Generative Models
Discriminative vs Generative

Understanding Machine Learning

Archetype: Naïve Bayes

probabilistic model:

$$P(Y|X_1,\cdots,X_p) = \frac{P(Y,X_1,\cdots,X_p)}{P(X_1,\cdots,X_p)} = \frac{P(Y)P(X_1,\cdots,X_p|Y)}{P(X_1,\cdots,X_p)} \propto P(Y)P(X_1,\cdots,X_p|Y)$$

Bayes' rule   constant   to be estimated

approach:

1. estimate $P(Y,\boldsymbol{X})$ → generative model (can be used to generate new samples)
2. calculate $P(Y|\boldsymbol{X})$ from $P(Y,\boldsymbol{X})$ → used for discriminative task (classification)

Independence Assumption

(naïve) assumption: conditional independence of features given target

$$P\left(X_j | Y, X_1, \cdots, X_{j-1}, X_{j+1}, \cdots, X_p\right) = P\left(X_j | Y\right)$$

$$\implies P\left(Y | X_1, \cdots, X_p\right) = \frac{P(Y) \prod_{j=1}^{p} P\left(X_j | Y\right)}{P\left(X_1, \cdots, X_p\right)}$$
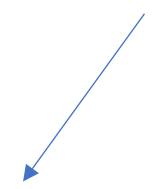
→ independent feature contributions (ignoring feature correlations)
→ robust against curse of dimensionality

Estimation of Feature Contributions

separate estimations of $P(X_j|Y)$ for each feature

requires assumption of distributions (e.g., Gaussian naïve Bayes) or non-parametric methods (kernel density estimation)

Gaussian feature likelihoods: $P(x_{ij}|y) = \dfrac{1}{\sqrt{2\pi\sigma_{y,j}^2}} \exp\left(-\dfrac{(x_{ij}-\mu_{y,j})^2}{2\sigma_{y,j}^2}\right)$

parameter estimation (e.g., mean and variance of Gaussians) can be done with maximum likelihood method (y known in training)

$\rightarrow$ no Bayesian methods needed

Maximum a Posteriori Classification

$$\hat{y}_i = \underset{y}{\mathrm{argmax}}\, P(y) \prod_{j=1}^{p} P(x_{ij}|y)$$

despite potentially inaccurate probability estimates (due to naïve independence assumption), good identification of correct class via maximum probability

→ bad for regression tasks (if independence assumption is too naïve, i.e., features are correlated)
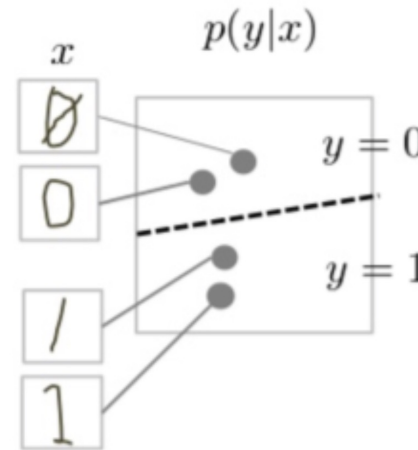
Generative vs Discriminative Models

generative models: predict joint probability $P(Y, \boldsymbol{X})$ (what allows to create new data samples) or directly generates new data samples
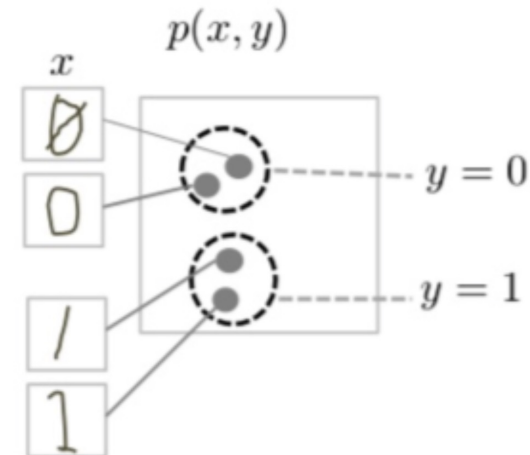
discriminative models: predict conditional probability $P(Y|\boldsymbol{X})$ or directly output (label for classification, real value for regression)

task of generative models more difficult: model full data distribution rather than merely find patterns in inputs to distinguish outputs

discriminative model

generative model

$p(y|x)$

$p(x,y)$

x

x

$y = 0$

$y = 1$

$y = 0$

$y = 1$

source

Naïve Bayes and Logistic Regression

generative-discriminative pair of classification algorithms

- binary case: logit of naïve Bayes' outputs, $\log\left(\frac{P(y_i=1|\boldsymbol{x}_i)}{P(y_i=0|\boldsymbol{x}_i)}\right)$, corresponds to output of logistic regression's linear predictor

