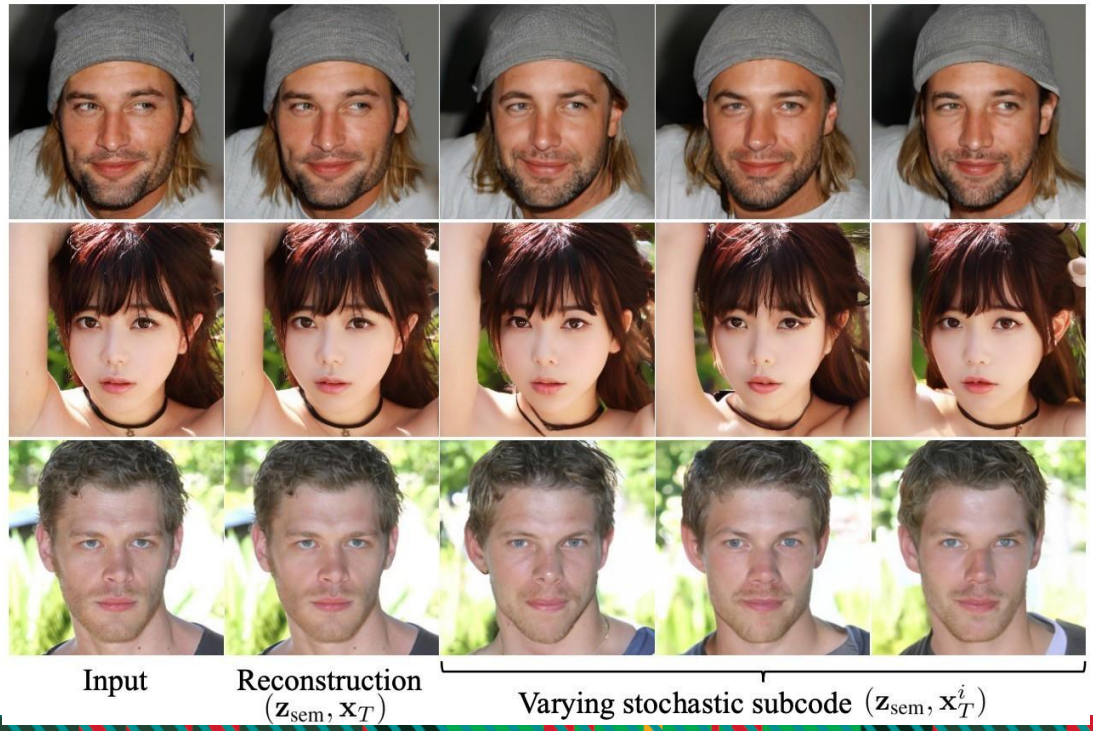
Very similar to StyleGAN based editing. Z_{sem} is the latent representation similar to the W/W+ space of StyleGAN

Preechakul et al., "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation", CVPR 2022.

Carnegie Mellon University

Diffusion Autoencoders

Learning semantic meaningful latent representations in diffusion models



Preechakul et al., "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation", CVPR 2022.

Carnegie Mellon University

Super-Resolution

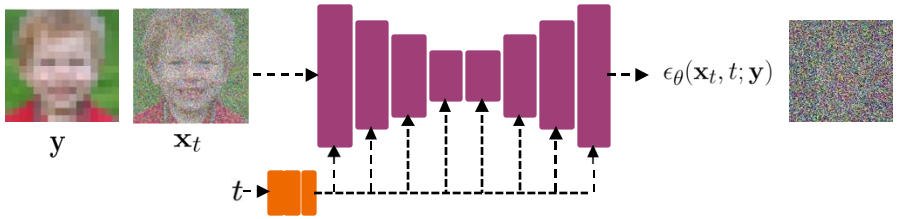
Super-Resolution via Repeated Refinement (SR3)

Image super-resolution can be considered as training $p(\mathbf{x}|\mathbf{y})$ where $\mathbf{y}$ is a low-resolution image and $\mathbf{x}$ is the corresponding high-resolution image

Train a score model for $\mathbf{x}$ conditioned on $\mathbf{y}$ using:

$$\mathbb{E}_{\mathbf{x},\mathbf{y}}\, \mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\, \mathbb{E}_t\, ||\epsilon_\theta(\mathbf{x}_t, t; \mathbf{y}) - \epsilon||_p^p$$

The conditional score is simply a U-Net with $\mathbf{x_t}$ and $\mathbf{y}$ (resolution image) concatenated.

