

Diffusion Models

Naeemullah Khan

naeemullah.khan@kaust.edu.sa



جامعة الملك عبد الله
للعلوم والتكنولوجيا
King Abdullah University of
Science and Technology



LMH
Lady Margaret Hall

July 25, 2025

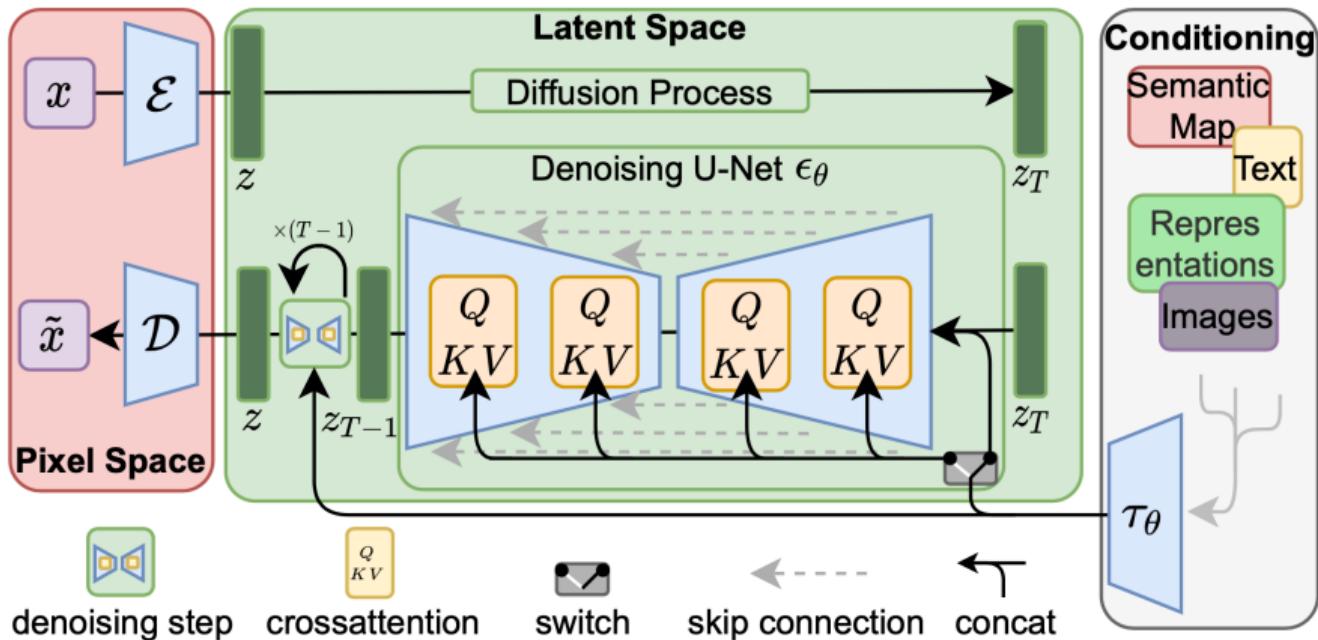


Table of Contents

1. GANs Recap
2. Motivation
3. Learning Outcomes
4. Introduction
5. Denoising
6. Forward Process
7. Noise Schedules
8. Reverse Denoising Process
9. Evidence Lower Bound (ELBO) & Variational Inference
10. Learning Denoising Models
11. Training & Sampling Algorithms
12. Network Architectures
13. Latent Diffusion Models (LDMs)

Table of Contents (cont.)

14. Issues with Diffusion Models

15. Applications

16. Summary & Limitations

17. References

We have previously discussed Generative Adversarial Networks (GANs), which are a powerful class of generative models. Here are some key points to remember:

- ▶ Instead of explicitly modeling the data probability distribution, GANs learn it implicitly.

We have previously discussed Generative Adversarial Networks (GANs), which are a powerful class of generative models. Here are some key points to remember:

- ▶ Instead of explicitly modeling the data probability distribution, GANs learn it implicitly.
- ▶ Two networks are trained jointly: the **generator** creates fake samples to fool the **discriminator**, while the discriminator tries to distinguish between real and fake samples.

We have previously discussed Generative Adversarial Networks (GANs), which are a powerful class of generative models. Here are some key points to remember:

- ▶ Instead of explicitly modeling the data probability distribution, GANs learn it implicitly.
- ▶ Two networks are trained jointly: the **generator** creates fake samples to fool the **discriminator**, while the discriminator tries to distinguish between real and fake samples.
- ▶ GANs have been state-of-the-art in image generation due to the high quality of their outputs.

The Journey of Generative Modeling:

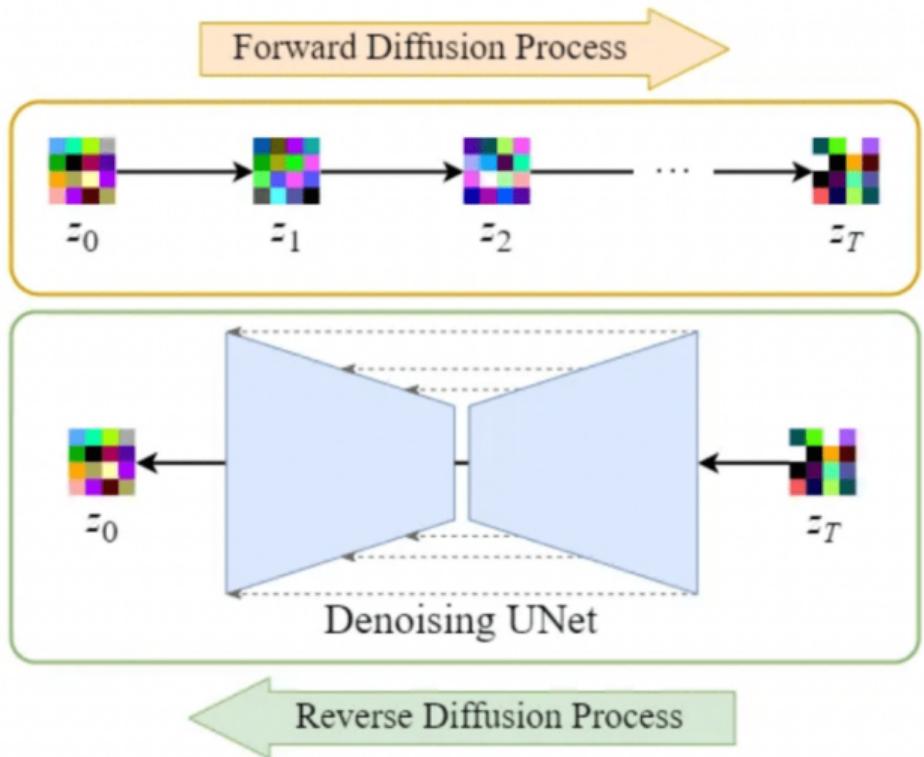
- ▶ Early successes: GANs and VAEs enabled impressive image synthesis and data generation.
- ▶ Drawbacks:
 - GANs: Training instability, mode collapse, and sensitivity to hyperparameters.
 - VAEs: Blurry outputs and limited expressiveness.
- ▶ Key Challenge: Achieving both high sample quality and stable, reliable training.

Where do we go from here?

- ▶ Need for models that are robust, interpretable, and capable of generating diverse, high-fidelity samples.

