Deep Generative models: VAE, GAN, and Diffusion

Rubin Won
UST-ETRI
MS Student
rubrub@etri.re.kr
Nov 16th, 2023

Contents

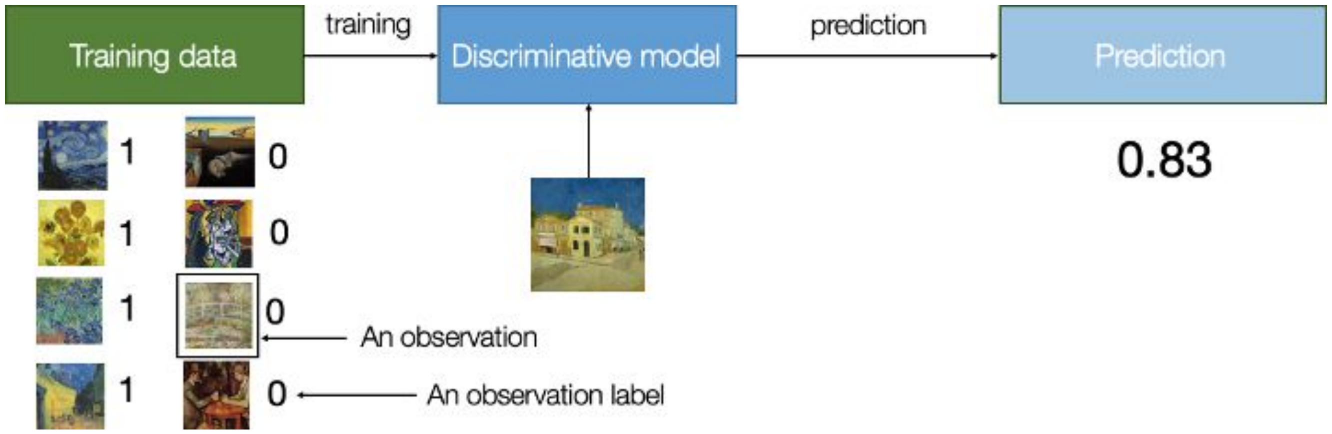
Recap

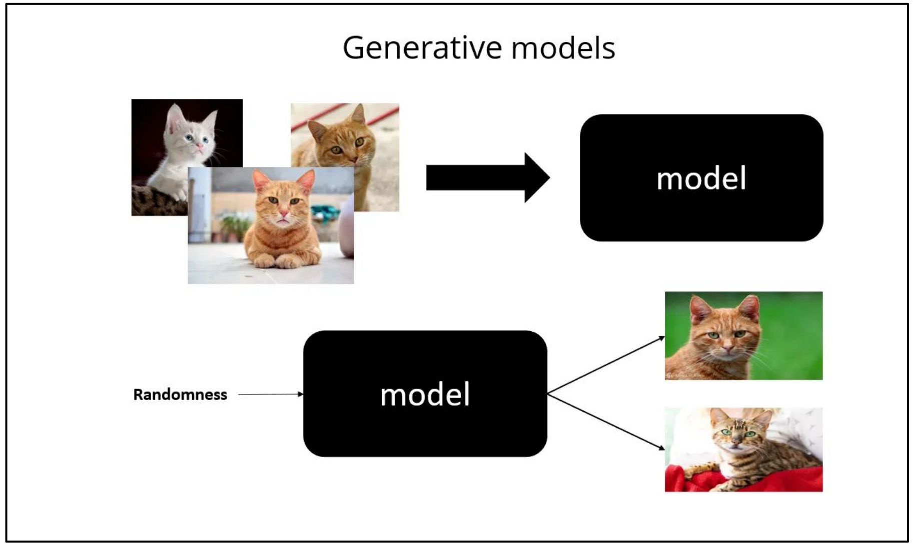
01 Recap - What is Generative Model?

➢ A **generative model** can be defined as follows:

- A generative model describes how a dataset is **generated**, in terms of a **probabilistic model.** By **sampling** from this model, we are able to **generate new data.**

➢ A generative model must also be **probabilistic** rather than deterministic.