- for discrete inputs or Gaussian naïve Bayes: naïve Bayes can be reparametrized as linear classifier

for discriminative task: identical in asymptotic limit (infinite training samples) if independence assumption holds (otherwise naïve Bayes less accurate)

naïve Bayes has greater bias but lower variance than logistic regression → to be preferred for scarce training data (if bias, i.e., independence assumption, correct)

Data Generation

generative models can be used for discriminative tasks (although potentially inferior to direct discriminative methods)
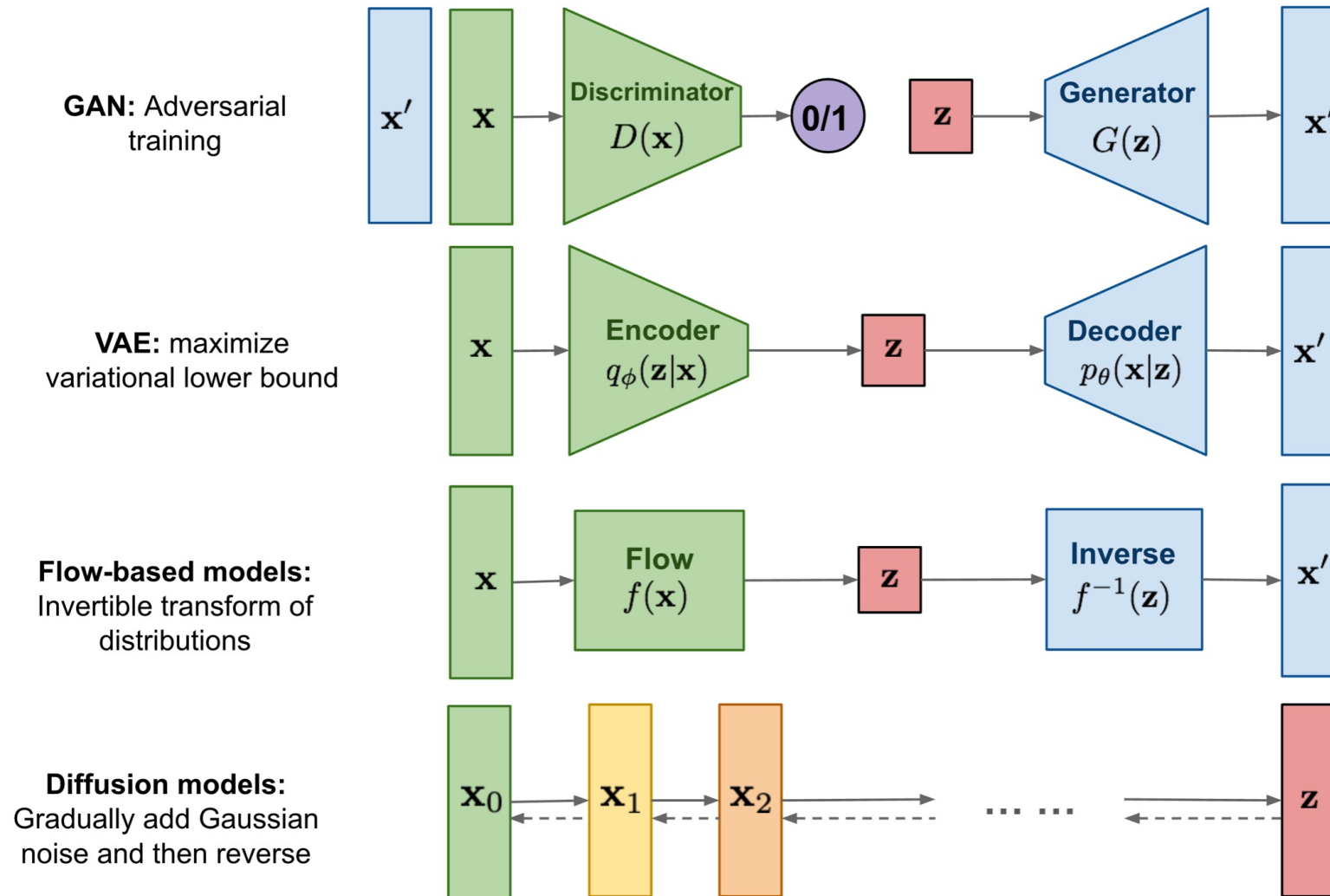
but generative methods do more than discriminative ones: model full data distribution