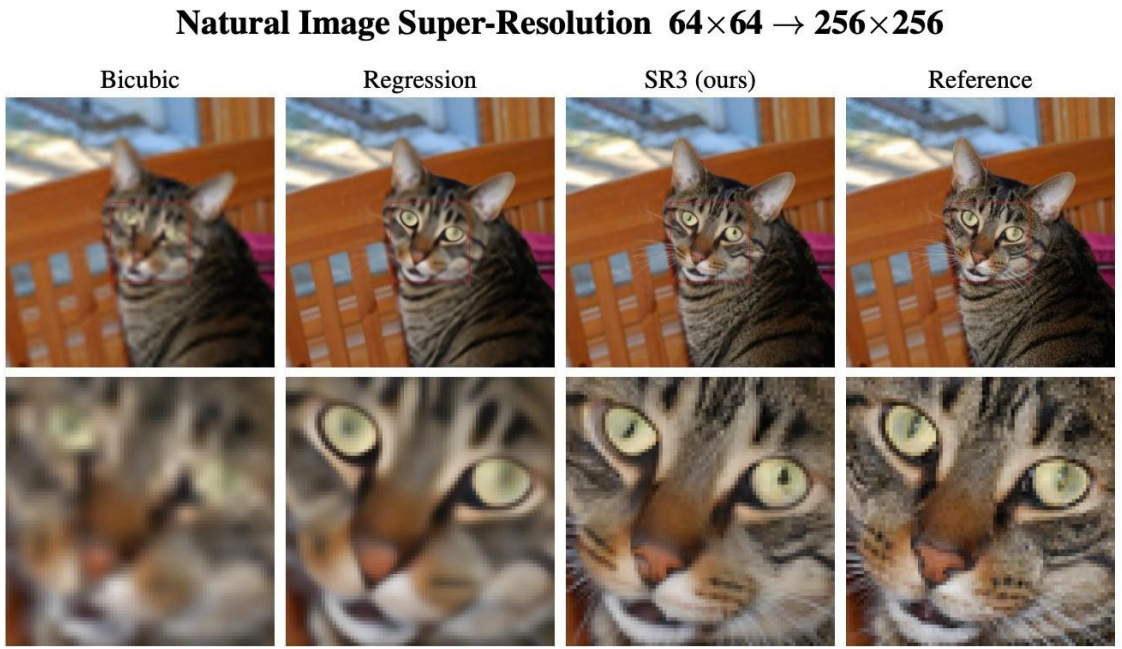


Saharia et al., Image Super-Resolution via Iterative Refinement, 2021

Carnegie Mellon University

Super-Resolution

Super-Resolution via Repeated Refinement (SR3)



Saharia et al., Image Super-Resolution via Iterative Refinement, 2021

Carnegie Mellon University

Image-to-Image Translation

Palette: Image-to-Image Diffusion Models

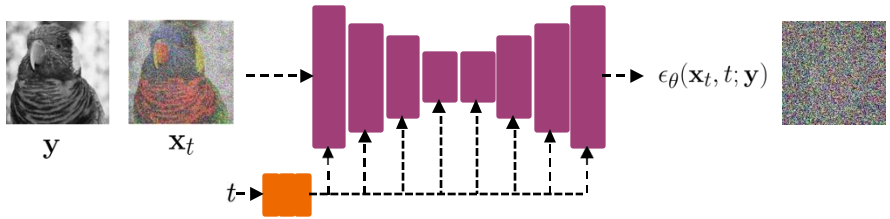
Many image-to-image translation applications can be considered as training $p(\mathbf{x}|\mathbf{y})$ where $\mathbf{y}$ is the input image.

For example, for colorization, $\mathbf{x}$ is a colored image and $\mathbf{y}$ is a gray-level image.

Train a score model for $\mathbf{x}$ conditioned on $\mathbf{y}$ using:

$$\mathbb{E}_{\mathbf{x},\mathbf{y}}\, \mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\, \mathbb{E}_t\, ||\epsilon_\theta(\mathbf{x}_t, t; \mathbf{y}) - \epsilon||_p^p$$

The conditional score is simply a U-Net with $\mathbf{x_t}$ and $\mathbf{y}$ concatenated.