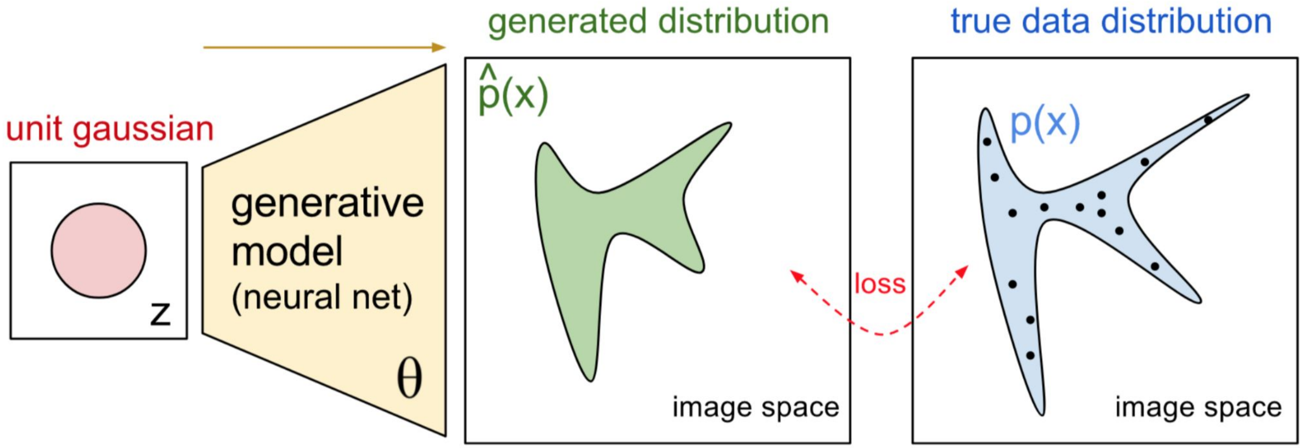
< Generative modeling process >

Introduction & Background - **What is Generative Model?**

➢ Generative model is a statistical model of the **joint probability distribution P(X, Y)** on given observable variable X and target variable Y

Goal for generative models:

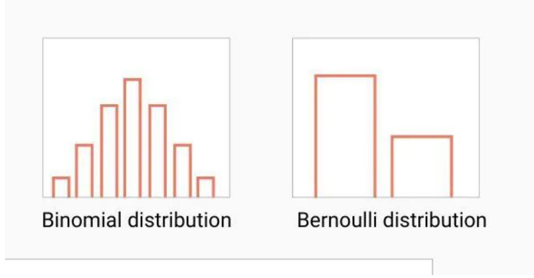
➢ To learn the underlying **data distribution** of the training data.
- Once this distribution is learned, the **model can generate new data points** that are statistically similar to the training data.



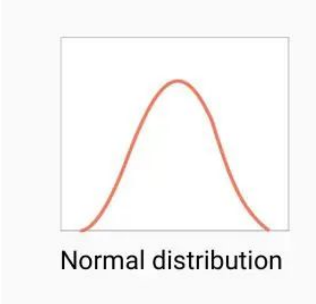
VAEs

VAEs - Data distributions for random variables

1. **Discrete** Random Variables → **Binomial and multinomial distributions**

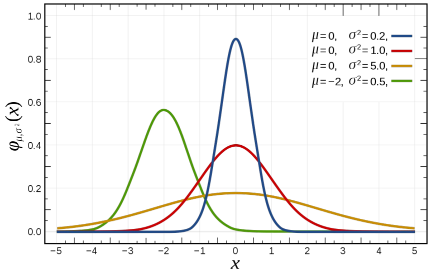


1. **Continuous** Random Variables → **Gaussian distributions**



Those distributions are called **parametric distributions** since they are governed by small number of adaptive parameters (e.g. mean, variance)

→ We need a procedure for determining suitable values for the **parameters** given an observed dataset.



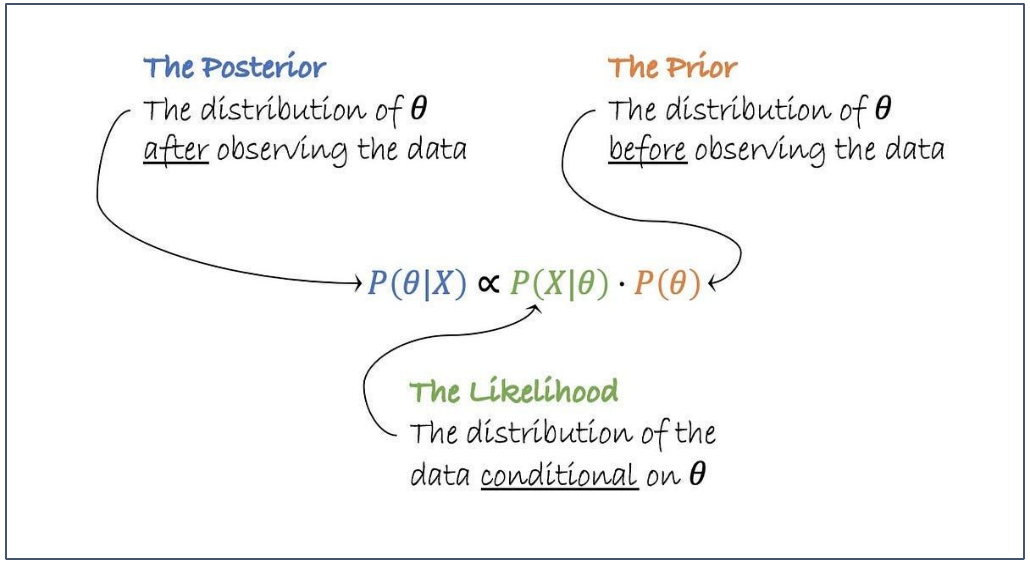
02 **VAEs** - What is Variational Inference?

