LG전자 Deep Learning 과정

Variational Autoencoders

Gunhee Kim

Computer Science and Engineering

서울대학교
SEOUL NATIONAL UNIVERSITY

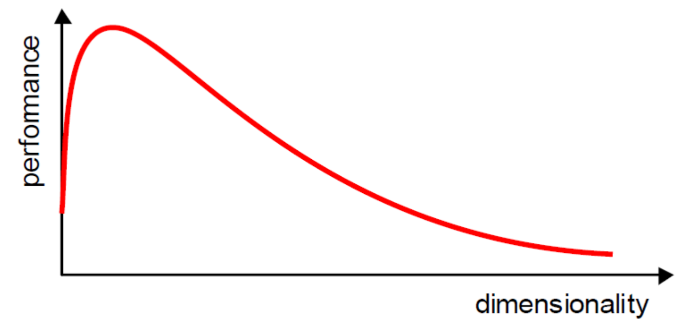
Outline

- Autoencoders
- Variational Autoencoders

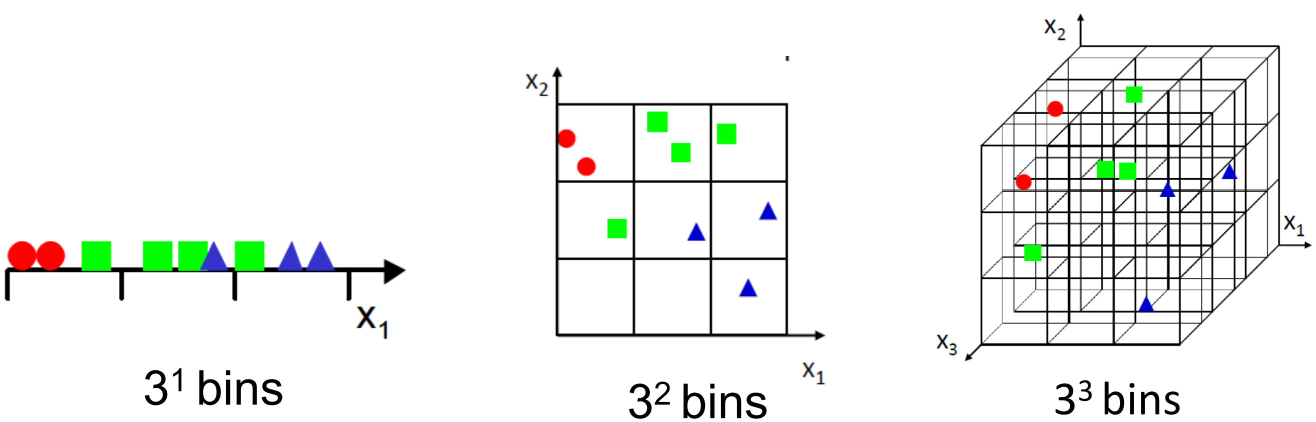
Curse of Dimensionality

Increasing the number of features will not always improve classification accuracy

- In practice, the inclusion of more features might lead to worse performance



The number of training examples required increases exponentially with dimensionality d (i.e., k^d)



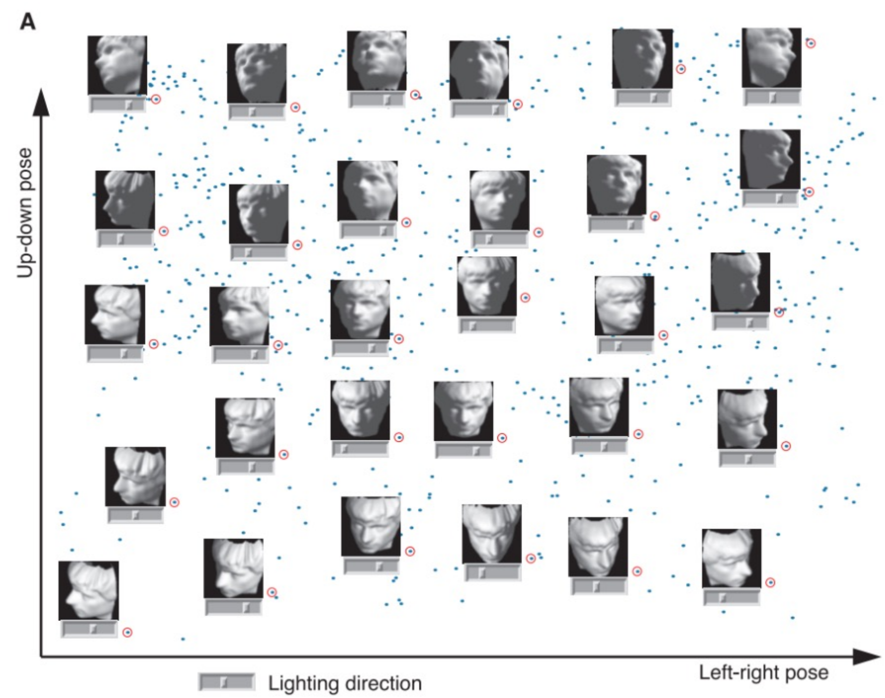
[Source: George Bebis]

Why Dimensionality Reduction?