Enter Diffusion Models:

- ▶ Inspired by thermodynamics, diffusion models gradually transform noise into data.
- ▶ **Highlights:**
 - Simple training objective
 - Stable optimization
 - State-of-the-art sample quality
 - Probabilistic and interpretable framework



After this session, you will be able to:

- ▶ Explain the core principles behind diffusion-based generative models.
- ▶ Understand the forward and reverse processes in diffusion.
- ▶ Relate the training objective to variational inference and ELBO.
- ▶ Describe model architectures used in state-of-the-art diffusion models.
- ▶ Analyze and critique applications like GLIDE, DALL·E 2, Imagen, etc.

Diffusion Models: **Introduction**

Diffusion models are a class of probabilistic generative models that learn to reverse a gradual noising process applied to data. They have gained popularity due to their ability to generate samples with high fidelity and diversity.

Diffusion models are a class of probabilistic generative models that learn to reverse a gradual noising process applied to data. They have gained popularity due to their ability to generate samples with high fidelity and diversity.

- ▶ **Inspiration:** Rooted in concepts from thermodynamics and stochastic processes.

Diffusion models are a class of probabilistic generative models that learn to reverse a gradual noising process applied to data. They have gained popularity due to their ability to generate samples with high fidelity and diversity.

- ▶ **Inspiration:** Rooted in concepts from thermodynamics and stochastic processes.
- ▶ **Key Idea:**
 - *Forward process:* Progressively corrupt data by adding Gaussian noise.
 - *Reverse process:* Learn to recover the original data by reversing the noising process.

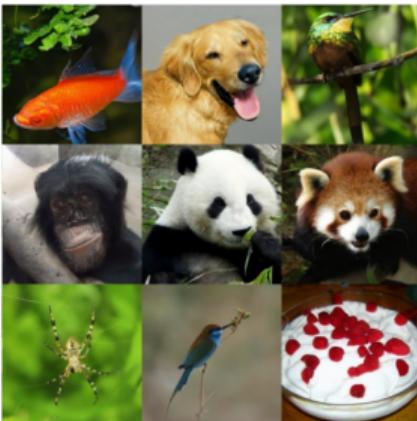
Diffusion models are a class of probabilistic generative models that learn to reverse a gradual noising process applied to data. They have gained popularity due to their ability to generate samples with high fidelity and diversity.

- ▶ **Inspiration:** Rooted in concepts from thermodynamics and stochastic processes.
- ▶ **Key Idea:**
 - *Forward process:* Progressively corrupt data by adding Gaussian noise.
 - *Reverse process:* Learn to recover the original data by reversing the noising process.
- ▶ **Popularity:** Known for producing high-quality and diverse generated samples.

- ▶ **Forward:** Clean → Noise (additive Gaussian steps)
- ▶ **Reverse:** Learn **U-Net** to remove noise step by step
- ▶ **Training:** Model predicts the noise added
- ▶ **Sampling:** Roll back from pure noise to a data sample

Diffusion Models: Denoising

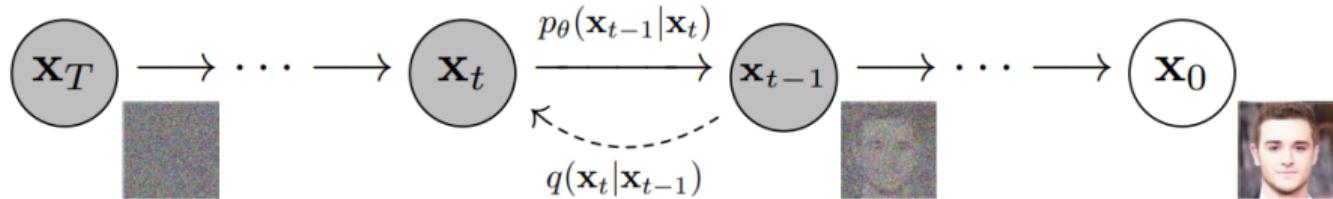
Denoising diffusion models, also known as score-based generative models, have recently emerged as a powerful class of generative models. They achieve remarkable results in high-fidelity image generation, often outperforming GANs.



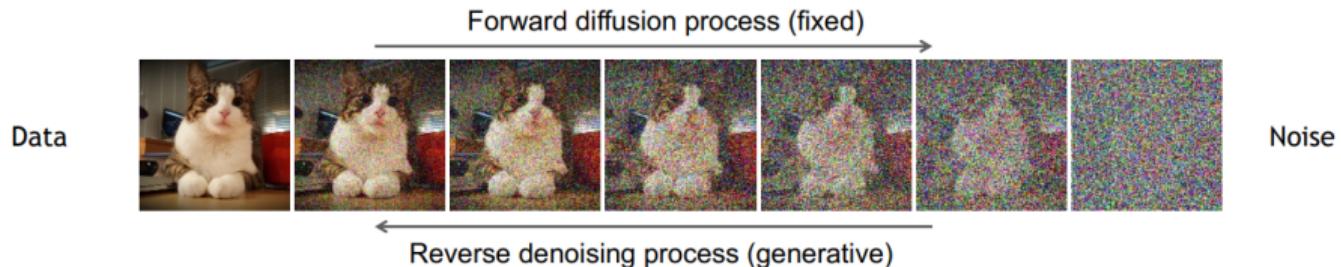
Diffusion Models Beat GANs on Image Synthesis **Dhariwal & Nichol, OpenAI, 2021**

Denoising diffusion models consist of two processes:

- ▶ A fixed (predefined) **forward diffusion process** q , which gradually adds Gaussian noise to an image until only pure noise remains.
- ▶ A learned **reverse denoising diffusion process** p_θ , where a neural network is trained to gradually denoise an image starting from pure noise, eventually recovering the original image.



Denoising Diffusion Probabilistic Models (cont.)



Diffusion Models: **Forward Process**

- ▶ The forward process is a Markov chain that iteratively adds Gaussian noise to the image at each timestep.
- ▶ Given clean data \mathbf{x}_0 , we generate \mathbf{x}_t as:

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I})$$

- ▶ Over many steps (e.g., 1000), the data becomes indistinguishable from pure noise.

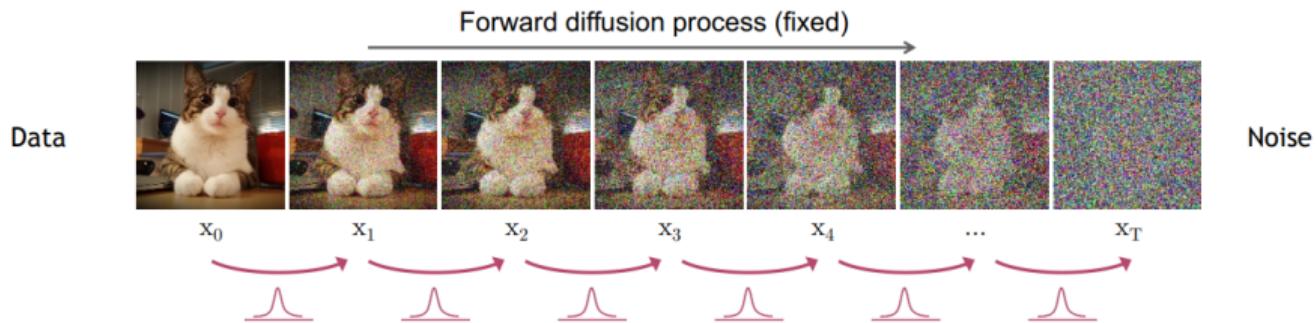
Forward Diffusion Process (cont.)