Purpose of Inference: In probabilistic modeling, the **goal is to find the distribution of data**.
→ This involves identifying the parameters of the model.

Posterior distribution of latent given data.

$$p(z|X) = \frac{p(X|z)p(z)}{\int p(X|z)p(z)dz}$$

This becomes intractable!



➢ We use **Variational inference** to solve this issue.

Variational inference approximates the posterior distribution **p(z|X)** with some simpler distribution **q(z)** which is "close" to our target distribution.

KL divergence

$$KL(q(z)||p(z)) = \int q(z)\log\frac{q(z)}{p(z)}dz = \mathrm{E}_{q(z)}[\log\frac{q(z)}{p(z)}]$$

→ KL divergence achieve its minimum value when q(z) = p(z)

The optimization problem

$$q^*(z) = \underset{q\in Q}{\arg\min} KL(q(z)||p(z|x))$$

← Hard to optimize

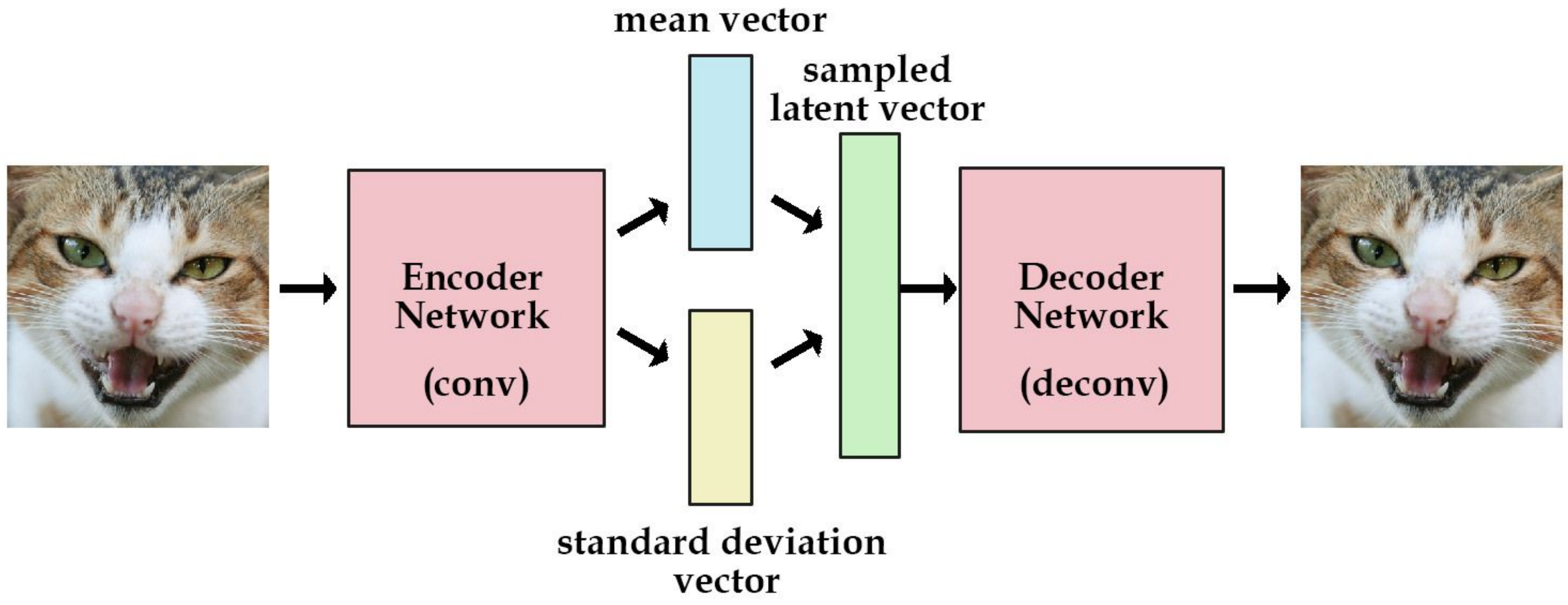
$$\begin{aligned} KL(q(z)||p(z|x)) &= \mathrm{E}_{q(z)}[\log\frac{q(z)}{p(z|x)}] \\ &= \mathrm{E}_{q(z)}[\log q(z)] - \mathrm{E}_{q(z)}[\log p(z|x)] \\ &= \mathrm{E}_{q(z)}[\log q(z)] - \mathrm{E}_{q(z)}[\log\frac{p(z,x)}{p(x)}] \\ &= \mathrm{E}_{q(z)}[\log q(z)] - \mathrm{E}_{q(z)}[\log p(z,x)] + \mathrm{E}_{q(z)}[\log p(x)] \\ &= \mathrm{E}_{q(z)}[\log q(z)] - \mathrm{E}_{q(z)}[\log p(z,x)] + \log p(x) \end{aligned}$$