Feature selection: some features may be irrelevant

Visualization: especially for high dimensional data

Intrinsic dimensionality: smaller than # of features



[Source: J. B. Tenenbaum]

Feature Selection vs Extraction

Feature selection (supervised)

- Chooses a subset of the original features

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_p \end{bmatrix} \longrightarrow \boldsymbol{y} = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \cdot \\ x_{ik} \end{bmatrix}$$

$$k \ll p$$

Feature extraction (unsupervised)

- Finds a set of new features (i.e., through some mapping $f(x)$) from the existing features
- The mapping $f(x)$ could be linear or non-linear

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_p \end{bmatrix} \xrightarrow{f(x)} \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ y_k \end{bmatrix}$$

$$k \ll p$$

Feature Extraction

Often called dimensionality reduction or manifold learning

How to find an optimum mapping $y = f(x)$ is equivalent to optimizing an objective function?

Minimize information loss

- The goal is to represent the data as accurately as possible (i.e., no loss of information) in the lower-dimensional space

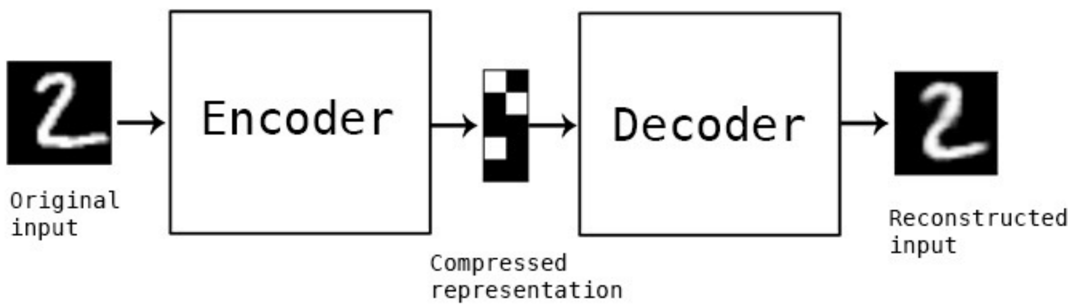
Maximize discriminatory information

- The goal is to enhance the class-discriminatory information in the lower-dimensional space

Autoencoders

An unsupervised neural network model

- Used for dimensionality reduction (e.g., feature selection and extraction)
- Lossy dimensionality reduction with few hidden units
- e.g., 10×10 images as input, and 50 hidden units
  → compressed representation of images



- Encoder: represent (or compress) input data into a low-dim code
- Decoder: decompress a code into a data
- Encoder and decoder are implemented by neural networks

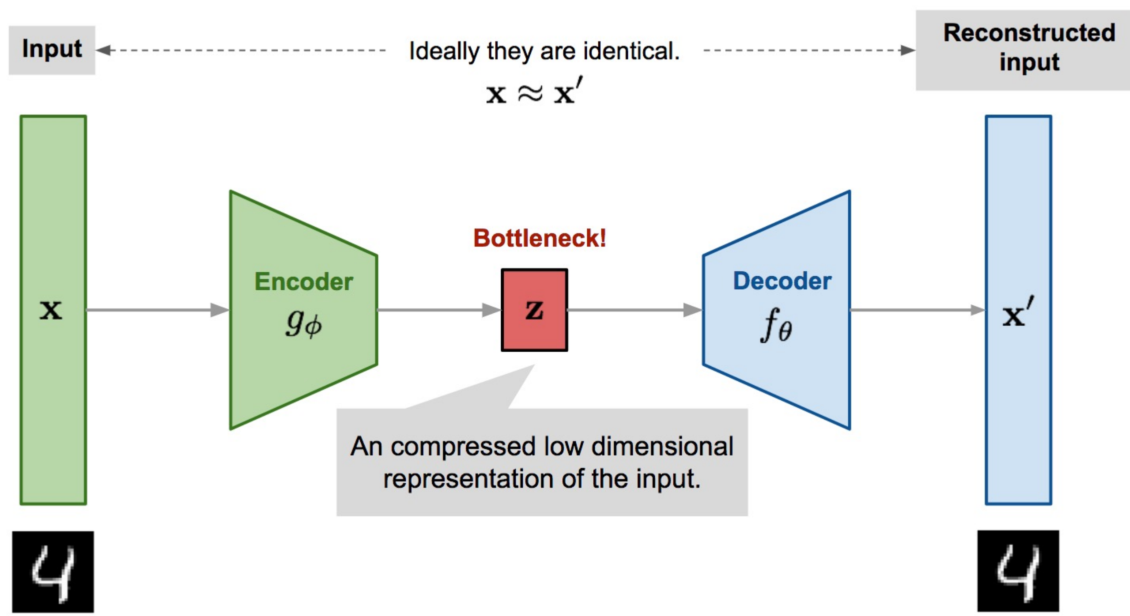
Formulation

Objective for dimensionality reduction

- Encoder $g_\phi: \mathcal{X} \longrightarrow \mathcal{Z}$ and decoder $f_\theta: \mathcal{Z} \longrightarrow \mathcal{X}$