- ▶ This process is predefined and not learned.
- ▶ The full forward process:

$$q(\mathbf{x}_{1:T} | \mathbf{x}_0) = \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1})$$

- ▶ Each new (slightly noisier) image at time step t is drawn from a conditional Gaussian distribution with $\mu_t = \sqrt{1 - \beta_t} \mathbf{x}_{t-1}$ and $\sigma_t^2 = \beta_t$, which we can do by sampling $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and then setting $\mathbf{x}_t = \sqrt{1 - \beta_t} \mathbf{x}_{t-1} + \sqrt{\beta_t} \epsilon$.

Forward Diffusion Process (cont.)



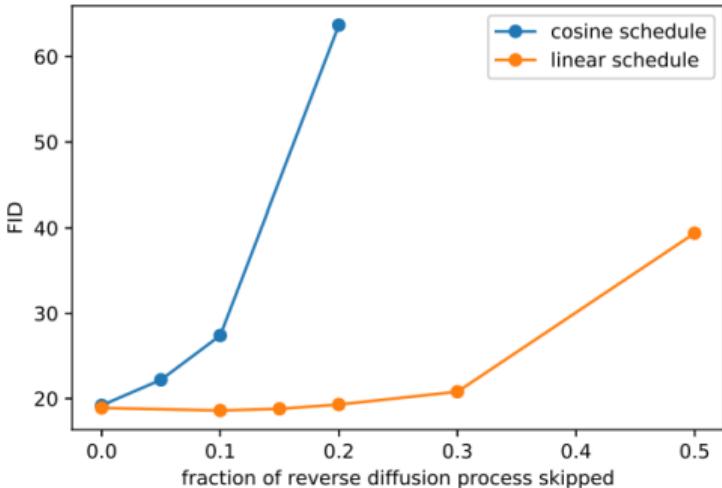
- ▶ What if we want to go directly from q_0 to q_t ?
 - ▶ Let $\bar{\alpha}_t = \prod_{s=1}^t (1 - \beta_s)$, then

$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I})$$

- ▶ This allows us, during training, to optimize random terms of the loss function L (i.e., randomly sample t and optimize L_t).

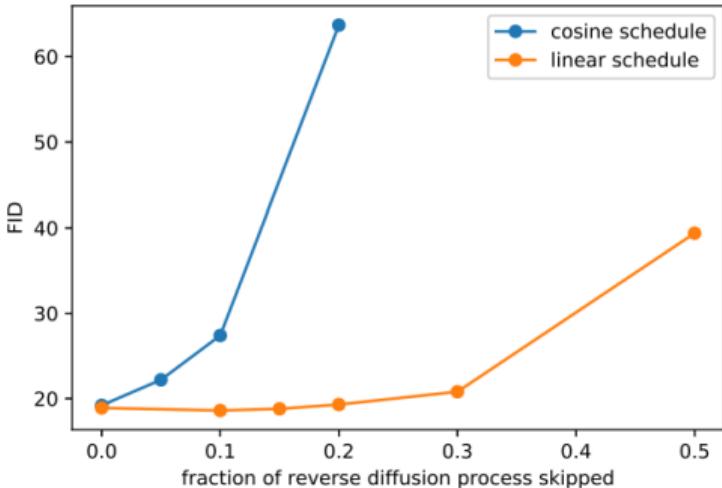
Diffusion Models: Noise Schedules

- ▶ The noise schedule $\{\beta_t\}$ controls the amount of noise added at each timestep.



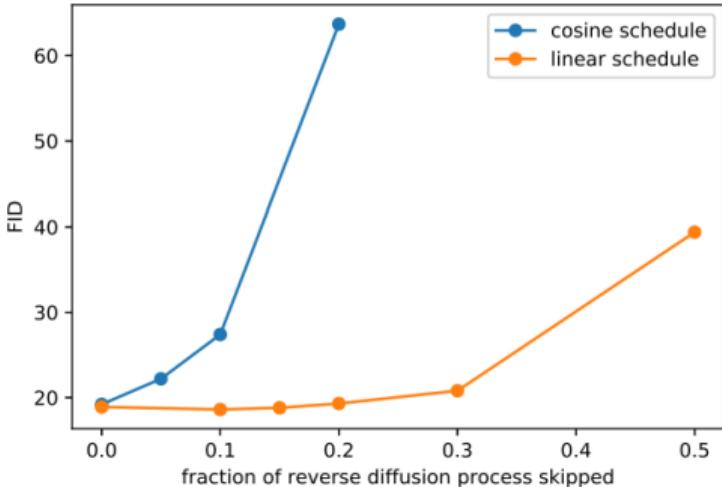
Comparison of linear and cosine noise schedules.

- ▶ The noise schedule $\{\beta_t\}$ controls the amount of noise added at each timestep.
- ▶ Common schedules:
 - **Linear:** Simple and intuitive.
 - **Cosine:** Preserves perceptual signal longer; often yields better image generation results.



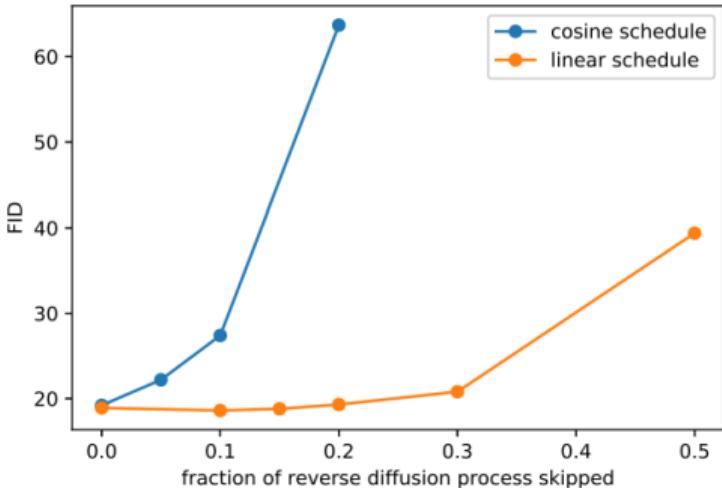
Comparison of linear and cosine noise schedules.

- ▶ The noise schedule $\{\beta_t\}$ controls the amount of noise added at each timestep.
- ▶ Common schedules:
 - **Linear:** Simple and intuitive.
 - **Cosine:** Preserves perceptual signal longer; often yields better image generation results.
- ▶ The choice of schedule impacts training dynamics and the quality of generated samples.



Comparison of linear and cosine noise schedules.

- ▶ The noise schedule $\{\beta_t\}$ controls the amount of noise added at each timestep.
- ▶ Common schedules:
 - **Linear:** Simple and intuitive.
 - **Cosine:** Preserves perceptual signal longer; often yields better image generation results.
- ▶ The choice of schedule impacts training dynamics and the quality of generated samples.
- ▶ Visualizing different schedules helps understand how noise accumulates over time.



Comparison of linear and cosine noise schedules.

Diffusion Models: Reverse Denoising Process

Reverse Denoising Process

In the reverse denoising process, we denoise Gaussian noise to generate an image.