ELBO

$$\mathrm{ELBO}(q) = \mathrm{E}_{q(z)}[\log p(z,x)] - \mathrm{E}_{q(z)}[\log q(z)]$$

→ Since we need to “minimize” KL divergence, we give **lower bound** of this and **maximize** it.

VAE: probabilistic autoencoder



- ➢ **Encoder**: takes the observed data x and produces the latent variable z.
- ➢ **Decoder**: uses the z produced by the encoder to reconstruct x.

VAEs - Creating a Latent Vector

Decoder part of VAE assumes a gaussian distribution.

$$p(x|z) = N\left(x|f_{\mu}(z), f_{\sigma}(z)^2 \times I\right)$$

Using MLE parameter estimation, by marginal log-likelihood log p(x) we need to optimize log p(x)

$$\log p(x) = \log \sum_{z} p\left(x|f_{\mu}(z), f_{\sigma}(z)^2 \times I\right)p(z)$$

← Hard to optimize

Using Variational Inference - We have the ELBO function

$$\log p(x) \geq \boxed{E_{z\sim q(z)}[\log p(x|z)] - D_{KL}(q(z)||p(z))}$$

(log evidence) we want this to be high
to say that the model explains the data well

MAXIMIZE ELBO to maximize log p(x)

In typical variational inference, q(z) is set to be a gaussian distribution

$$q(z) = N\left(\mu_q, \sigma_q^2\right)$$

→ for more complex data, parameters of q are set as functions of x

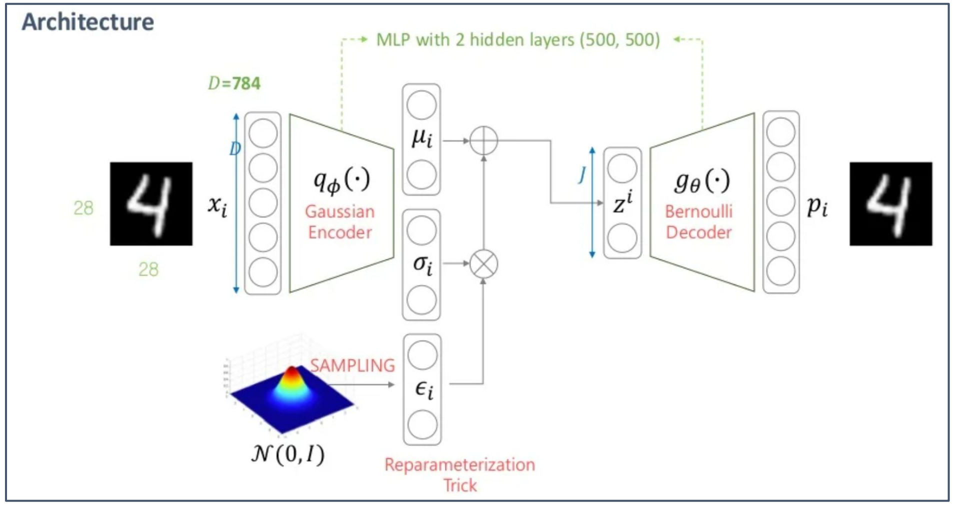
$$\boxed{q(z|x) = N\left(\mu_q(x), \Sigma_q(x)\right)}$$

02 **VAEs** - Creating a Latent Vector

$$q(z|x) = N\left(\mu_q(x), \Sigma_q(x)\right)$$

We train q to maximize the **ELBO** → the distribution of q will continuously change as x changes.