$$\phi, \theta = \min_{\phi,\theta} \left\| X - (f_\theta \circ g_\phi) X \right\|^2$$

- Feature space $\mathcal{Z}$ has often lower dimensionality than input space $\mathcal{X}$



[Source: Lilian Weng]

Formulation

A simple NN model

- An input $\boldsymbol{x} \in \mathbb{R}^d = \mathcal{X}$ maps to a code (i.e., latent variable) $\boldsymbol{z} \in \mathbb{R}^p = \mathcal{Z}$, which is reconstructed to $\boldsymbol{x}'$

$$\boldsymbol{z} = \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$$

$$\boldsymbol{x}' = \sigma(\boldsymbol{W}'\boldsymbol{z} + \boldsymbol{b}')$$

- Encoder's weight matrix $\boldsymbol{W}$ and bias $\boldsymbol{b}$ could be the same with those of decoder $\boldsymbol{W}'$ and $\boldsymbol{b}'$

Training

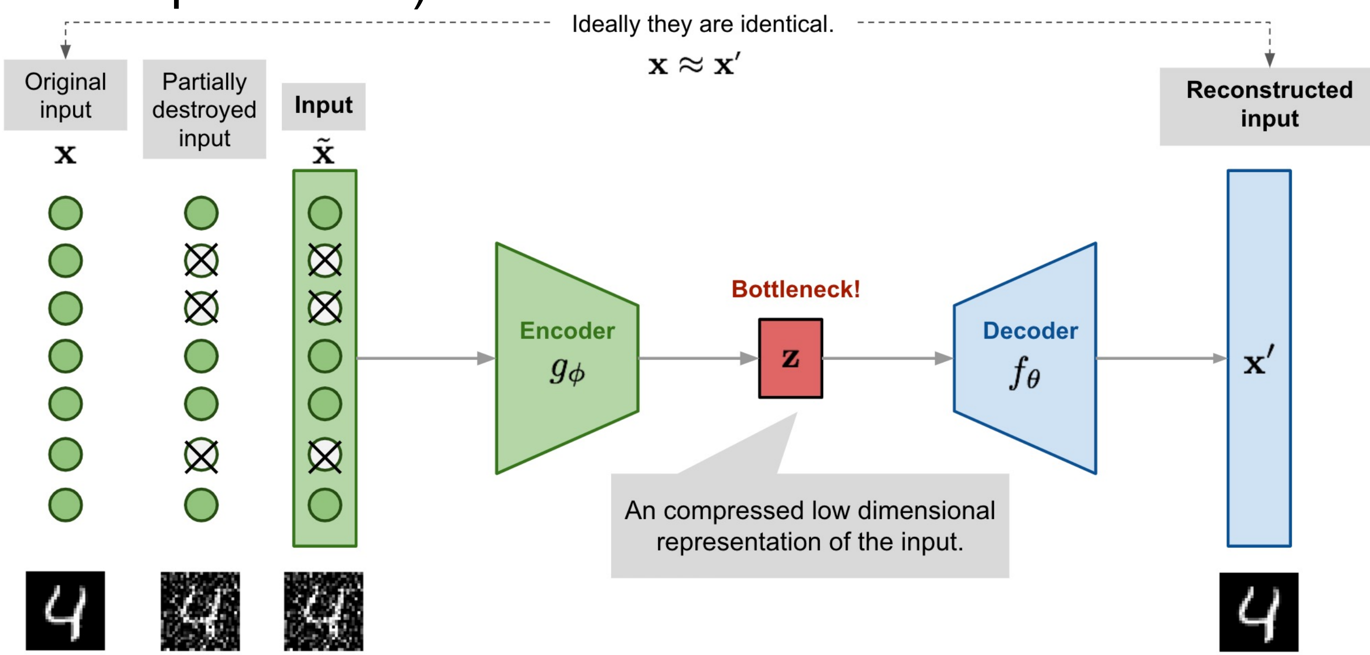
- AEs are also trained to minimize reconstruction errors (averaged over some input training set)

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|^2 = \|\boldsymbol{x} - \sigma(\boldsymbol{W}'\sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}) \ + \boldsymbol{b}')\|$$