Reverse Denoising Process

In the reverse denoising process, we denoise Gaussian noise to generate an image.

- ▶ We start with a sample from the noise distribution $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; 0, 1)$.

Reverse Denoising Process

In the reverse denoising process, we denoise Gaussian noise to generate an image.

- ▶ We start with a sample from the noise distribution $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; 0, 1)$.
- ▶ The goal is to iteratively denoise this sample to recover the original image \mathbf{x}_0 .

In the reverse denoising process, we denoise Gaussian noise to generate an image.

- ▶ We start with a sample from the noise distribution $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; 0, 1)$.
- ▶ The goal is to iteratively denoise this sample to recover the original image \mathbf{x}_0 .
- ▶ At each time step t , we want to sample \mathbf{x}_{t-1} given \mathbf{x}_t .

In the reverse denoising process, we denoise Gaussian noise to generate an image.

- ▶ We start with a sample from the noise distribution $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; 0, 1)$.
- ▶ The goal is to iteratively denoise this sample to recover the original image \mathbf{x}_0 .
- ▶ At each time step t , we want to sample \mathbf{x}_{t-1} given \mathbf{x}_t .
- ▶ The reverse process is defined as $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, which we will learn using a neural network.

Reverse Denoising Process

- ▶ The reverse process is a Markov chain that iteratively denoises the image.
- ▶ We want to sample \mathbf{x}_{t-1} from $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$.
- ▶ However, $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is unknown and needs to be learned.

Reverse Denoising Process (cont.)

- ▶ We approximate it with a neural network that predicts the mean and variance of the Gaussian distribution.
- ▶ If β_t is small enough at each time step, we can assume $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is a Gaussian distribution:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

where μ_θ and Σ_θ are neural networks conditioned on \mathbf{x}_t and t .

Reverse Denoising Process (cont.)



- ▶ For simplicity, we can fix the variance Σ_θ to a constant value (e.g., $\beta_t \mathbf{I}$) and only learn the mean μ_θ .

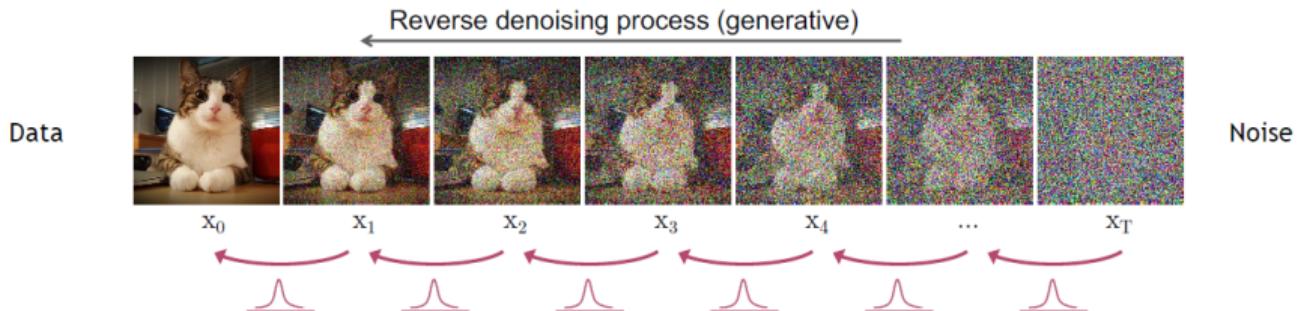
Reverse Denoising Process (cont.)

- The reverse process can then be expressed as:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \beta_t \mathbf{I})$$

and the joint distribution over all time steps is:

$$p_{\theta}(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^T p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)$$



Reverse Denoising Process (cont.)



Reverse denoising process in diffusion models.

Diffusion Models: Evidence Lower Bound (ELBO) & Variational Inference

- ▶ Direct maximization of the marginal likelihood $p(x)$ is often intractable.

- ▶ Direct maximization of the marginal likelihood $p(\mathbf{x})$ is often intractable.
- ▶ **Solution:** Use the *Evidence Lower Bound* (ELBO) as a tractable proxy objective:

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \right] = \text{ELBO}$$

- ▶ Direct maximization of the marginal likelihood $p(\mathbf{x})$ is often intractable.
- ▶ **Solution:** Use the *Evidence Lower Bound* (ELBO) as a tractable proxy objective:

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \right] = \text{ELBO}$$

- ▶ **ELBO Decomposition:**

- **Reconstruction term:** Measures how well the model can reconstruct the data \mathbf{x} from the latent variables \mathbf{z} .
- **KL divergence:** Penalizes the difference between the approximate posterior $q(\mathbf{z}|\mathbf{x})$ and the true/model prior $p(\mathbf{z})$.

- ▶ Direct maximization of the marginal likelihood $p(\mathbf{x})$ is often intractable.
- ▶ **Solution:** Use the *Evidence Lower Bound* (ELBO) as a tractable proxy objective:

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \right] = \text{ELBO}$$

- ▶ **ELBO Decomposition:**

- **Reconstruction term:** Measures how well the model can reconstruct the data \mathbf{x} from the latent variables \mathbf{z} .
- **KL divergence:** Penalizes the difference between the approximate posterior $q(\mathbf{z}|\mathbf{x})$ and the true/model prior $p(\mathbf{z})$.

- ▶ The full training objective is derived from ELBO:

$$\mathcal{L}_{\text{ELBO}} = \sum_t \mathbb{E}[D_{KL}(q(x_{t-1}|x_t, x_0) \| p_\theta(x_{t-1}|x_t))]$$

- ▶ Direct maximization of the marginal likelihood $p(\mathbf{x})$ is often intractable.
- ▶ **Solution:** Use the *Evidence Lower Bound* (ELBO) as a tractable proxy objective:

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \right] = \text{ELBO}$$

- ▶ **ELBO Decomposition:**

- **Reconstruction term:** Measures how well the model can reconstruct the data \mathbf{x} from the latent variables \mathbf{z} .
- **KL divergence:** Penalizes the difference between the approximate posterior $q(\mathbf{z}|\mathbf{x})$ and the true/model prior $p(\mathbf{z})$.

- ▶ The full training objective is derived from ELBO:

$$\mathcal{L}_{\text{ELBO}} = \sum_t \mathbb{E}[D_{KL}(q(x_{t-1}|x_t, x_0) \| p_\theta(x_{t-1}|x_t))]$$

- ▶ **Training:** Maximize the ELBO to train the model, which encourages both accurate reconstruction and regularization of the latent space

Diffusion Models: **Learning Denoising Models**

In diffusion models, we learn a denoising function that can reverse the noise addition process. The training objective is to minimize the difference between the predicted noise and the actual noise added at each step.

- ▶ For training, we can form a variational upper bound, commonly used for training variational autoencoders:

$$\mathbb{E}_{q(x_0)} [-\log p_\theta(x_0)] \leq \mathbb{E}_{q(x_0)q(x_{1:T}|x_0)} \left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \right]$$

- ▶ The ELBO for this process can be expressed as a sum of losses at each time step t :

$$L = L_0 + L_1 + \cdots + L_T$$

Learning Denoising Models (cont.)