Encoder includes two neural networks: (f_μ, f_σ)



Finally, pick a **noise** from the **zero-mean Gaussian** and add and multiply it with the mean and variance outputted by f μ and f σ to create the **sampled latent vector z**

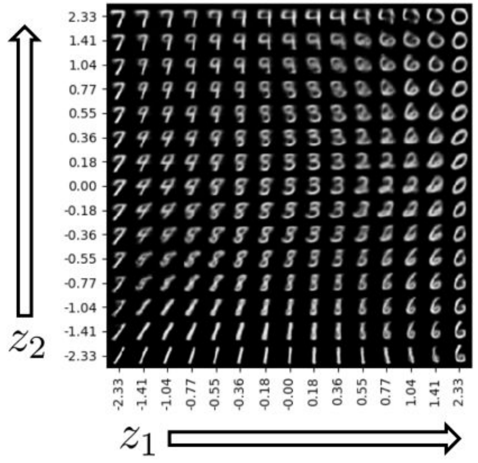
$$z = \mu(x) + \sigma(x) \times \epsilon, \quad \epsilon \sim N(0, 1)$$

➔ Thus, even if x is the same z can differ due to the noise drawn from the zero-mean Gaussian when creating z. (The reason the term **'variational'** is prefixed in VAE)

02 VAEs - Performance

➢ Based on the proposed scheme, variational autoencoder successfully generates images:

➢ Interpolation of latent variables induce **transitions** in generated images:



GANs

03 **GANs -** Implicit Deep Generative Model

Deep Generative Model

Explicit Density Modeling

Implicit Density Modeling

(Approximate Density)
Variational Autoencoder

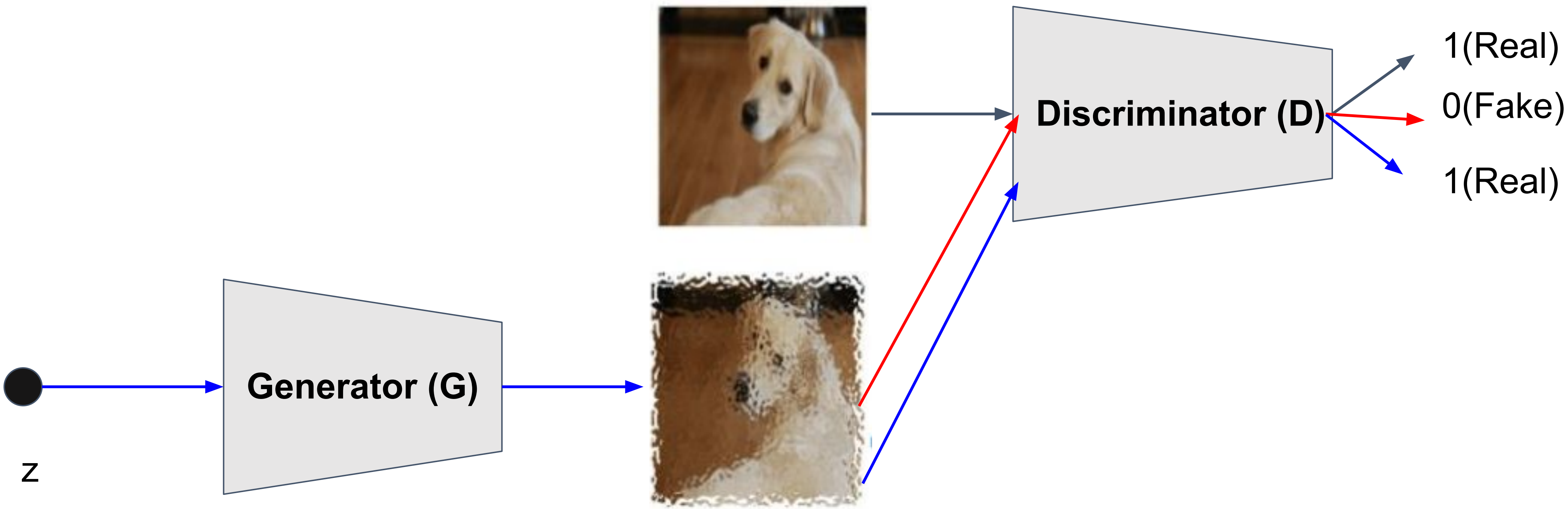
(Direct Sampling)
Generative Adversarial Network (GAN)

What is Implicit Deep Generative model?

- Unlike explicit models, implicit models do not explicitly define its density. Meaning, they do not provide an explicit likelihood for the data.