→ allows generation of new data samples (can be images, text, video, audio, proteins, materials, time series, structured data, …)

large (auto-regressive) language models examples of generative models

Different Types of Generative Models

GAN: Adversarial training

VAE: maximize variational lower bound

Flow-based models: Invertible transform of distributions

Diffusion models: Gradually add Gaussian noise and then reverse
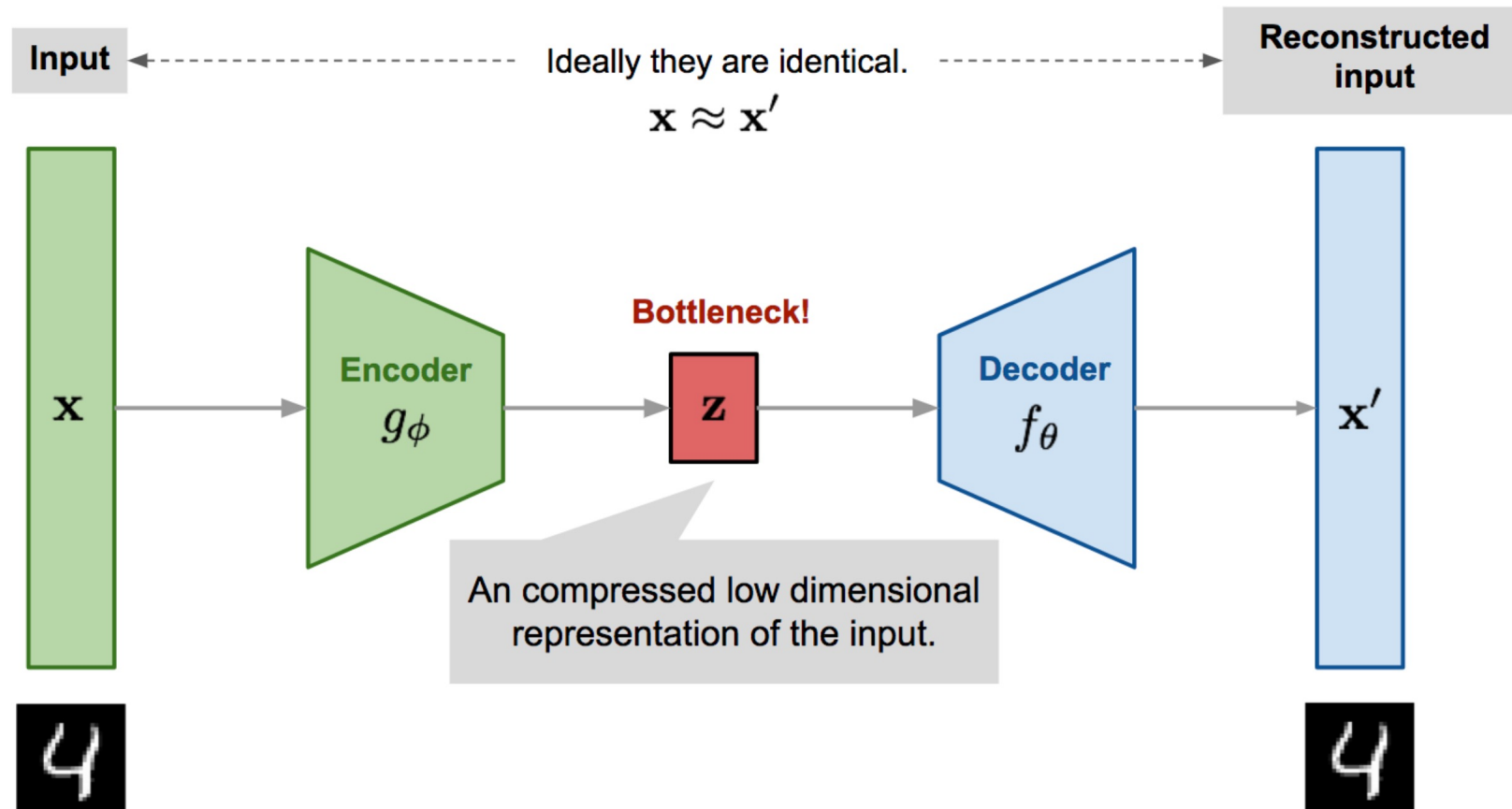
source

Variational Autoencoders (VAE)

Recap: Autoencoder

(deep) encoder network

(deep) decoder network

learned together by minimizing differences between original input and reconstructed input (expressed as losses)

compressed intermediate representation: dimensionality reduction

Input $\longleftarrow$ --- Ideally they are identical. --- $\longrightarrow$ Reconstructed input

$$\mathbf{x} \approx \mathbf{x}'$$

$\mathbf{x}$

Encoder g_ϕ

Bottleneck!

$\mathbf{z}$

Decoder f_θ

$\mathbf{x}'$

An compressed low dimensional representation of the input.