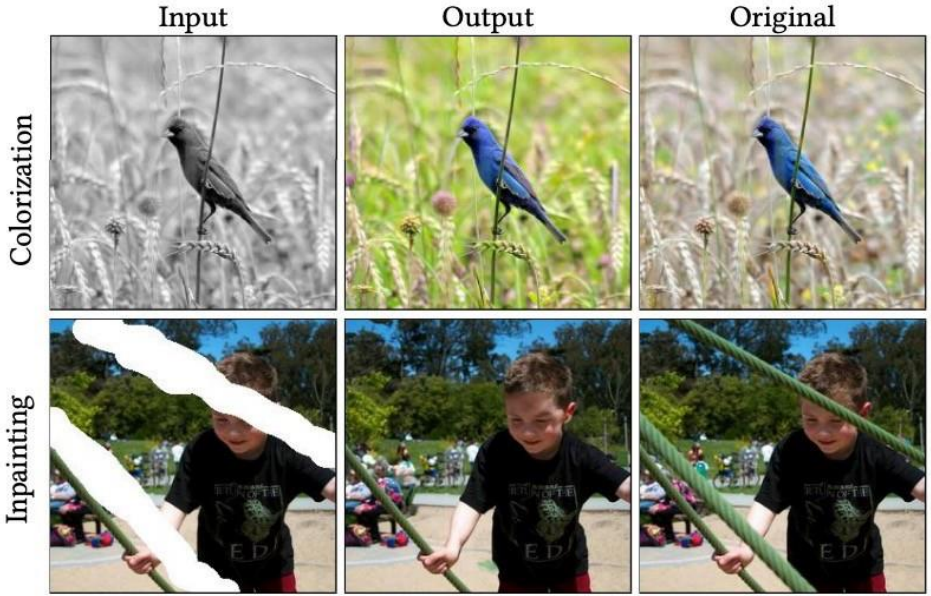


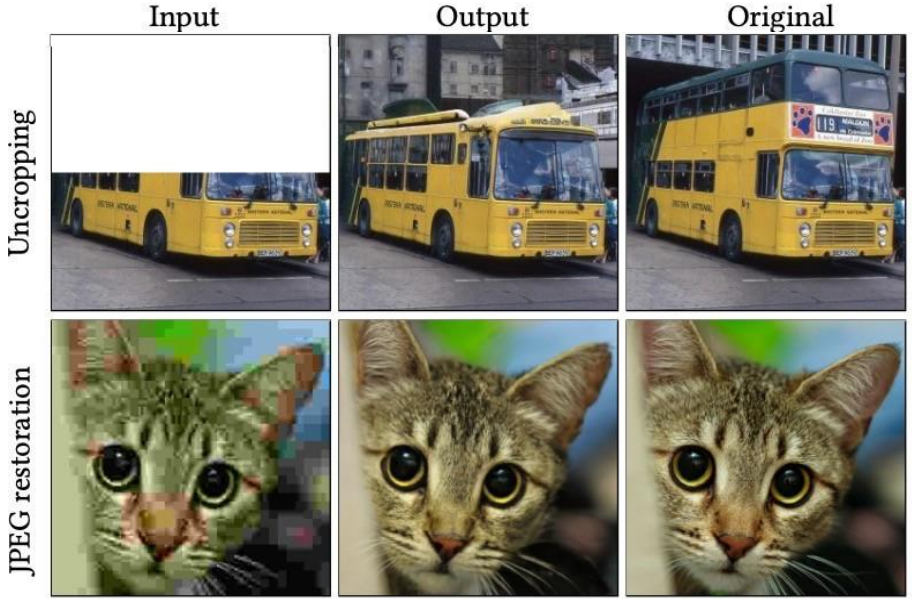
Saharia et al., Palette: Image-to-Image Diffusion Models, 2022

Carnegie Mellon University

Image-to-Image Translation

Palette: Image-to-Image Diffusion Models





Saharia et al., Palette: Image-to-Image Diffusion Models, 2022

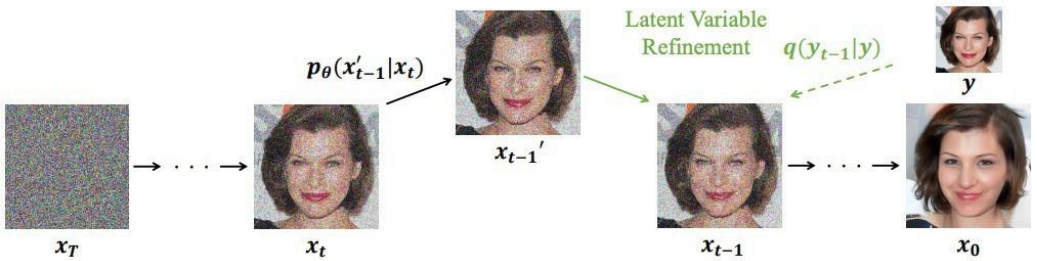
Carnegie Mellon University

Conditional Generation

Iterative Latent Variable Refinement (ILVR)

A simple technique to guide the generation process of an unconditional diffusion model using a reference image.

Given the conditioning (reference) image $\mathbf{y}$ the generation process is modified to pull the samples towards the reference image.



for $t = T, ..., 1$ **do**
  $\mathbf{z} \sim N(\mathbf{0}, \mathbf{I})$
  $x'_{t-1} \sim p_\theta(x'_{t-1}|x_t)$ ▷ unconditional proposal
  $y_{t-1} \sim q(y_{t-1}|y)$ ▷ condition encoding
  $x_{t-1} \leftarrow \phi_N(y_{t-1}) + x'_{t-1} - \phi_N(x'_{t-1})$
end for