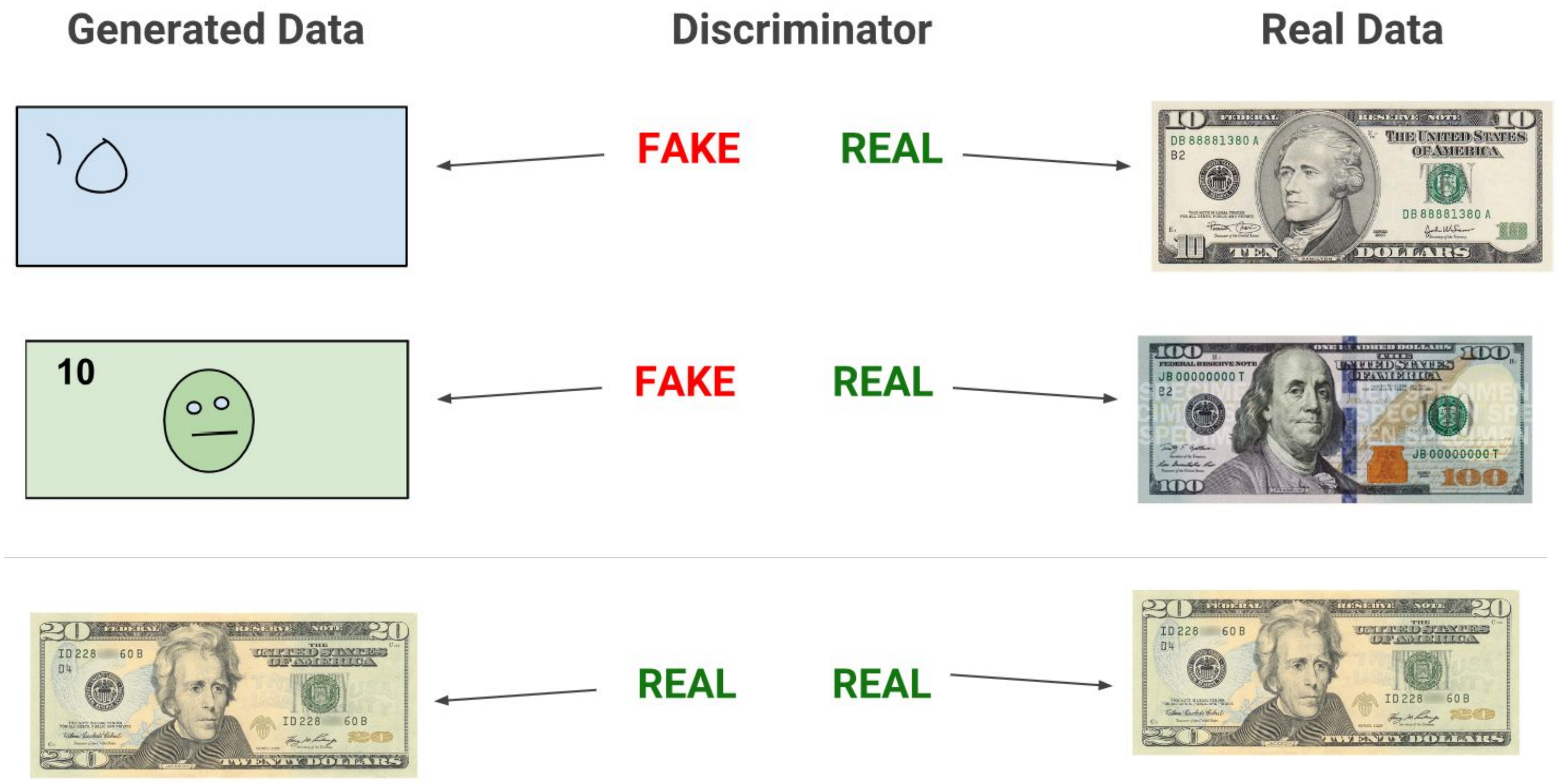
- Implicit density models define a stochastic procedure that can directly generate data.

GANs - Overview of GAN Structure



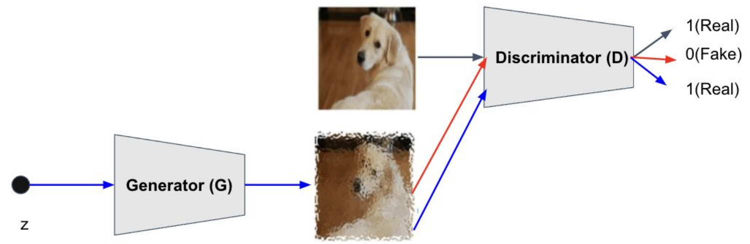
This is the only place where sampling happens

GANs - Overview of GAN Structure



03

GANs - Formalization of GAN



Real-World Data Training:

- The discriminator (D) aims to maximize the probability of correctly labeling real data as "True" (D(x) = 1).

Objective Function:

- The GAN is defined by a value function **V(D, G)** that both players (the generator G and the discriminator D) aim to optimize.
- It is a combination of two terms representing the **binary cross-entropy loss for both real and generated data:**

$$V(D,G) = E_{x\sim p_{data(x)}}[logD(x)] + E_{z\sim p_z(z)}[\log(1 - D(G(z))\,]$$

$$\min_G \max_D V(D,G)$$

G: should minimize log(1-D(G(x))), maximize D(G(x))
D: should minimize D(G(x))