Autoencoder Architecture for Generative Tasks

goal: generation of variations of input data rather than compressed representation

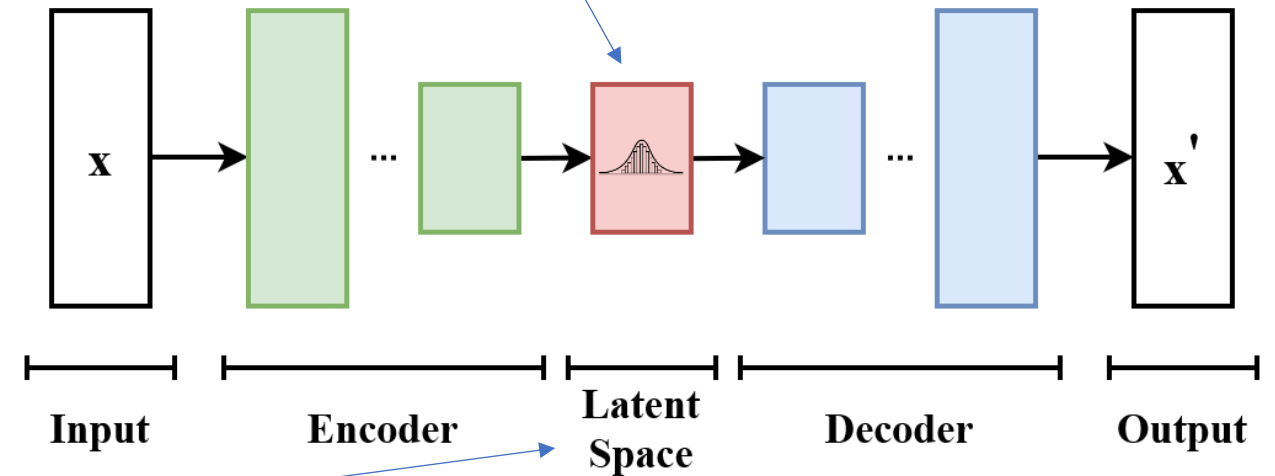→ learn variational distribution instead of identity function

to be precise: parametrized variational distribution of latent encoding variables $\boldsymbol{z}$

prior (simple distribution, in usual VAE: Gaussian): $p_{\boldsymbol{\theta}}(\boldsymbol{z})$

posterior: $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = \dfrac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{z})}{\int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p_{\boldsymbol{\theta}}(\boldsymbol{z})d\boldsymbol{z}}$

$p_{\boldsymbol{\theta}}(\boldsymbol{x})$: mixture of Gaussians

from which to sample

from wikipedia

Variational Bayesian Method

Encoder and Decoder Networks

encoder: find posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$

unfortunately, generally intractable (integral over $\boldsymbol{z}$ expensive)

→ approximate by $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$

VAE: $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ expressed by neural network with weights $\boldsymbol{\phi}$

→ amortized inference:
$q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ learned in training, $\boldsymbol{z}$ inferred from $\boldsymbol{x}$ in prediction (sharing variational parameters across all data points)
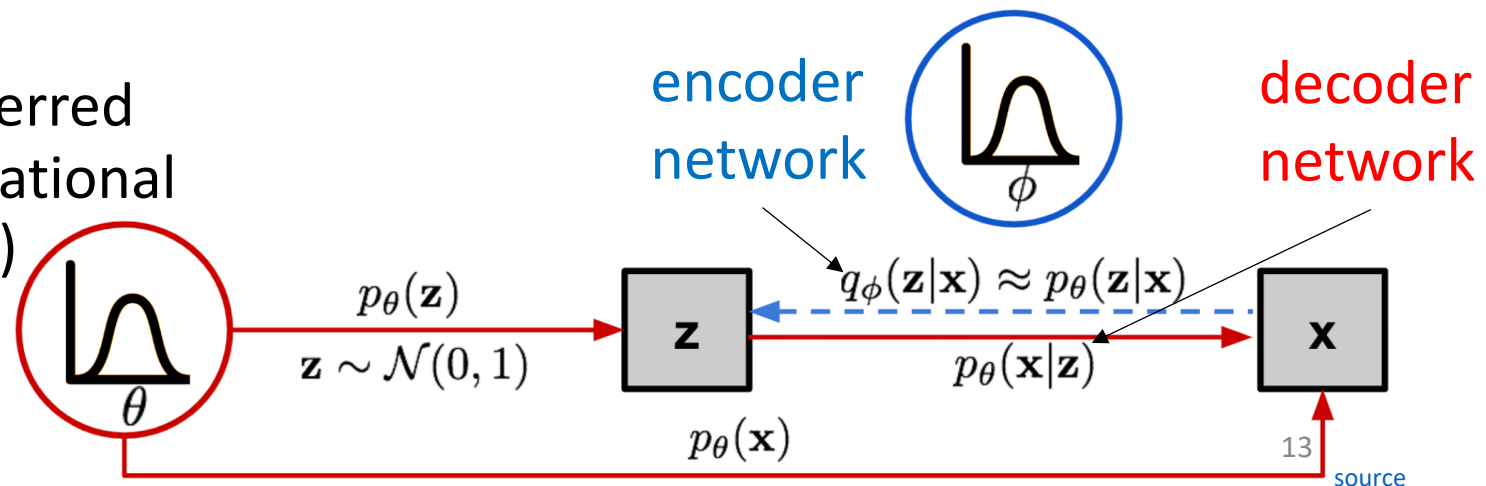
in VAE: network weights $\boldsymbol{\theta}$

decoder: generate new sample $\boldsymbol{x}_i$

1. sample $\boldsymbol{z}_i$ (from Gaussian)