- ▶ We can reparametrize the mean so that the neural network learns to predict the added noise (using a network $\epsilon_\theta(\mathbf{x}_t, t)$ for noise level t in the KL terms that constitute the losses). This means our neural network becomes a noise predictor, rather than a direct mean predictor. The mean can be computed as:

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \right)$$

- ▶ The final objective function L_t (for a random time step t and $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$) is:

$$\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)\|^2 = \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon, t)\|^2$$

- ▶ For a complete derivation, see [here](#).

Diffusion Models: **Training & Sampling Algorithms**

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
      
$$\nabla_{\theta} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, t)\|^2$$

6: until converged
```

Training Algorithm

1. Sample x_0 from the data distribution.
2. Randomly select a timestep t .
3. Corrupt x_0 with noise to obtain x_t .
4. Model predicts noise $\epsilon_{\theta}(x_t, t)$.
5. Compute loss: mean squared error between predicted and true noise.
6. Optimize using SGD or Adam.

Algorithm 2 Sampling

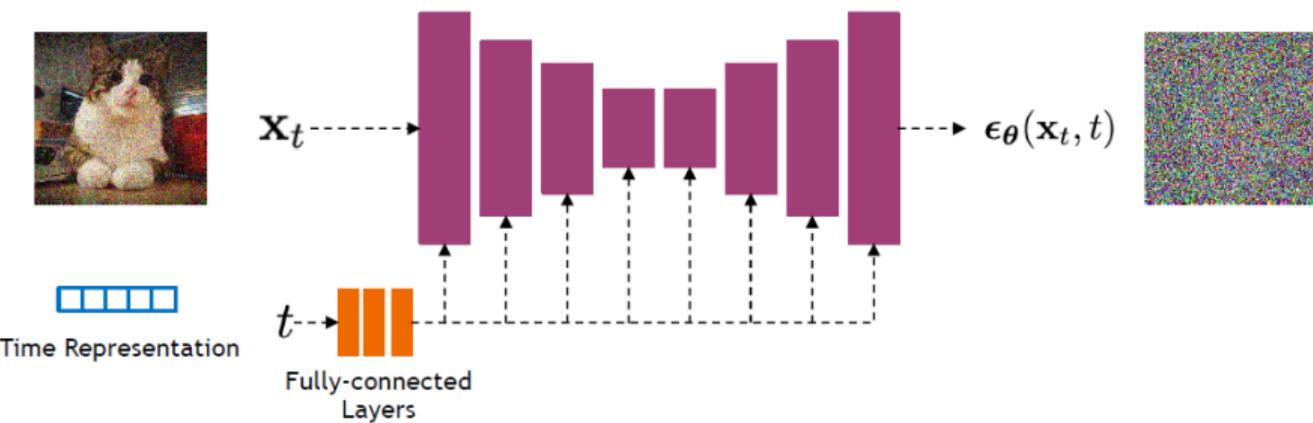
```
1:  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
2: for  $t = T, \dots, 1$  do
3:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
4:    $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t,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

Sampling Algorithm

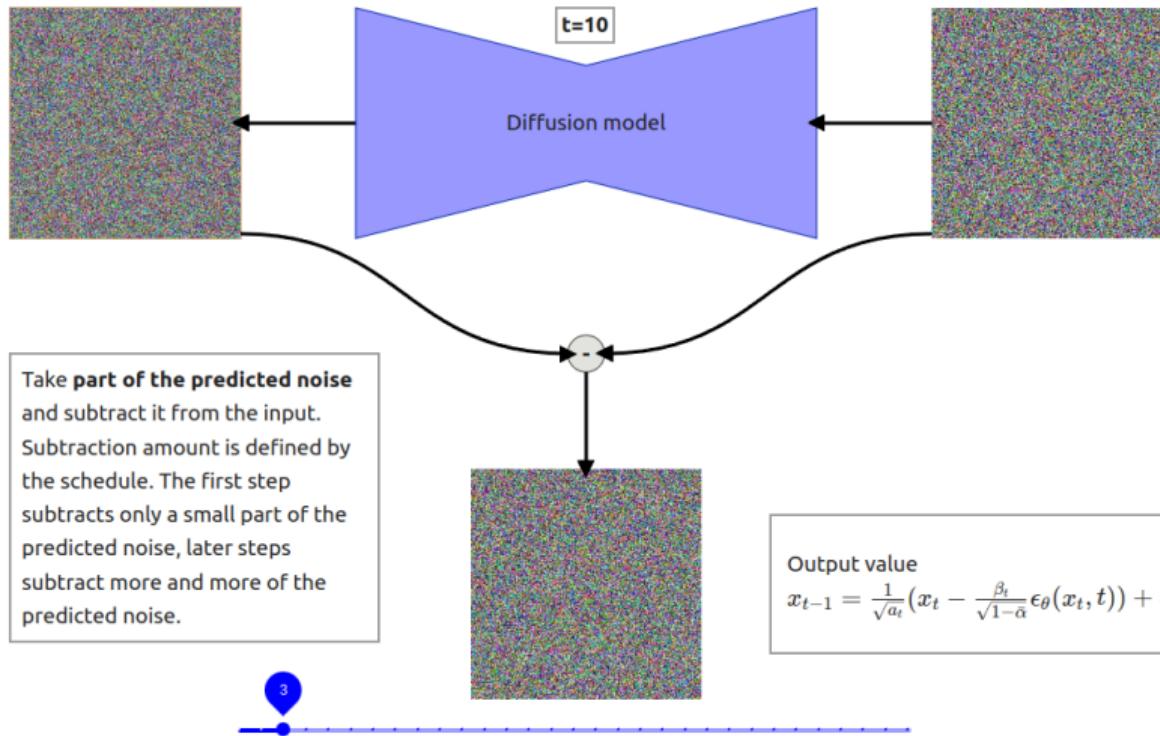
1. Start from Gaussian noise x_T .
2. For each timestep $t = T, \dots, 1$:
 - 2.1 Predict noise $\epsilon_\theta(x_t, t)$.
 - 2.2 Compute posterior mean $\mu_\theta(x_t, t)$.
 - 2.3 Sample x_{t-1} from the Gaussian posterior.
 - 2.4 (Optional) Apply guidance (e.g., classifier-free) for improved results.

Diffusion Models: Network Architectures

- ▶ Diffusion models commonly use U-Net architectures with ResNet blocks and self-attention layers to represent $\epsilon_\theta(x_t, t)$.
- ▶ Time is encoded using sinusoidal positional embeddings or random Fourier features.



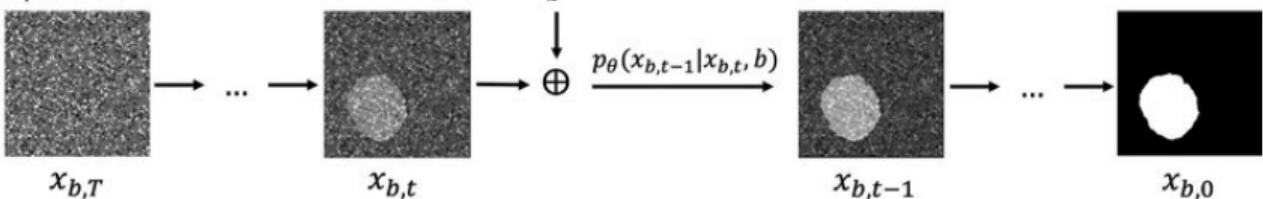
Network Architectures (cont.)