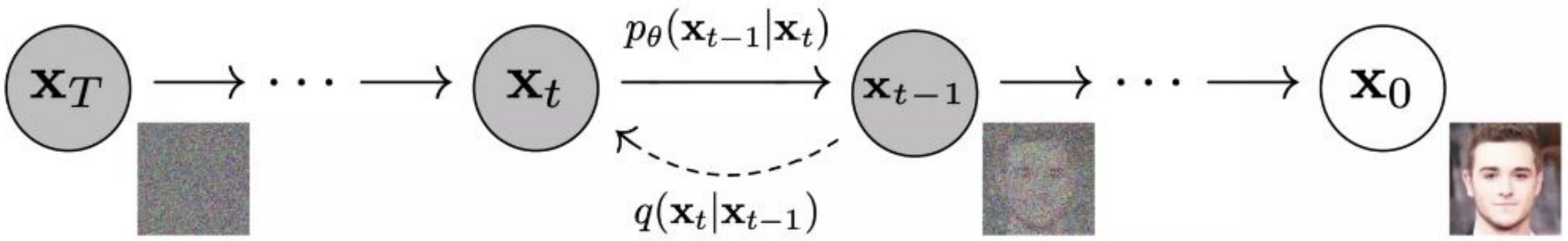
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \boxed{\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].}$$

The discriminator tries to maximize this function, while the generator tries to minimize it.

Diffusion models

➢ Diffusion model tries to learn the reverse of noise generation procedure

Forward step(q): Iteratively add noise to the original sample



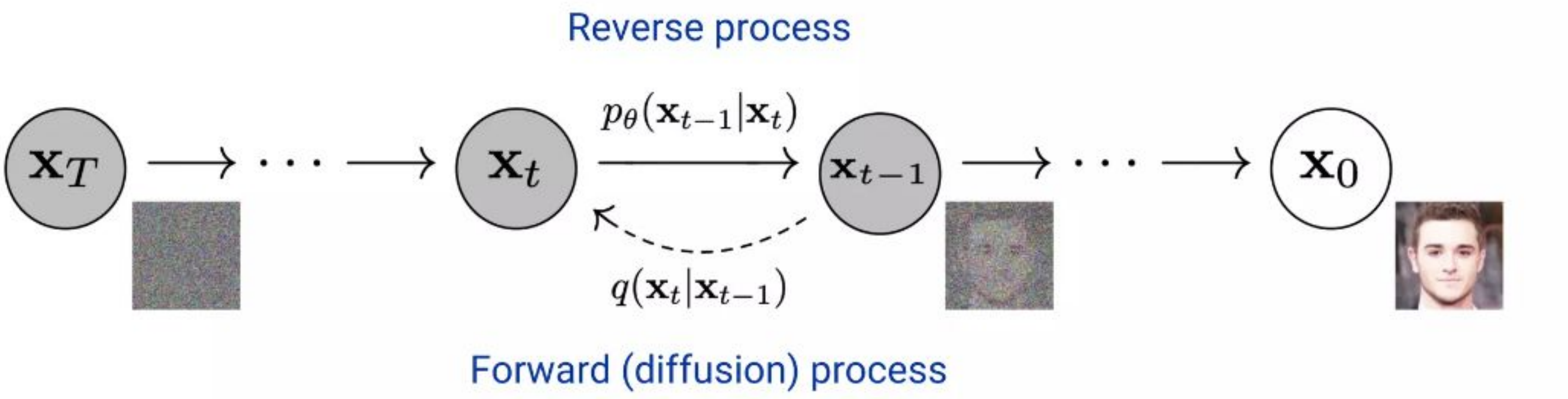
Forward (diffusion) process

Diffusion Models - General procedure

➢ Diffusion model tries to learn the reverse of noise generation procedure

Forward step(q): Iteratively add noise to the original sample

Reverse step(pθ): Recover the original sample from the noise (generation happens here)



ELBO of diffusion models

$$\begin{aligned} logp(\mathbf{x}) \geq & \mathbb{E}_{q(x_1|x_0)}[logp_\theta(\mathbf{x}_0|\mathbf{x}_1)] - \\ & D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)) - \\ & \sum_{t=2}^{T} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}[D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))] \\ & = L_0 - L_T - \sum_{t=2}^{T} L_{t-1} \end{aligned}$$

Simplified Version of the loss function:

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

04 **Diffusion Models -** training algorithm

(1) We take a random sample x0 from the real unknown and complex data distribution **q(x0)**

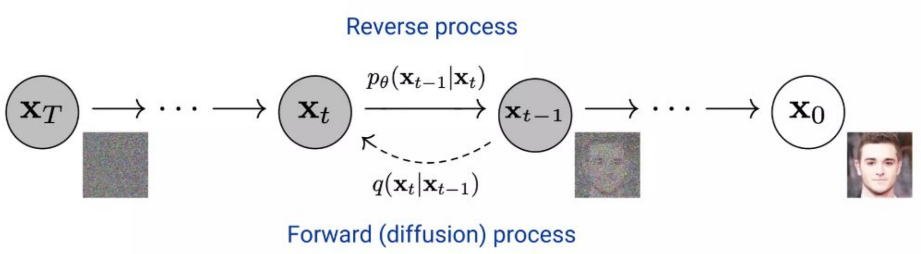
(2) Then, we sample a noise level t uniformly between 1 and T (i.e., a random time step)

(3) We sample some noise from a Gaussian distribution and corrupt the input by this noise at level t using the nice property defined above.