Low-pass filter

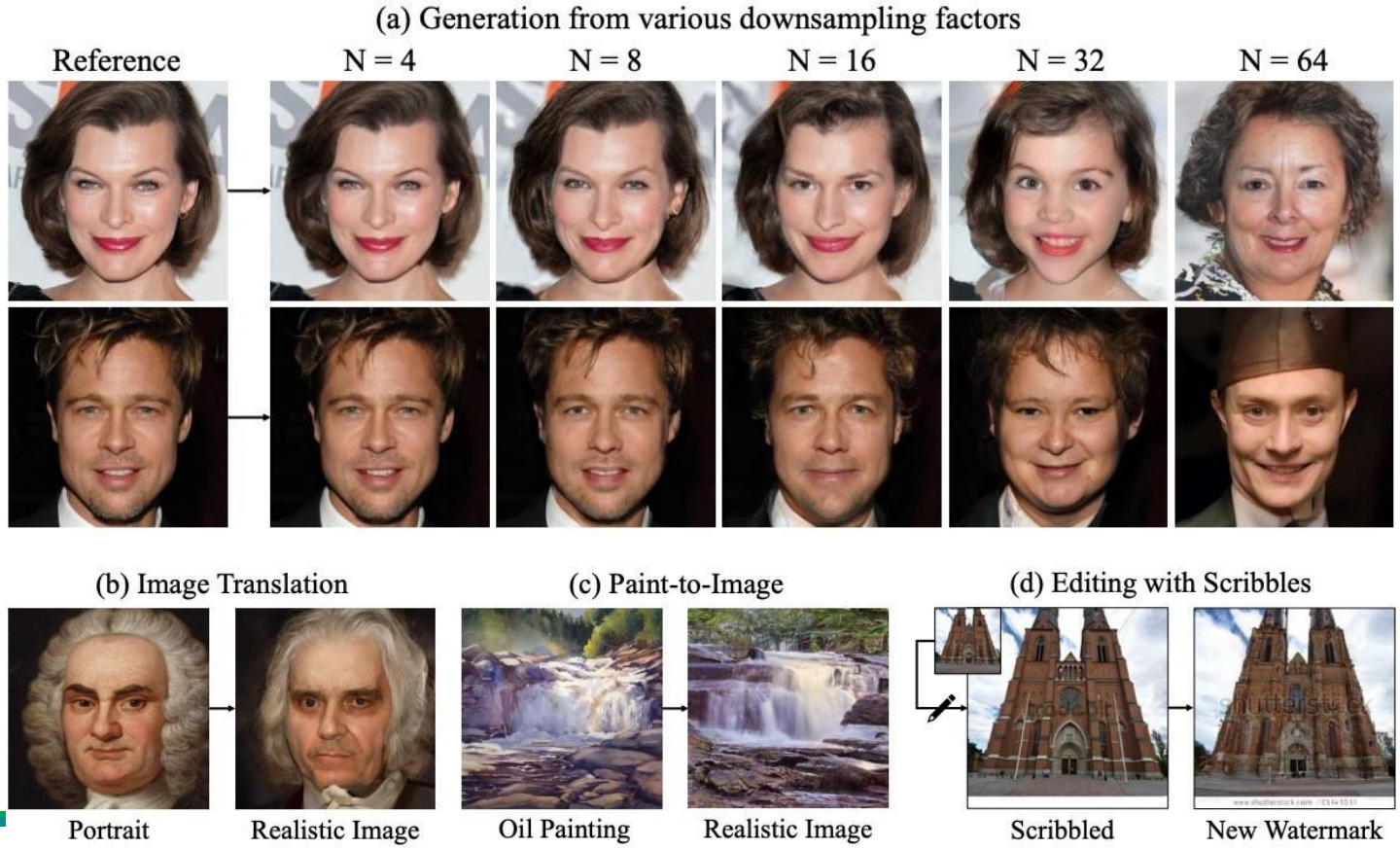
Downsampling / Upsampling by a factor of N

Choi et al., ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models, ICCV 2021

Carnegie Mellon University

Conditional Generation

Iterative Latent Variable Refinement (ILVR)



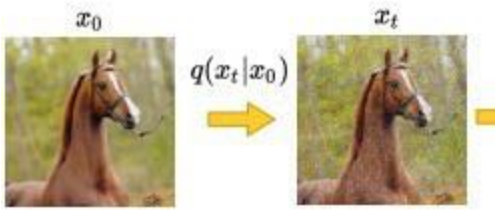
Choi et al., ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models, ICCV 2021