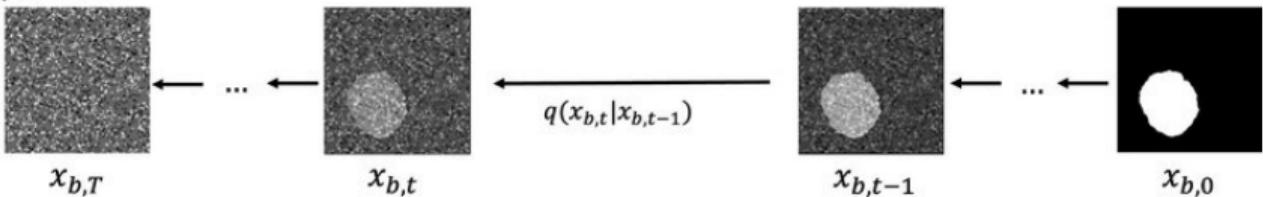
Network Architectures (cont.)



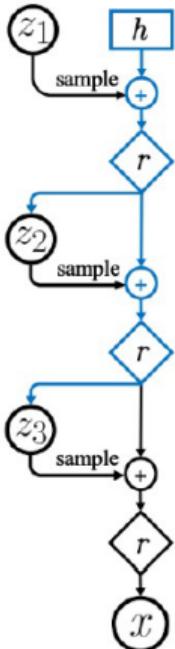
backward process:



forward process:

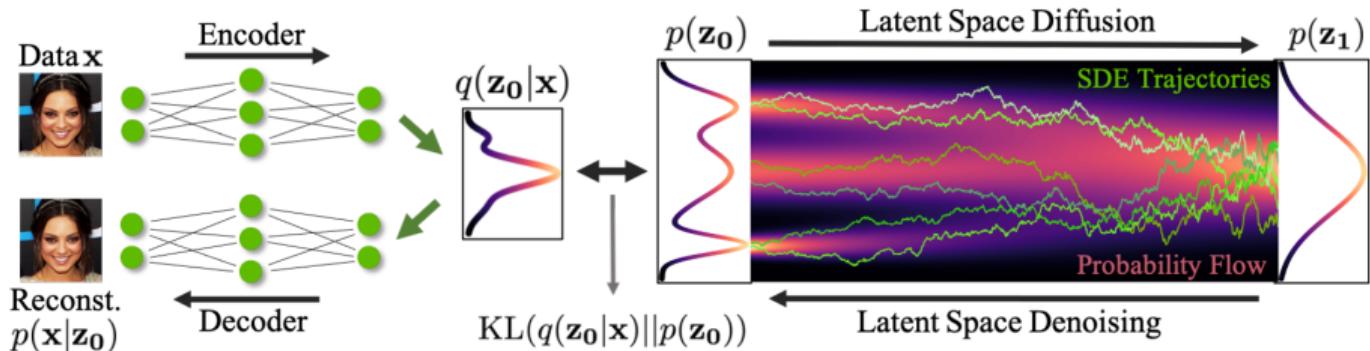


- ▶ Diffusion models can be viewed as a special form of hierarchical VAEs (one VAE after another).
- ▶ In diffusion models:
 - The encoder is fixed.
 - The latent variables have the same dimension as the data.
 - The denoising model is shared across different timesteps.
 - The model is trained with a reweighted variational bound.



Diffusion Models: **Latent Diffusion Models (LDMs)**

What is a Latent Space?



- ▶ A **latent space** is an abstract, compressed space where high-dimensional data (like images) is represented in a lower-dimensional form.
- ▶ Think of it as a hidden layer of meaning—each point in this space represents a potential image or concept.
- ▶ In classical models like GANs or VAEs, the latent space is directly sampled from (e.g., a vector \mathbf{z}), and the generator turns it into an image.
- ▶ **Latent** space = “Hidden space of ideas or features”

Two main ways latent spaces are used in diffusion:

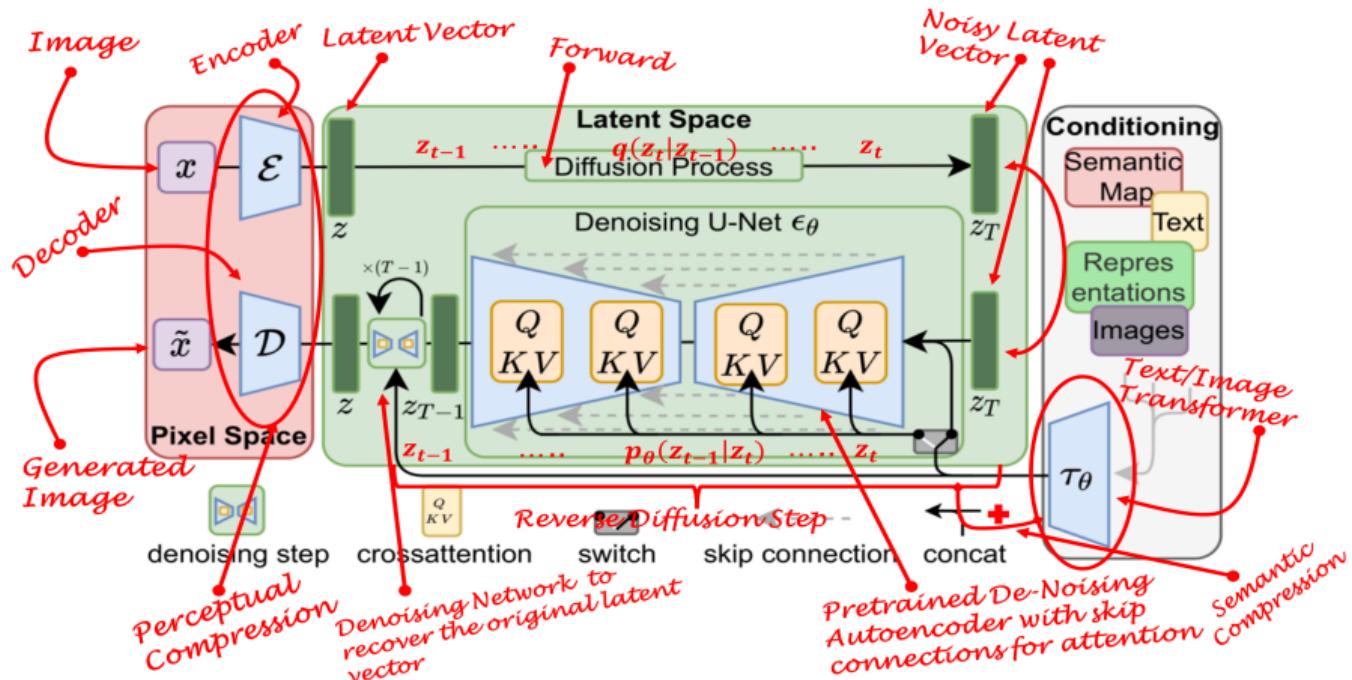
► **1. Latent Diffusion Models (LDMs):**

- Diffusion is performed in a compressed latent space, not directly on high-dimensional images.
- **Pipeline:** Compress image → Denoise in latent space → Decode back to image.
- Enables faster, cheaper training and inference, and allows for semantic control (e.g., text prompts, masks).