Variants – Denoising Autoencoder

A stochastic and more robust extension

- Can avoid the risk of overfitting
- Randomly corrupt input (by adding noises to or masking some values) and let AE reconstruct its denoising one
- With fewer data, add more noise (e.g., adding noise 30% corruption level)



[Source: Lilian Weng]

Variants – Sparse Autoencoder

A sparse constraint on the hidden unit activation

- Can avoid the risk of overfitting
- Only a small number of hidden units are activated simultaneously (i.e., one hidden neuron should be inactivated most of time)
- Average activation of hidden unit i in layer l over training data

$$\hat{\rho}_i^{(l)} = \frac{1}{m}\sum_{j=1}^{m} a_i^{(l)}(x^{(j)})$$

- Set sparsity parameter $\hat{\rho}_i^{(l)} = \rho = 0.05$

Variants – Sparse Autoencoder

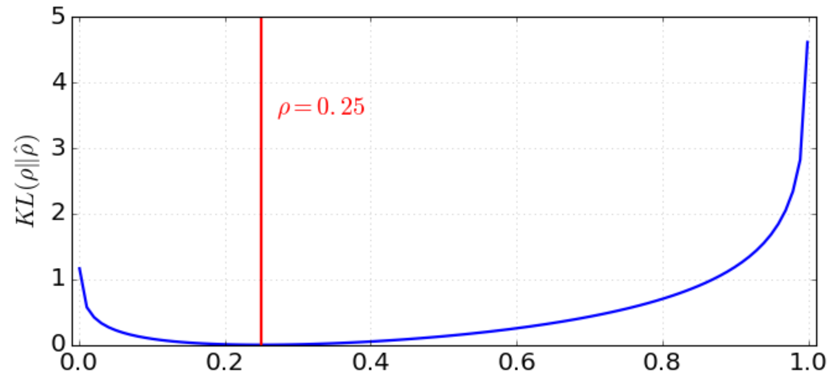
The objective

- The constraint is achieved by adding a penalty term
- The KL-divergence between two Bernoulli distributions, one with mean ρ and the other with mean $\hat{\rho}_i^{(l)}$

$$\mathcal{L} = \frac{1}{m}\sum_{j=1}^{m}(\boldsymbol{x}_i - f_\theta(g_\phi(\tilde{\boldsymbol{x}}_i)))^2 + \beta\sum_{l=1}^{L}\sum_{j=1}^{s_l} KL(\rho||\hat{\rho}_i^{(l)})$$

$$= \frac{1}{m}\sum_{j=1}^{m}(\boldsymbol{x}_i - f_\theta(g_\phi(\tilde{\boldsymbol{x}}_i)))^2 + \beta\sum_{l=1}^{L}\sum_{j=1}^{s_l} \rho\log\frac{\rho}{\hat{\rho}_i^{(l)}} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_i^{(l)}}$$

- The KL-divergence when $\rho = 0.25$ and $0 \le \hat{\rho}_i^{(l)} \le 1$



[Source: Lilian Weng]

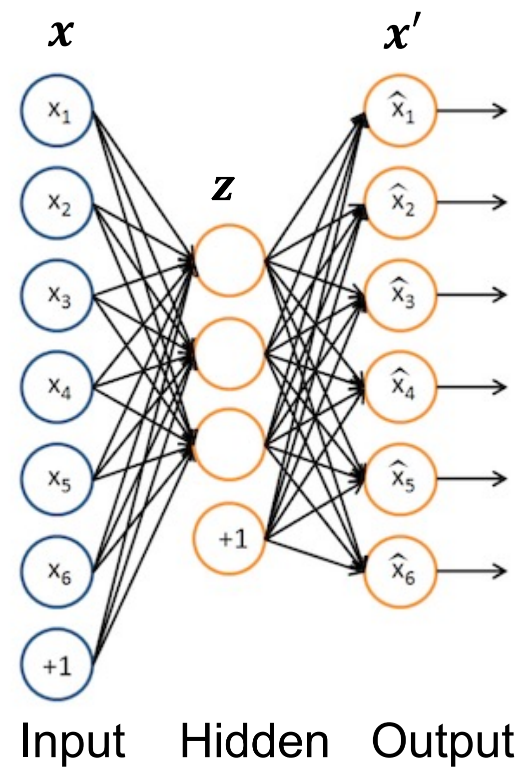
Visualization of Autoencoder

Train a sparse autoencoder on

- Use 10x10 images and 100 hidden units
- Each hidden unit i computes a function of the input

$$z_i = \sigma(\sum_{j=1}^{100} W_{ij} x_j + b_i)$$

- z_i is a non-linear feature of input $\boldsymbol{x}$
- What input image $\boldsymbol{x}$ would cause z_i to be maximally activated? (i.e., what is the feature that hidden unit i is looking for?)