Carnegie Mellon University

Semantic Segmentation

Label-efficient semantic segmentation with diffusion models

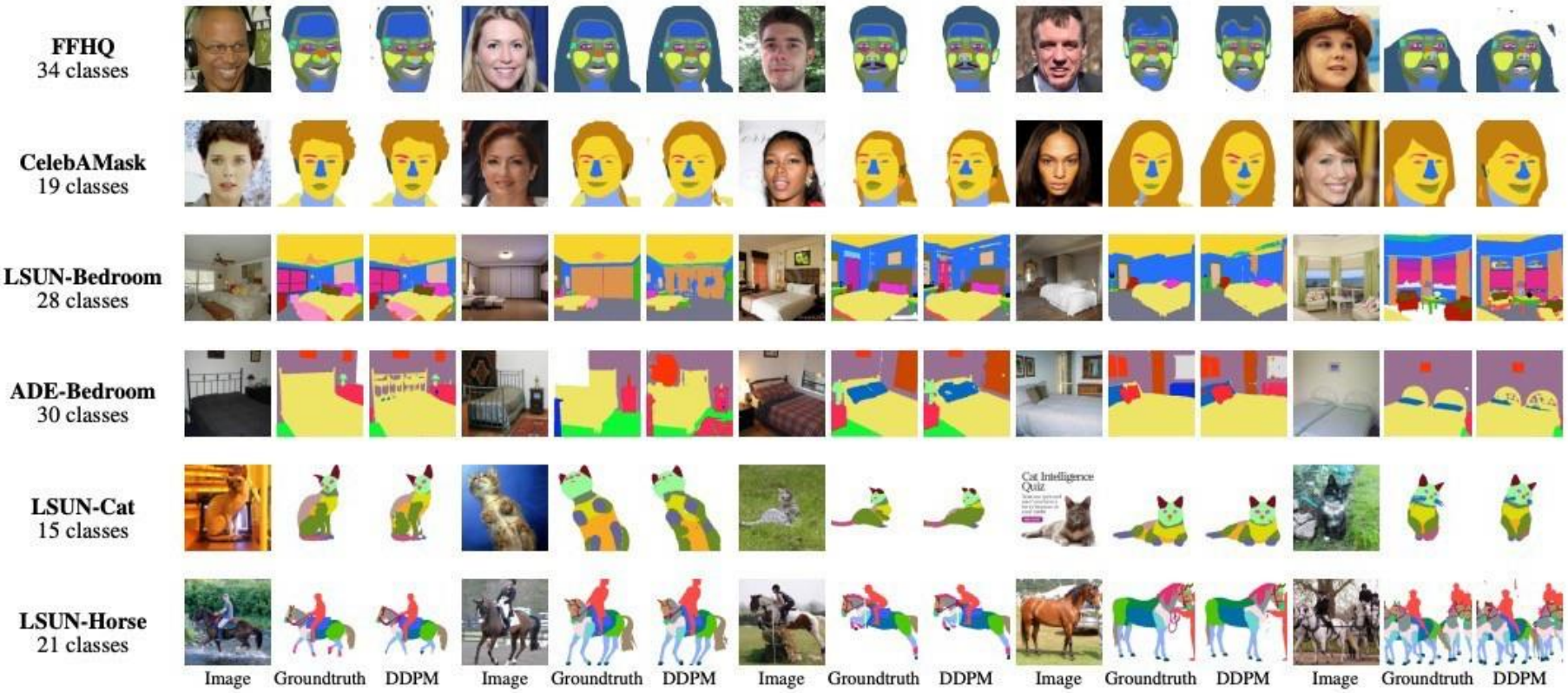
Can we use representation learned from diffusion models for downstream applications such as semantic segmentation?



Semantic Segmentation

Label-efficient semantic segmentation with diffusion models

The experimental results show that the proposed method outperforms Masked Autoencoders, GAN and VAE-based models.



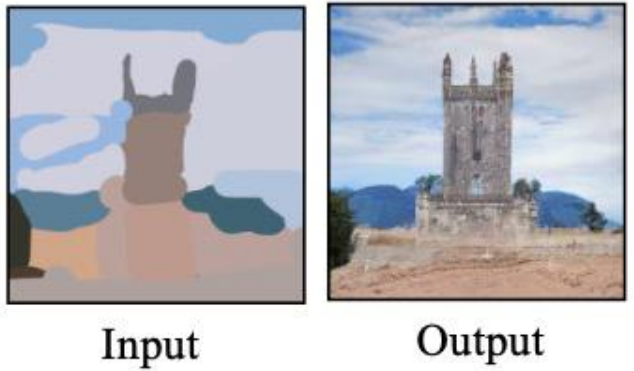
Baranchuk et al., Label-Efficient Semantic Segmentation with Diffusion Models, ICLR 2022

University

Image Editing

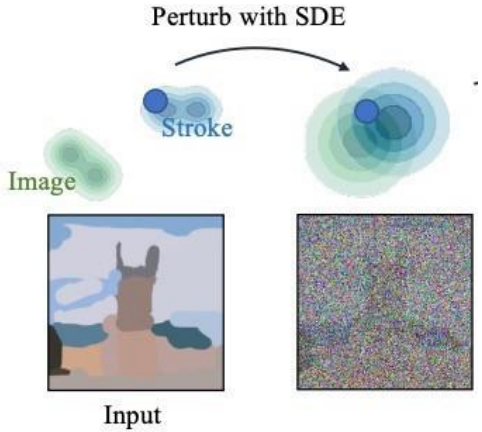
SDEdit

Goal: Given a stroke painting with color, generate a photorealistic image



Key Idea:

- Latent Distribution of stroke and real images do not overlap.
- But once we apply forward diffusion on them, their distribution start overlapping as finally it becomes gaussian noise.