2. generate $\boldsymbol{x}_i$ (similar to real data)

→ maximize: $p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}_i|\boldsymbol{z})\, p_{\boldsymbol{\theta}}(\boldsymbol{z})d\boldsymbol{z}$

(expensive → use only likely codes $\boldsymbol{z}$ given input $\boldsymbol{x}$: need for encoder)

encoder network

decoder network

$q_{\phi}(\mathbf{z}|\mathbf{x}) \approx p_{\theta}(\mathbf{z}|\mathbf{x})$

$p_{\theta}(\mathbf{z})$

$\mathbf{z} \sim \mathcal{N}(0,1)$

$\mathbf{z}$

$p_{\theta}(\mathbf{x}|\mathbf{z})$

$\mathbf{x}$

$p_{\theta}(\mathbf{x})$

θ

13

VAE Loss: ELBO

VAE loss function to be minimized according to network weights:

$$L(\boldsymbol{x}_i; \boldsymbol{\theta}, \boldsymbol{\phi}) = -\ln p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) + D_{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}_i)||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}_i)\right)$$

maximize likelihood of observed data (minimize reconstruction error)

and

minimize difference of approximation $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}_i)$ to exact posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}_i)$
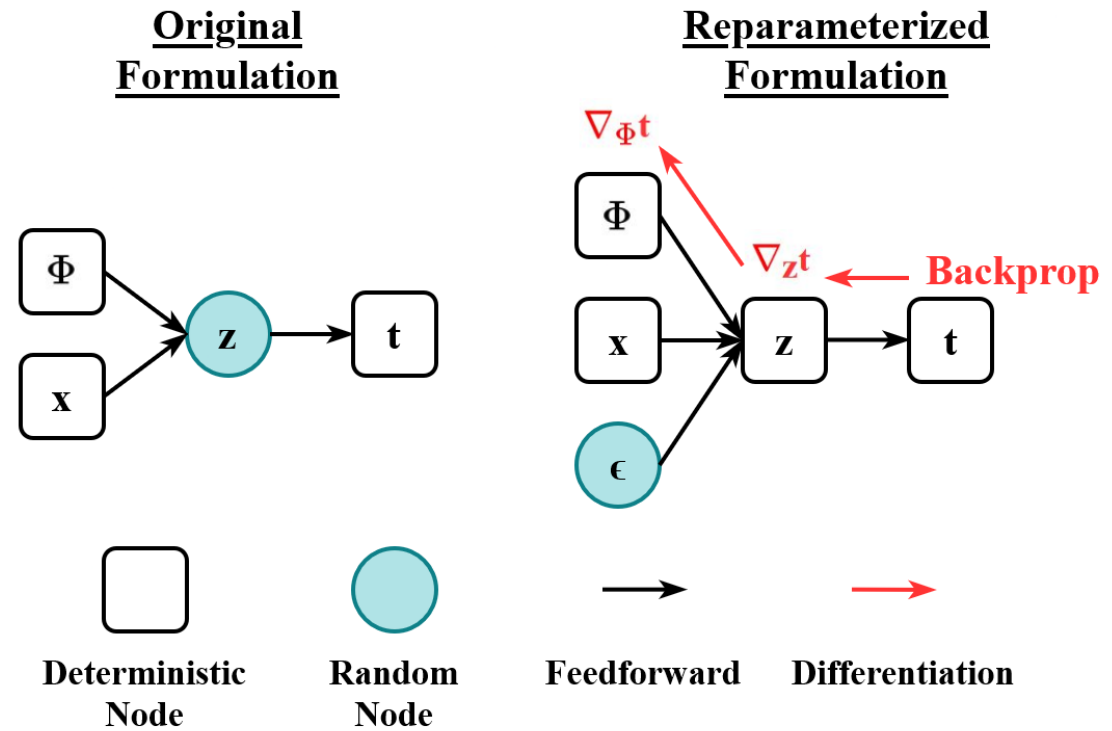
can be interpreted as regularizer

corresponds to maximizing evidence lower bound (ELBO), i.e., maximizing lower bound of probability to generate real data sample:

$$\ln p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \geq \ln p_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - \underbrace{D_{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}_i)||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}_i)\right)}_{\text{non-negative}} = E_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}_i)}\left[\ln \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}_i)}\right]$$
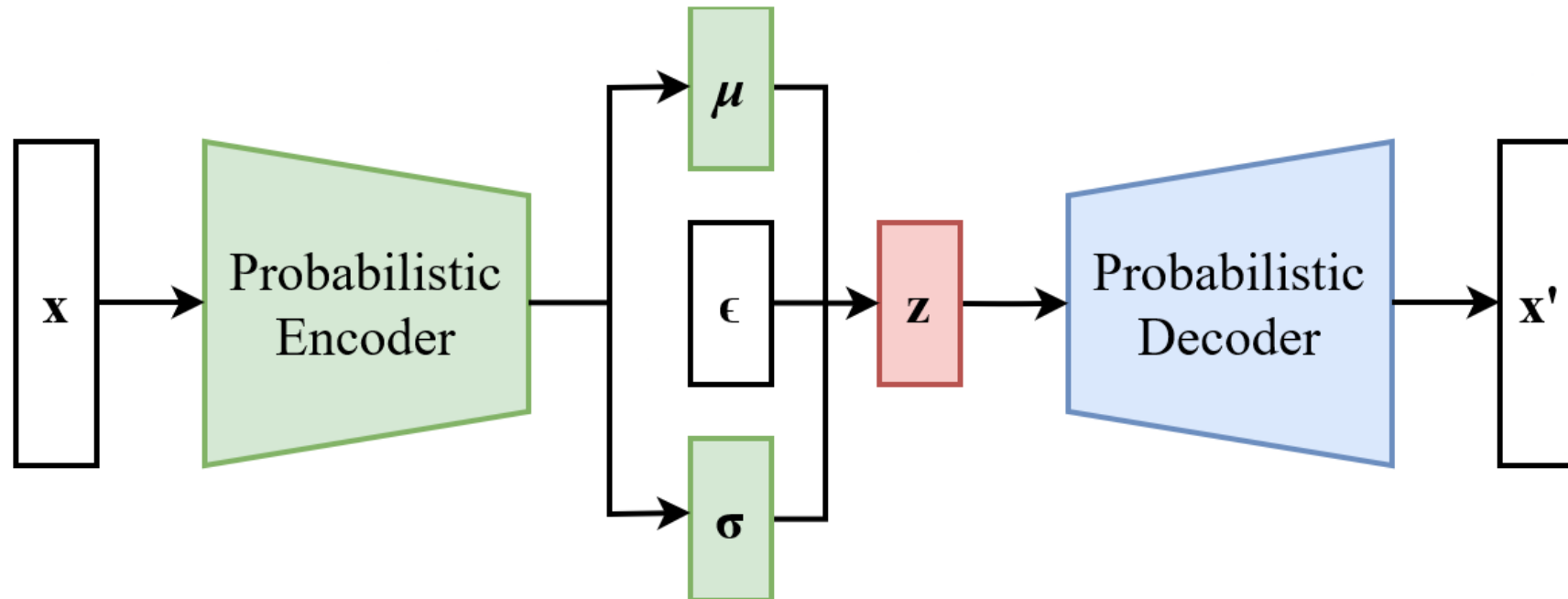
Reparameterization Trick

...

from wikipedia

Putting Everything Together

from wikipedia
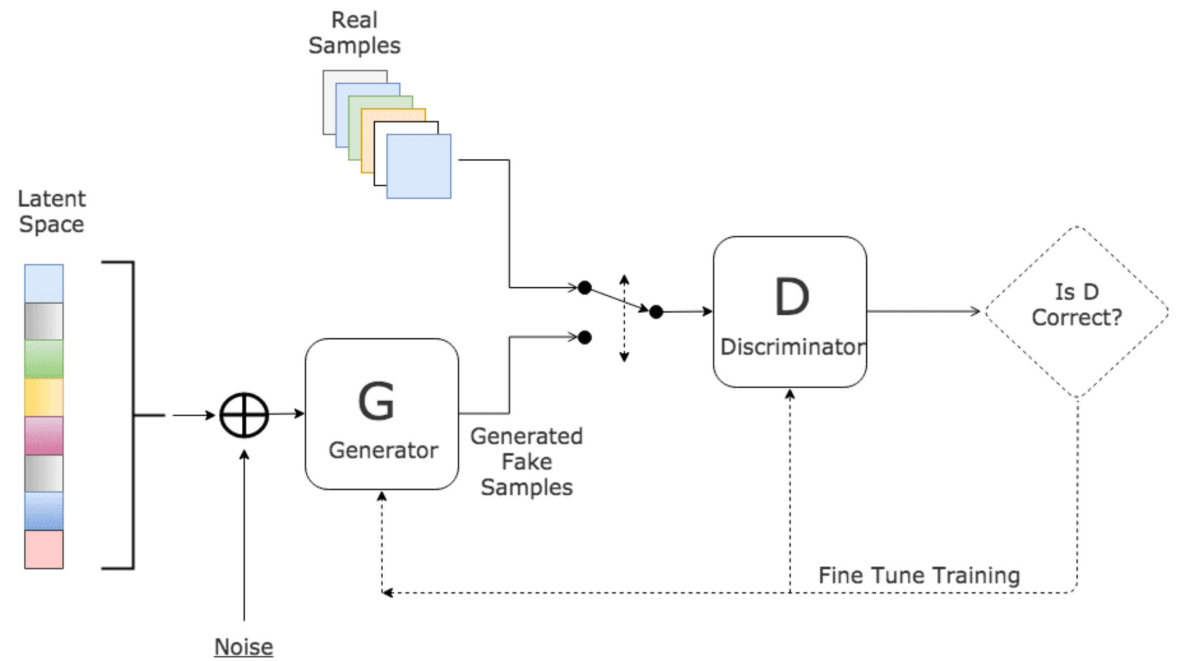
Generative Adversarial Networks (GAN)