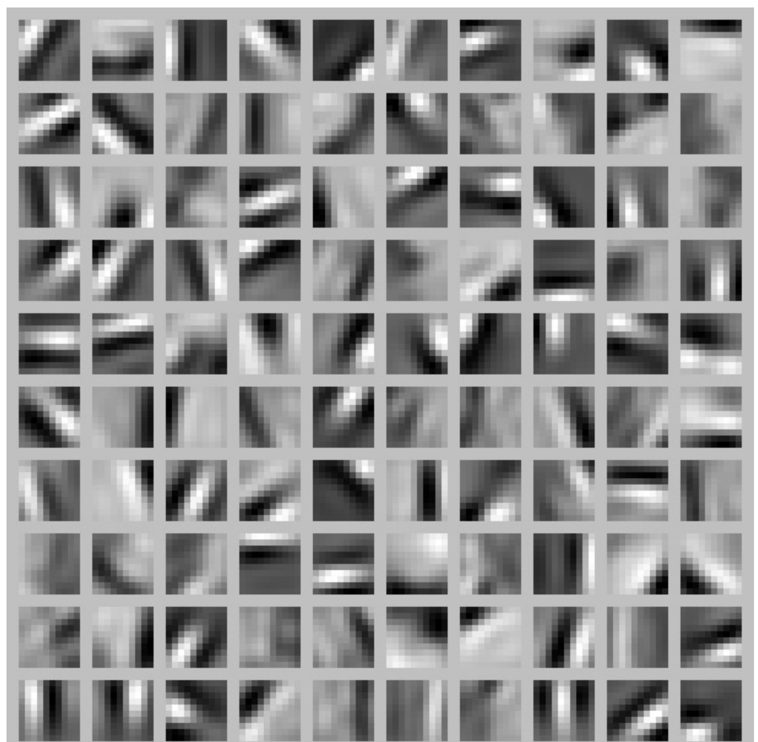


[Source: Andrew Ng]

Visualization of Autoencoder

The input that maximally activates hidden unit i is given by setting pixel x_j (for all $j = 1, \ldots, 100$ pixels)

$$x_j = \frac{W_{ij}}{\sqrt{\sum_{j=1}^{100} (W_{ij})^2}} \quad \text{with} \quad \|\boldsymbol{x}\|^2 \leq 1$$



100 such images, one per hidden unit

[Source: Andrew Ng]

Outline

- Autoencoders
- Variational Autoencoders

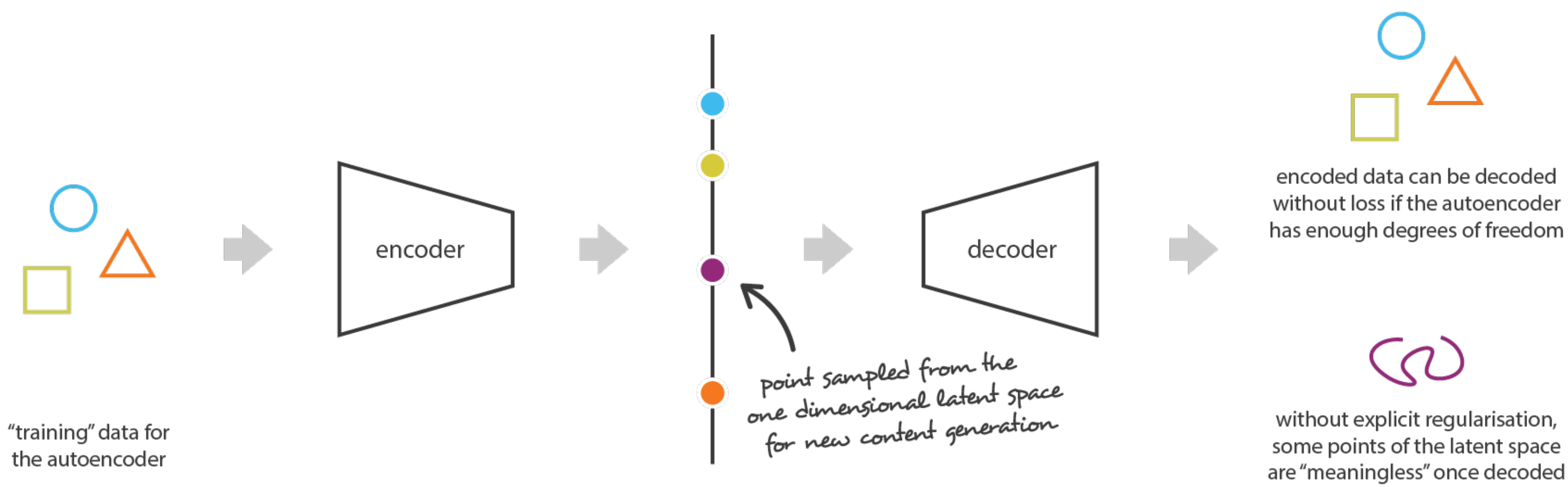
Limitation of AE for Generation

AE is trained to encode and decode with the minimum loss

- Do NOT care how the latent space is organized

Severe overfitting

- Due to too large degree-of-freedom, some points of the latent space will give meaningless content once decoded