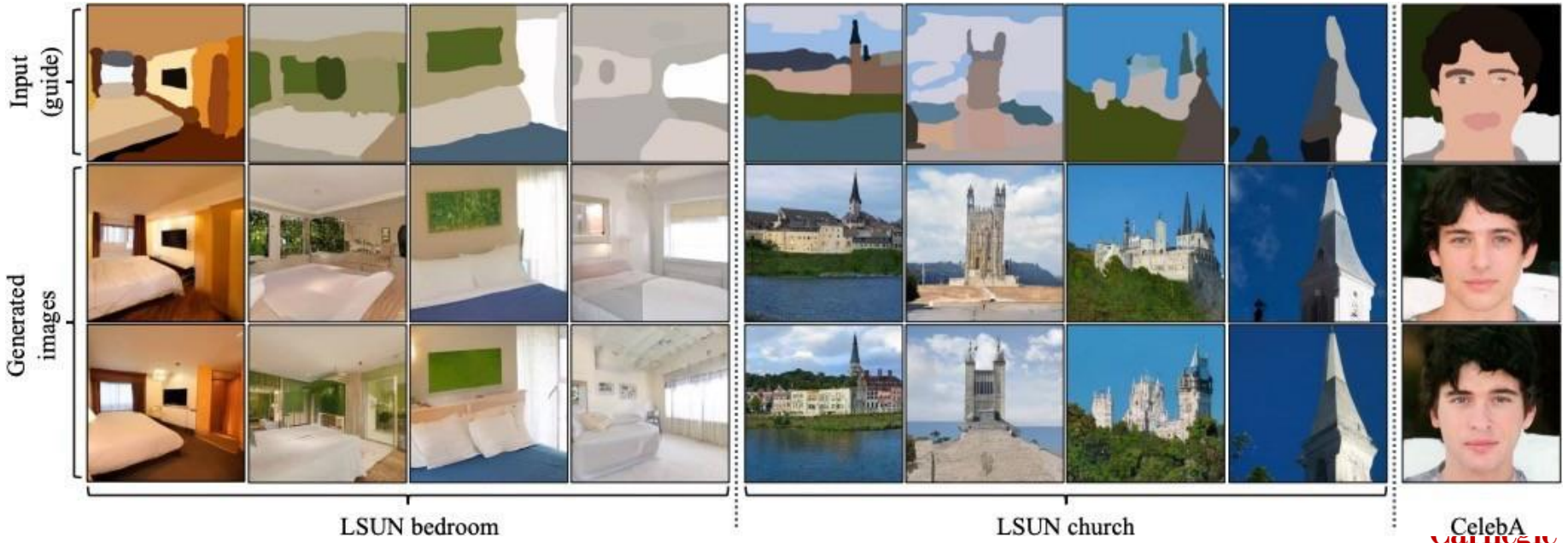


Meng et al., SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations, ICLR 2022

Carnegie Mellon University

Image Editing

SDEdit



Meng et al., SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations, ICLR 2022

Carnegie Mellon University

Video Generation

Samples from a text-conditioned video diffusion model, conditioned on the string ***fireworks***.

Ho et al., "Video Diffusion Models", *arXiv*, 2022
Harvey et al., "Flexible Diffusion Modeling of Long Videos", arXiv, 2022
Yang et al., "Diffusion Probabilistic Modeling for Video Generation", arXiv, 2022
Höppe et al., "Diffusion Models for Video Prediction and Infilling", arXiv, 2022
Voleti et al., "MCVD: Masked Conditional Video Diffusion for Prediction, Generation, and Interpolation", *arXiv*, 2022

Carnegie Mellon University

Video Generation

Video Generation Tasks:

- Unconditional Generation (Generate all frames)
- Future Prediction (Generate future from past fames)
- Past Prediction (Generate past from future fames)
- Interpolation (Generate intermediate frames)

➡ Learn a model of the form:

$$p_{\boldsymbol{\theta}}\left(\mathbf{x}^{t_1}, \cdots, \mathbf{x}^{t_K} \middle| \mathbf{x}^{\tau_1}, \cdots, \mathbf{x}^{\tau_M}\right)$$

Given frames: $\mathbf{x}^{\tau_1}, \cdots, \mathbf{x}^{\tau_M}$

Frames to be predicted: $\mathbf{x}^{t_1}, \cdots, \mathbf{x}^{t_K}$

Ho et al., "Video Diffusion Models", *arXiv*, 2022

Harvey et al., "Flexible Diffusion Modeling of Long Videos", arXiv, 2022

Yang et al., "Diffusion Probabilistic Modeling for Video Generation", *arXiv*, 2022