Zero-Sum Game

- …
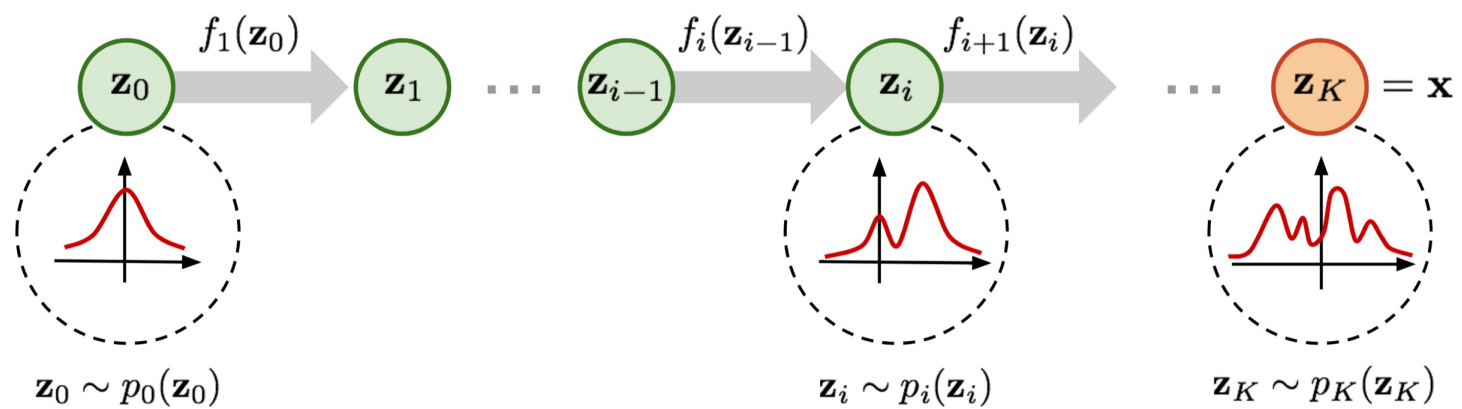- two neural networks playing a zero-sum game …

- …
- … noise

Issues in GANs

- …
- …potentially unstable training and less diversity in generation

Flow-Based Methods

- ...

$$f_1(\mathbf{z}_0) \qquad f_i(\mathbf{z}_{i-1}) \qquad f_{i+1}(\mathbf{z}_i)$$

$$\mathbf{z}_0 \quad \mathbf{z}_1 \quad \cdots \quad \mathbf{z}_{i-1} \quad \mathbf{z}_i \quad \cdots \quad \mathbf{z}_K = \mathbf{x}$$

$$\mathbf{z}_0 \sim p_0(\mathbf{z}_0) \qquad \mathbf{z}_i \sim p_i(\mathbf{z}_i) \qquad \mathbf{z}_K \sim p_K(\mathbf{z}_K)$$

- specialized architectures to construct reversible transform

Diffusion Models

- inspired by non-equilibrium thermodynamics
- Markov chain of diffusion steps to slowly add random noise to data
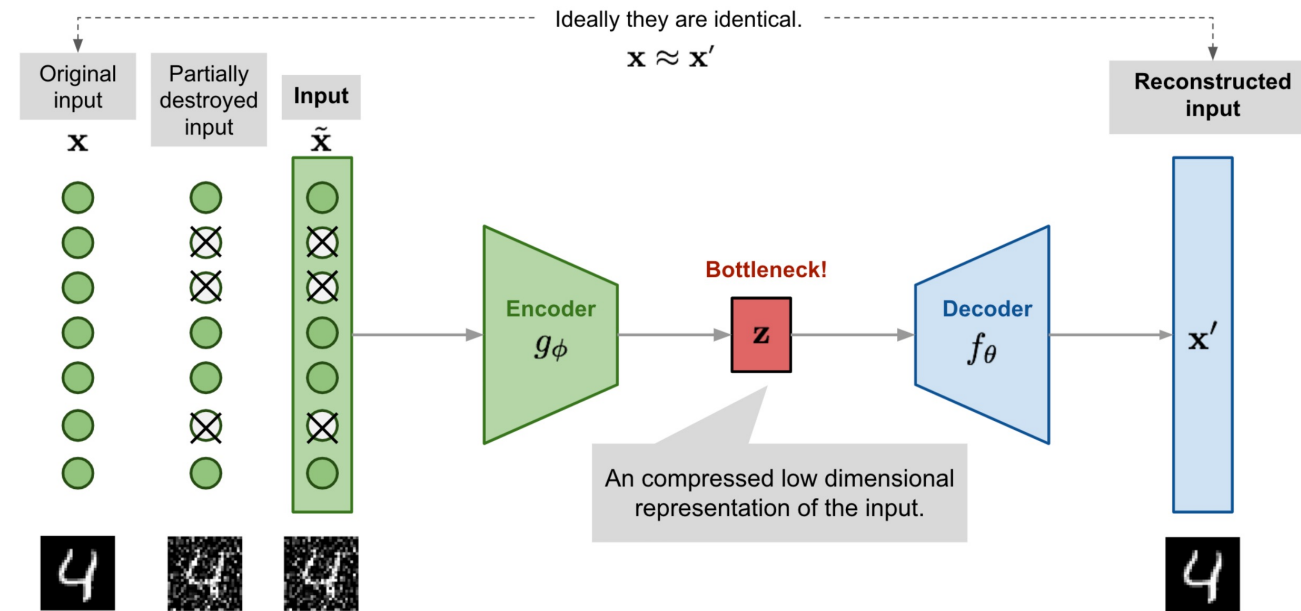- …chain of denoising autoencoders…