[Source: Joseph Rocca]

Idea of VAE

VAE follows AE

- Composed of both an encoder and a decoder
- Trained to minimize the reconstruction error between the input and the encoded-decoded data

Instead, its training is regularized

- The latent space has good properties that enable generative process
- One key difference: instead of encoding an input as a single point, we encode it as a **distribution** over the latent space

Idea of VAE

Basic steps

- (1) Encode the input as distribution over the latent space
- (2) Sample a point from the latent distribution
- (3) Decode the sampled point
- (4) Backpropagate the reconstruction error through the network

AE	input $\boldsymbol{x}$	$\longrightarrow$	latent representation $\boldsymbol{z} = e(\boldsymbol{x})$	$\longrightarrow$	input reconstruction $d(\boldsymbol{z})$		
VAE	input $\boldsymbol{x}$	$\longrightarrow$	latent representation $q(\boldsymbol{z}\|\boldsymbol{x})$	$\longrightarrow$	sampled input reconstruction $\boldsymbol{z} \sim q(\boldsymbol{z}\|\boldsymbol{x})$	$\longrightarrow$	input reconstruction $d(\boldsymbol{z})$

[Source: Joseph Rocca]

VAE from AE Perspective

Consists of an encoder, a decoder, and a loss function

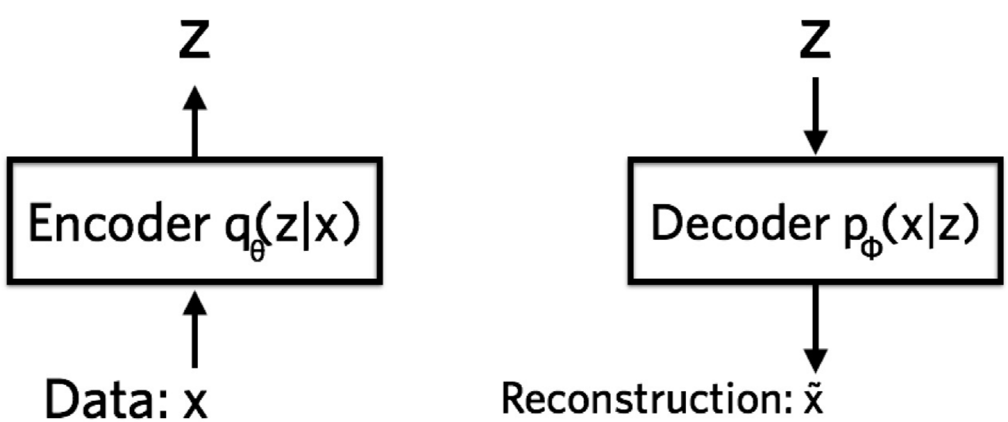
- Encoder and decoder are neural networks with parameter θ and ϕ

Encoder $q_\theta(\boldsymbol{z}|\boldsymbol{x})$

- Input: a data point $\boldsymbol{x}$, output: its low-dim representation $\boldsymbol{z}$

Decoder $p_\phi(\boldsymbol{x}|\boldsymbol{z})$

- Input: a representation $\boldsymbol{z}$, output: a data point $\boldsymbol{x}$



VAE

Loss function = reconstruction loss + regularization

- The reconstruction term is the same with AE
- The regularization term organizes the latent space by making distributions returned by the encoder close to normal distribution
- The loss for a data point is

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)}\left[\log p_\phi(\boldsymbol{x}_i|\boldsymbol{z})\right] + KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)||p(\boldsymbol{z}))$$