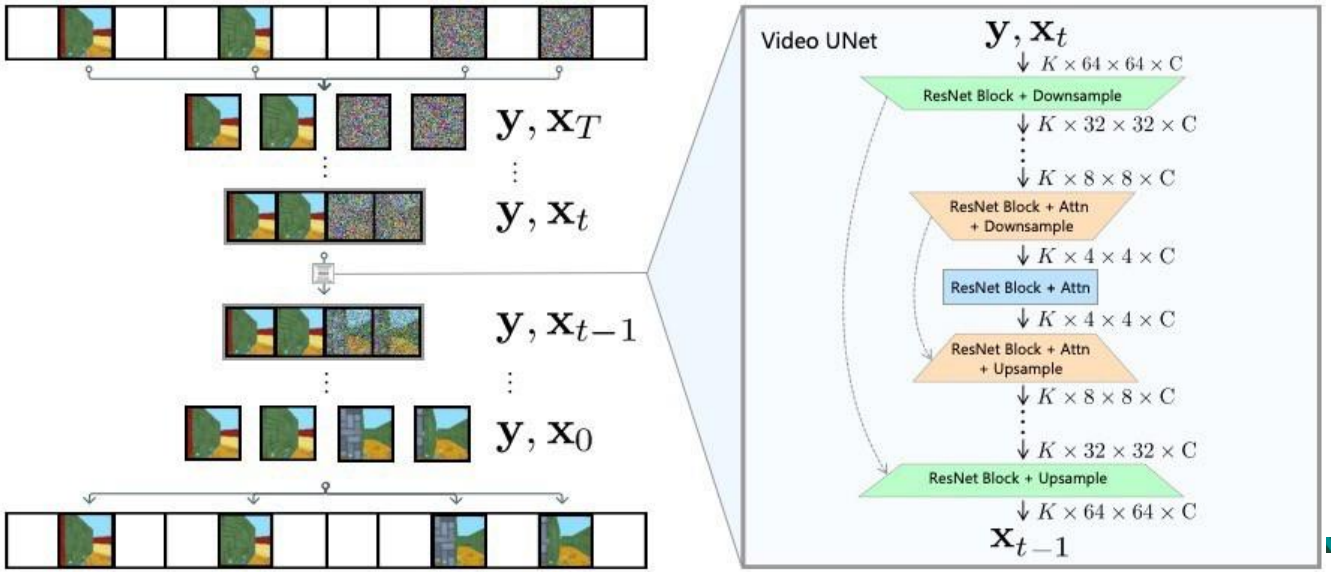
Höppe et al., "Diffusion Models for Video Prediction and Infilling", arXiv, 2022

Voleti et al., "MCVD: Masked Conditional Video Diffusion for Prediction, Generation, and Interpolation", *arXiv*, 2022

Video Generation

Learn one model for everything:

- Architecture as **one diffusion model** over **all frames concatenated**.
- Mask frames to be predicted; provide conditioning frames; vary applied masking/conditioning for different tasks during training.
- Use **time position encodings** to encode times.



(image from: Harvey et al., "Flexible Diffusion Modeling of Long Videos", *arXiv*, 2022)

Carnegie Mellon University

Video Generation

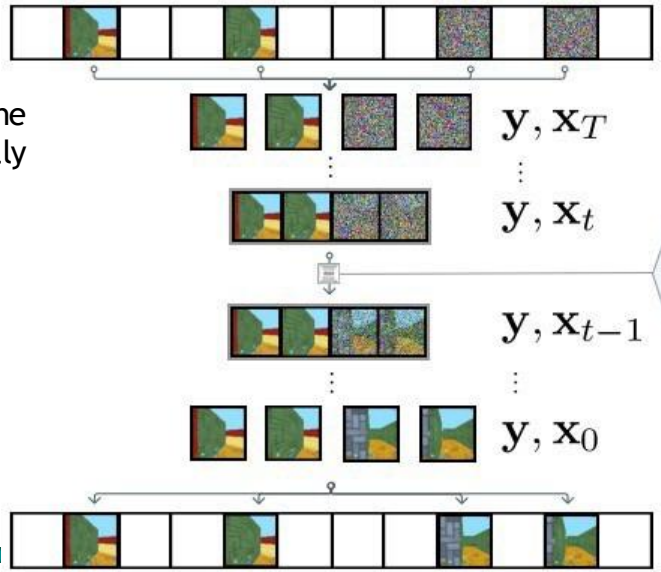
Architecture Details:

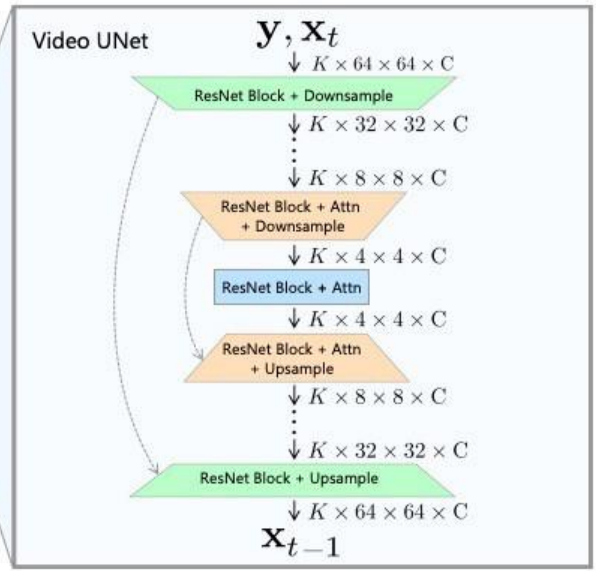
Data is 4D (image height, image width, #frames, channels)

- Option (1): 3D Convolutions. Can be computationally expensive.
- Option (2): Spatial 2D Convolutions + Attention Layers along frame axis.

➡ Additional Advantage:

Ignoring the attention layers, the model can be trained additionally on pure image data!





(image from: Harvey et al., "Flexible Diffusion Modeling of Long Videos", *arXiv*, 2022)

Video Generation

Results

Long term video generation in hierarchical manner:

- 1. Generate future frames in sparse manner, conditioning on frames far back
- 2. Interpolate in-between frames

1+ hour coherent video generation possible!