► **2. Latent Representations in Noise Vectors:**

- Even in standard diffusion models, the initial noise vector acts as a latent code.
- Changing the noise vector generates different images or morphs between them.

Where is the Latent Space in Diffusion? (cont.)



Why Latent Diffusion?

- ▶ Reduces computational cost by operating in a lower-dimensional space.
- ▶ Maintains high-quality image generation while being more efficient.
- ▶ Allows for more complex operations like text-guided generation and image editing.

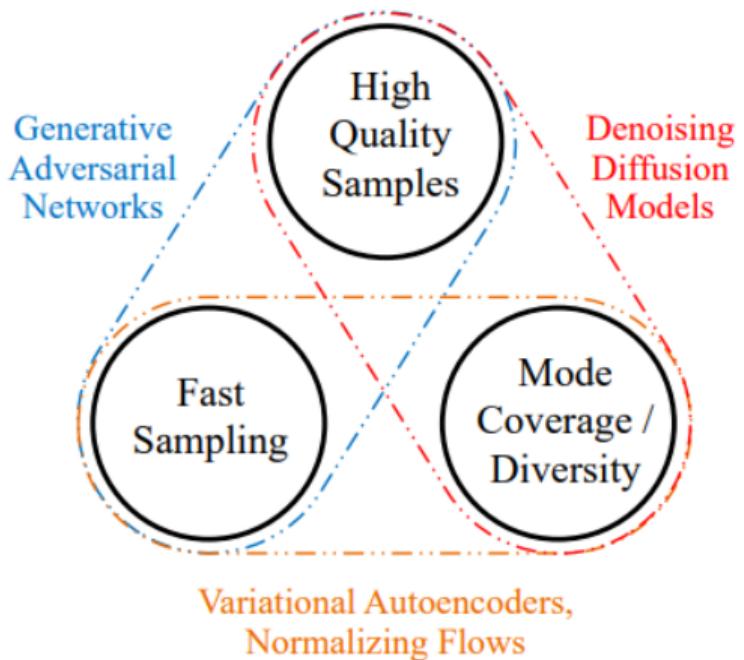
Applications:

- ▶ Text-to-image generation (e.g., Stable Diffusion) uses text embeddings in latent space.
- ▶ Image editing by modifying specific parts of the latent.
- ▶ Inversion: map a real image to latent, edit, then decode back.

Diffusion Models: Issues with Diffusion Models

- ▶ **Slow Sampling:** Diffusion models require hundreds to thousands of iterative steps to generate a sample, making them computationally expensive and slow compared to other generative models.

Trilemma of Generative Learning



Generative Learning Trilemma: Balancing sample quality, diversity, and fast sampling.

⁰Tackling the Generative Learning Trilemma with Denoising Diffusion GANs

► Faster Sampling:

- Denoising Diffusion Implicit Models (DDIM): Propose a non-Markovian process for faster and deterministic sampling by skipping steps and refining the sample.

► Improved Training:

- Improved Denoising Diffusion Probabilistic Models: Learn the noise variance σ^2 during training, rather than fixing it, for better flexibility and performance.

► Score-Based Modeling:

- Score-Based Generative Modeling through SDEs: Model the gradient of the log-density (score function) using stochastic differential equations for improved sample quality.

► Combining GANs and Diffusion:

- Denoising Diffusion GANs: Use GANs to model each denoising step, addressing the trilemma by improving speed and sample quality.

► High-Fidelity Generation:

- Cascaded Diffusion Models: Employ a cascade of diffusion models at different resolutions to boost sample fidelity.

Diffusion Models: Applications



“a man wearing a white hat”

Image Inpainting with GLIDE

⁰GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models

Application: DALL.E 2 - OpenAI



a shiba inu wearing a beret and black turtleneck



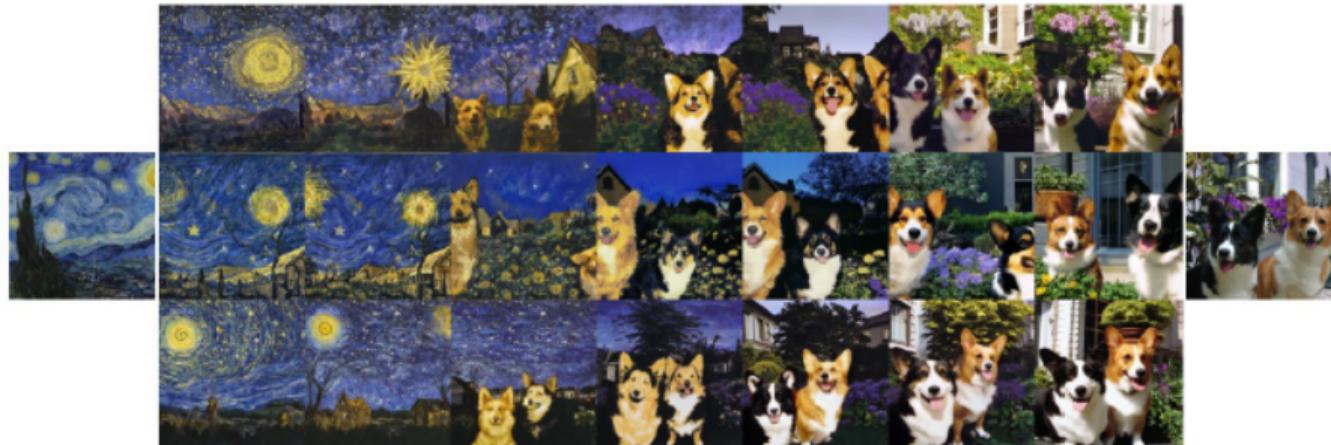
a close up of a handpalm with leaves growing from it

Text to image generation eith DALL.E 2



Fix the CLIP embedding z ,
Decode using different decoder latents x_T .

Image Variations



Interpolate image CLIP embeddings z_i

Use different x_T to get different interpolation trajectories.

Image interpolation



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.

Text Difference Image interpolation



Imagen

A brain riding a rocketship heading towards the moon.



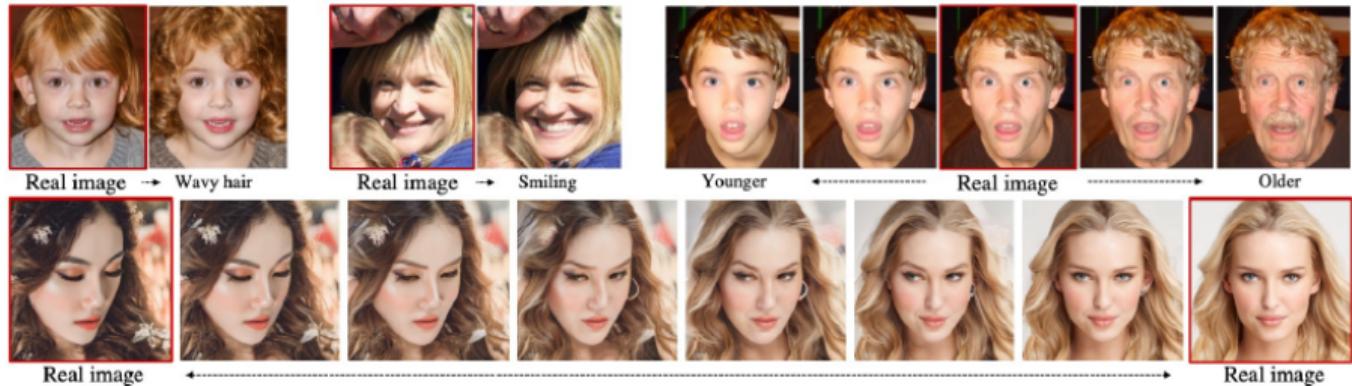
A dragon fruit wearing karate belt in the snow.