- Total loss is a summation of individual loss $\sum_{i=1}^{m} l_i$

Intuition about Regularization

Two main properties for regularity of latent space

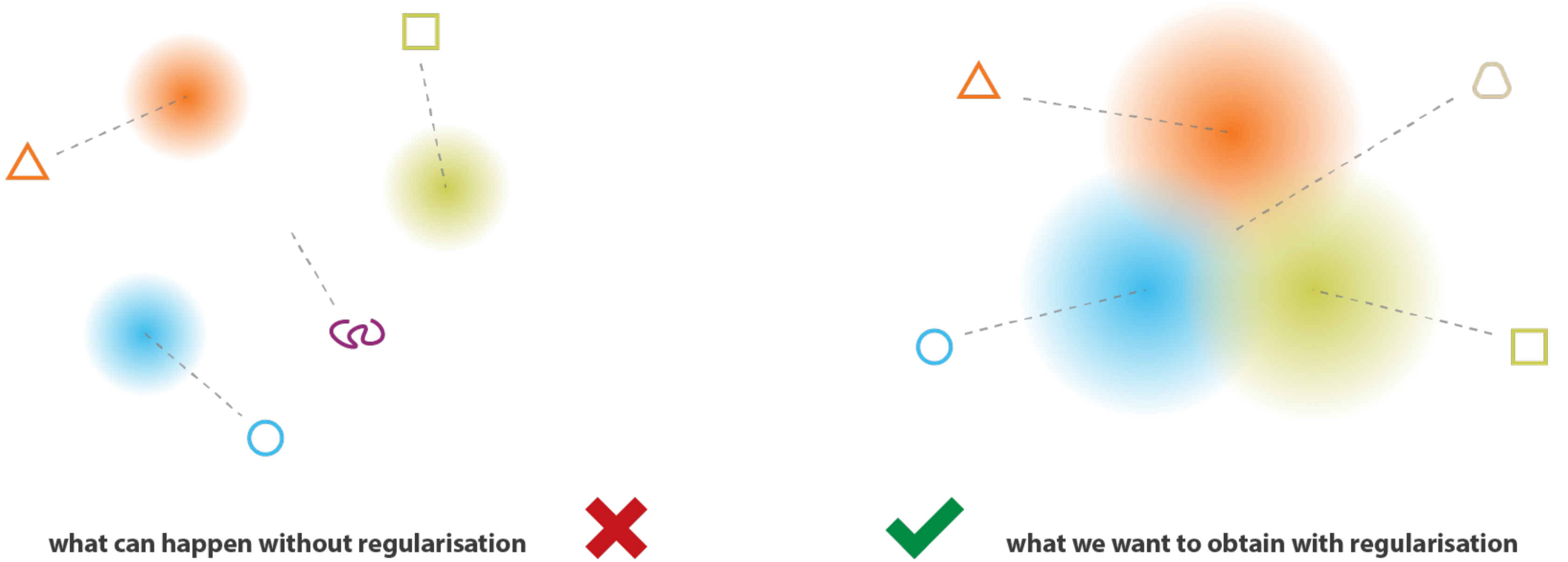
- Continuity: two close points in the latent space should not give two completely different contents once decoded
- Completeness: a point sampled from the latent distribution should give "meaningful" content once decoded

[Source: Joseph Rocca]

Intuition about Regularization

Regularization enforces distributions to be close to a standard normal (centered and reduced)

- The covariance matrices to be close to the identity (not to be skinny or focused on a point)
- The mean to be close to 0 (prevent too far apart from each others)

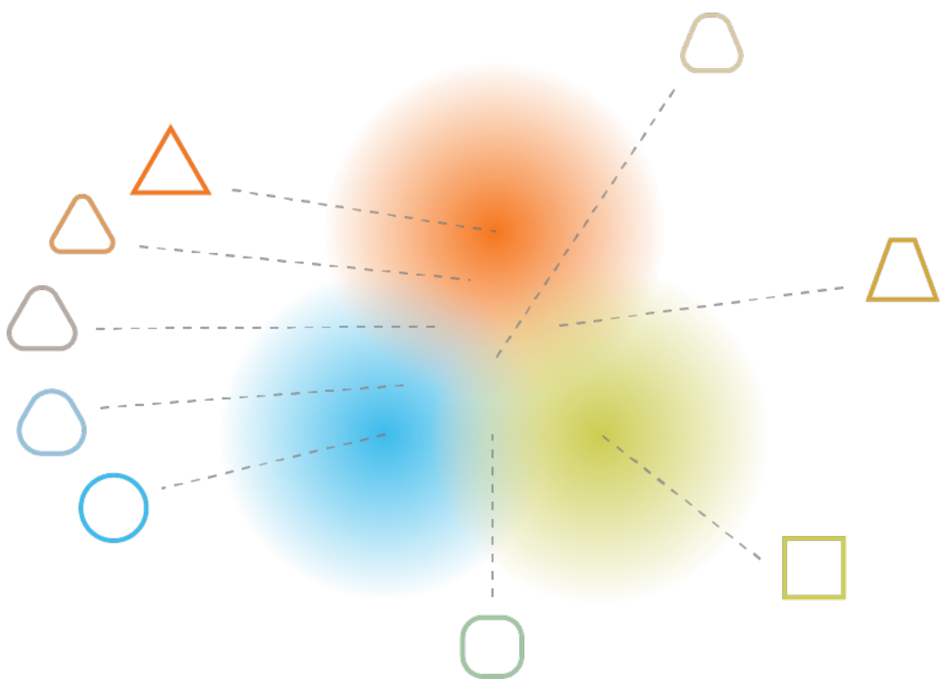


[Source: Joseph Rocca]

Intuition about Regularization

Overlapped distribution is encouraged

- To satisfy the expected continuity and completeness conditions
- However, any regularization including this comes at the price of a higher reconstruction error on the training data