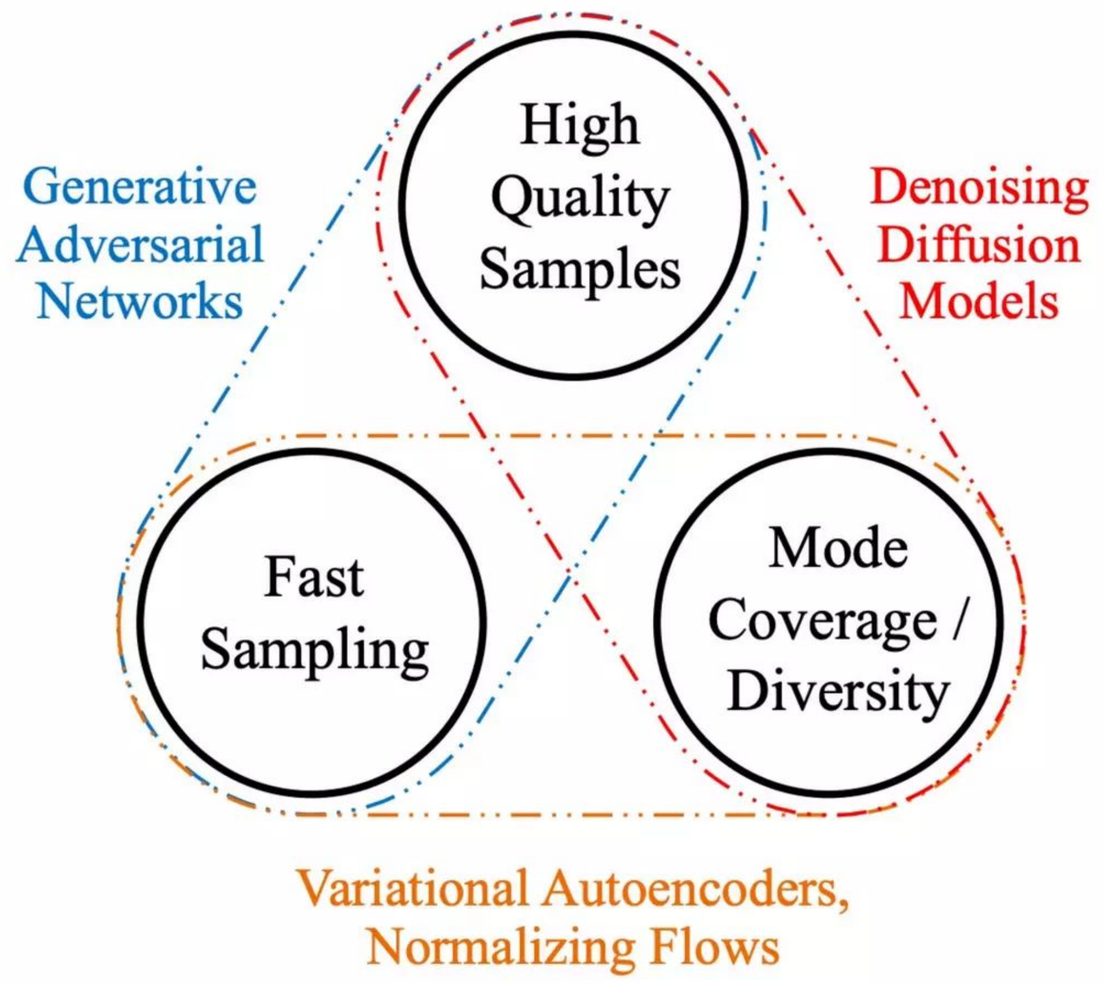
(4) The **neural network is trained to predict this noise** based on the corrupted image xt

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
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



Trilemma of Generative models: Quality vs. Diversity vs. Speed



Which model should we use?

VAEs (Variational Autoencoders):

- Good balance of diversity and speed, but generally **lower in output quality** compared to GANs and Diffusion Models.

GAN (Generative Adversarial Networks):

- **Fast** and capable of producing high-quality outputs.

Diffusion Models:

- Excellent in quality, with high diversity, but slower.

What's Next?

- ➢ Implementation
- ➢ Research results and review

Thank You!

Q & A