Test Data:

Generated:

(video from: Harvey et al., “Flexible Diffusion Modeling of Long Videos”, *arXiv*, 2022, https://plai.cs.ubc.ca/2022/05/20/flexible-diffusion-modeling-of-long-videos/)

Make-A-Video

Make-A-Video research builds on the recent progress made in text-to-image generation technology built to enable text-to-video generation. The system uses images with descriptions to learn what the world looks like and how it is often described. It also uses unlabeled videos to learn how the world moves. With this data, Make-A-Video lets you bring your imagination to life by generating whimsical, one-of-a-kind videos with just a few words or lines of text.

A dog wearing a Superhero outfit with red cape flying through the sky

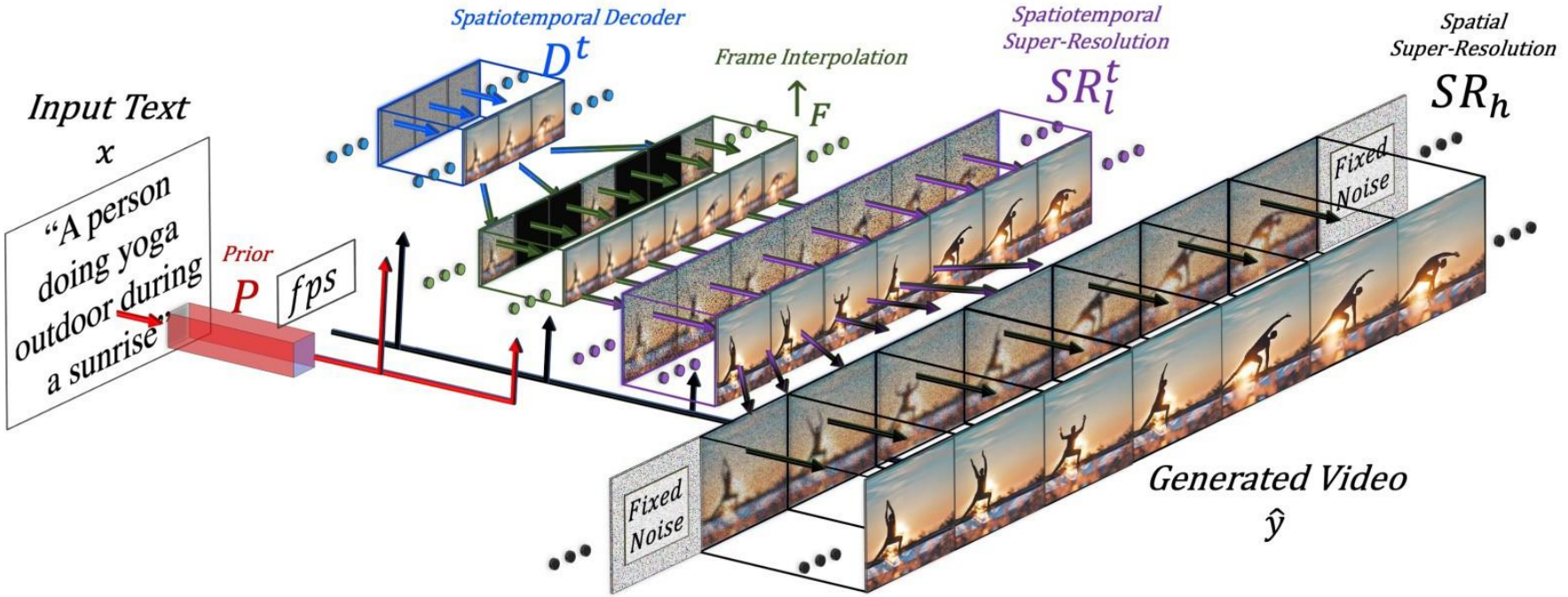


Figure 2: **Make-A-Video high-level architecture.** Given input text x translated by the prior P into an image embedding, and a desired frame rate fps, the decoder D^{t} generates 16 64×64 frames, which are then interpolated to a higher frame rate by $\uparrow_F$, and increased in resolution to 256×256 by SR_l^t and 768×768 by SR_h, resulting in a high-spatiotemporal-resolution generated video $\hat{y}$.

Open Problems

- Sampling from diffusion model is still slow (even with DDIM you need 250 steps)
  - How can one sample with even fewer steps?

- How good is the latent space of diffusion model for downstream tasks?
  - ResNet on ImageNet gives us great image features
  - LLM gives great text features
  - Can diffusion model beat imagenet feature?
  - Can diffusion model help us in discriminative tasks?

Open Problems

- How do we better control diffusion model for conditional generation?
  - With GANs we can do very fine-grained condition and can even use 3D intrinsics.
  - While Diffusion model produces impressive image quality, it often ignores detailed conditioning like segmentation mask, and works best for vague conditioning like text or class.
  - How do we enable that?

- Can we generate more involved videos with diffusion models?
  - This is something that is limited with GAN
  - Current diffusion models are limited in the kinds of video in can generate, but already a huge progress from GAN

Many many cool and creative applications that we haven't imagined before due to lack of such a powerful tool!

GAN vs Diffusion Model

Do you think GAN would disappear in few years and Diffusion model would be the de-facto for generative models?