Imagen

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

Diffusion Autoencoders



Changing the semantic latent z_{sem}

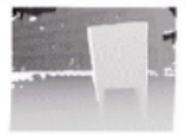
Learning semantic meaningful latent representations in diffusion models

Super Resolution



Natural Image Super Resolution $64 \times 64 \rightarrow 256 \times 256$

3D Shape Generation



Try It Yourself!

Text to Image generation with stable diffusion.

<https://huggingface.co/spaces/stabilityai/stable-diffusion>

Diffusion Models: **Summary & Limitations**

- ▶ Diffusion models add noise to data and learn to reverse it
- ▶ Core task: Predict noise at different steps
- ▶ Training is efficient; Sampling is iterative and slow
- ▶ U-Nets with attention are the winning architecture

The Generative AI Landscape

Text

MARKETING

copy.ai

 regie.ai

 Jasper

 copysmith

anyword

 Writesonic

 **CONTENDA**

 Hypotenuse AI

SUPPORT
(CHAT/EMAIL)

TO COME

KNOWLEDGE

 glean

 mem

SALES

 LAVENDER

 Smartwriter.ai

OTHER

 Character.AI

 AI DUNGEON

GENERAL WRITING

 wordtune

 OTHERSIDEAI

 WRITER

 COMPOSE AI

 Rytr

 LEX

MODELS: OPENAI GPT-3 DEEPMIND GOPHER ANTHROPIC AI2 ALIBABA, YANDEX, ETC.

FACEBOOK OPT HUGGING FACE BLOOM COHERE

Video

VIDEO EDITING/ GENERATION

 runway

PERSONALIZED VIDEOS

 tavus

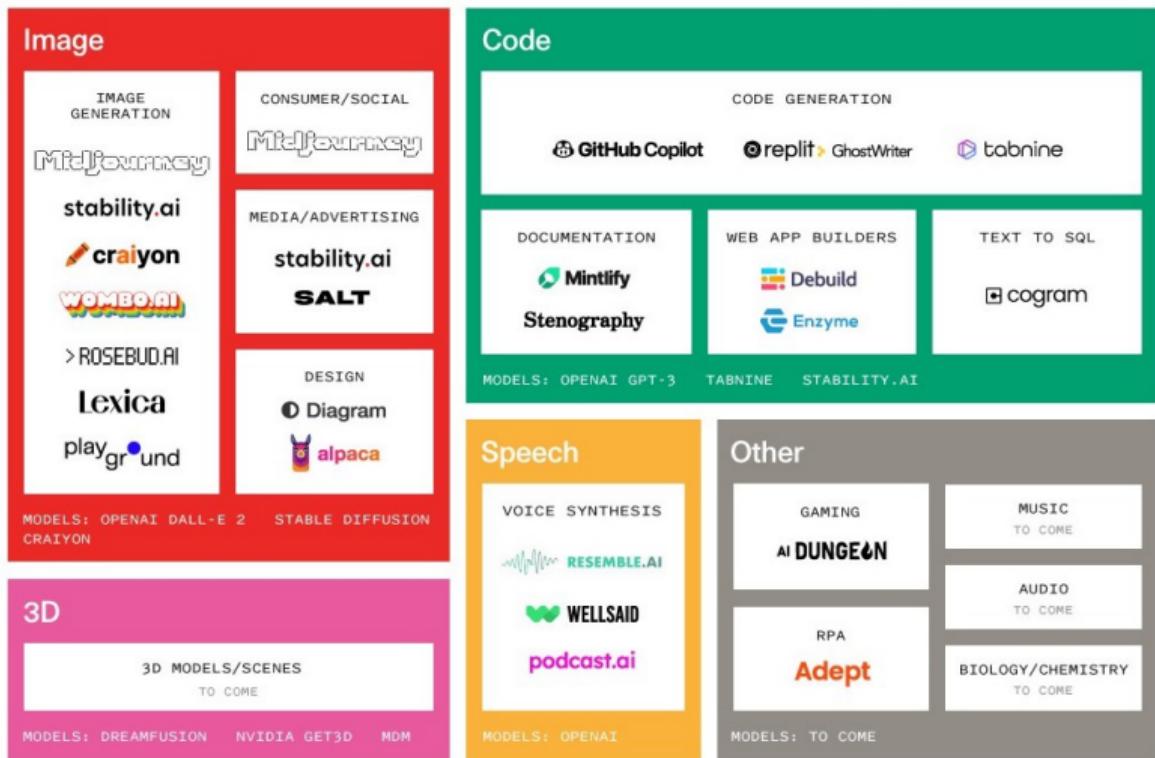
 synthesia

 Hour One.

 Rephrase.ai

MODELS: MICROSOFT X-CLIP META MAKE-A-VIDEO

The Generative AI Landscape



- ▶ **Slow sampling** – iterative process
- ▶ **Data-heavy** – needs lots of varied data
- ▶ **Distribution miss** – struggles in tails of data
- ▶ **Compute needs** – high GPU/time investment
- ▶ **Misinformation concerns** – used in deepfake creation

Tutorials and Surveys

- ▶ Kreis, K., Gao, R., & Vahdat, A.: [CVPR 2022 Tutorial: Diffusion Models](#)
- ▶ Rogge, L., & Rasul, K.: [The Annotated Diffusion Model](#)
- ▶ Yang Song: [Score-Based Generative Modeling: A Brief Overview](#)
- ▶ Lilian Weng: [What are Diffusion Models?](#)

Foundational Papers

- ▶ Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models
- ▶ Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., & Ganguli, S. (2015). Deep Unsupervised Learning using Nonequilibrium Thermodynamics
- ▶ Song, Y., & Ermon, S. (2019). Generative Modeling by Estimating Gradients of the Data Distribution
- ▶ Nichol, A., & Dhariwal, P. (2021). Improved Denoising Diffusion Probabilistic Models

Applications and Notable Models

- ▶ Ramesh, A., et al. (2022). Hierarchical Text-Conditional Image Generation with CLIP Latents (DALL·E 2)
- ▶ Saharia, C., et al. (2022). Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding (Imagen)
- ▶ Nichol, A., Dhariwal, P., et al. (2021). GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models
- ▶ Preechakul, T., et al. (2022). Diffusion Autoencoders: Toward a Meaningful and Decodable Representation
- ▶ Saharia, C., et al. (2022). Image Super-Resolution via Iterative Refinement
- ▶ Poole, B., et al. (2022). DreamFusion: Text-to-3D using 2D Diffusion

GANs and Related Reading

- ▶ Goodfellow, I., et al. (2014). [Generative Adversarial Nets](#)
- ▶ Creswell, A., et al. (2018). [Generative Adversarial Networks: An Overview](#)

Credits

Dr. Prashant Aparajeya

Computer Vision Scientist — Director(AISimply Ltd)

p.aparajeya@aisimply.uk

This project benefited from external collaboration, and we acknowledge their contribution with gratitude.