Denoising Autoencoder

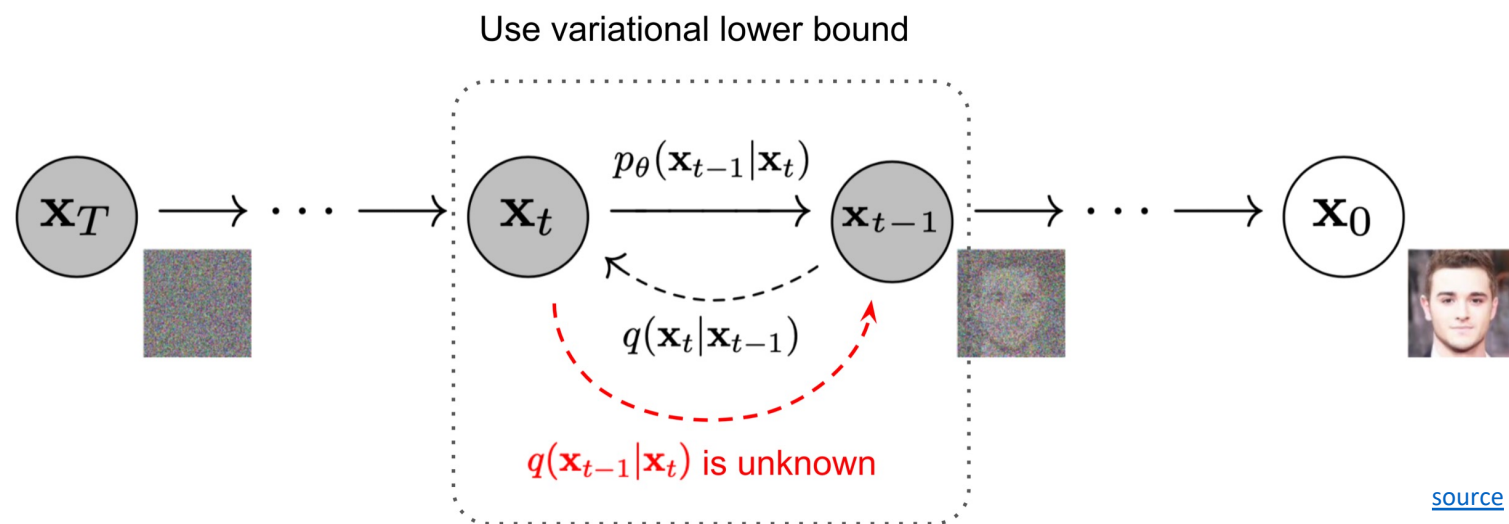goal: avoid overfitting and improve robustness of plain autoencoder

...

...

similar to dropout

- then learn to reverse the diffusion process to construct desired data samples from the noise
- Unlike VAE or flow models, diffusion models are learned with a fixed procedure and the latent variable has high dimensionality (same as the original data).

- …

Use variational lower bound

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

$$\mathbf{x}_T \longrightarrow \cdots \longrightarrow \mathbf{x}_t \longrightarrow \mathbf{x}_{t-1} \longrightarrow \cdots \longrightarrow \mathbf{x}_0$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is unknown

Image Generation

DALL-E 2

…

[Stable Diffusion](#)

[DreamStudio](#)

A dream of a distant galaxy, by Caspar David Friedrich, matte painting trending on artstation HQ

Literature

papers:
- variational autoencoder
- GAN
- normalizing flows
- latent diffusion

Movie-like Intelligence

emergent capabilities of complex systems almost impossible to foresee

mini examples in contemporary ML:

- large language models

- multi-agent reinforcement learning

one idea: reward is enough

philosophical: emotions or consciousness might also occur as emergent capabilities

34