[Source: Joseph Rocca]

VAE

Loss function = reconstruction loss + regularization

- The loss for a data point is

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)}\left[\log p_\phi(\boldsymbol{x}_i|\boldsymbol{z})\right] + KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)||p(\boldsymbol{z}))$$

The 1st term: negative log-likelihood

- The expectation is taken w.r.t the encoder's distribution over the representations
- If the decoder's output does not reconstruct the data well, it will incur a large cost in this loss function

VAE

Loss function = reconstruction loss + regularization

- The loss for a data point is

$$l_i(\theta, \phi) = -\mathbb{E}_{z\sim q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)}\left[\log p_\phi(\boldsymbol{x}_i|\boldsymbol{z})\right] + KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)||p(\boldsymbol{z}))$$

The 2nd term: regularization

- KL divergence between the decoder's distribution $q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)$ and actual $p(\boldsymbol{z})$
- Measure how much information is lost when using q to represent p (i.e., how close q is to p)
- In VAE, $p(\boldsymbol{z}) = \mathrm{Normal}(\boldsymbol{0}, \boldsymbol{1})$
- Make the representation space of $\boldsymbol{z}$ meaningful (i.e., if the encoder output is different from standard normal, it is penalized)

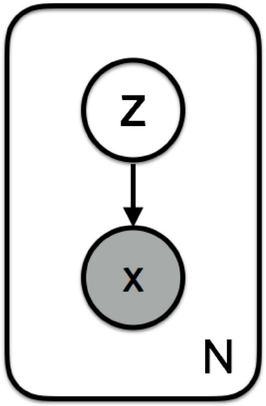
VAE from PGM Perspective

A probabilistic model of data $\boldsymbol{x}$ and latent variable $\boldsymbol{z}$

- Joint probability of the model

$$P(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$$

$$= (\text{likelihood}) \times (\text{prior})$$



- Data generating process

For each data point i

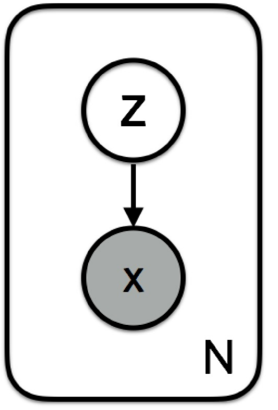
- Draw latent variable $\boldsymbol{z}_i \sim p(\boldsymbol{z})$
- Draw data point $\boldsymbol{x}_i \sim p(\boldsymbol{x}|\boldsymbol{z}_i)$

VAE from PGM Perspective

Goal: calculate posterior $p(\boldsymbol{z}|\boldsymbol{x})$

- Infer good values of the latent variables given observed data

$$P(\boldsymbol{z}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p(\boldsymbol{x})}$$



- Computing denominator $p(\boldsymbol{x})$ is intractable!

$$P(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\mathrm{d}\boldsymbol{z}$$

Solution: use variational inference

- Approximate the posterior $P(\boldsymbol{z}|\boldsymbol{x})$ with $q_\theta(\boldsymbol{z}|\boldsymbol{x})$ of a known distribution with parameter θ
- How to decide whether $q_\theta(\boldsymbol{z}|\boldsymbol{x})$ approximates $P(\boldsymbol{z}|\boldsymbol{x})$ well?

VAE from PGM Perspective

Find θ that minimizes $KL(q_\theta(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x}))$

$$q_\theta^*(\boldsymbol{z}|\boldsymbol{x}) = \underset{\theta}{\operatorname{argmin}}\, KL(q_\theta(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x}))$$

- The KL divergence becomes

$$KL(q_\theta(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x})) = \sum q_\theta(\boldsymbol{z}|\boldsymbol{x}) \log \frac{q_\theta(\boldsymbol{z}|\boldsymbol{x})}{p(\boldsymbol{z}|\boldsymbol{x})} = \mathbb{E}_q[\log \frac{q_\theta(\boldsymbol{z}|\boldsymbol{x})p(\boldsymbol{x})}{p(\boldsymbol{x},\boldsymbol{z})}]$$

$$= \mathbb{E}_q[\log q_\theta(\boldsymbol{z}|\boldsymbol{x})] - \mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z})] + \log p(\boldsymbol{x})$$

- Now re-organize it

$$\log p(\boldsymbol{x}) = \underbrace{\mathbb{E}_q[\log p(\boldsymbol{x},\boldsymbol{z})] - \mathbb{E}_q[\log q_\theta(\boldsymbol{z}|\boldsymbol{x})]}_{\text{ELBO (Evidence Lower BOund)}} + \underbrace{KL(q_\theta(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}|\boldsymbol{x}))}_{\geq 0}$$

$\boxed{\text{Minimizing KL divergence = Maximizing ELBO!}}$

VAE from PGM Perspective

Goal: maximize ELBO (tractable)

- For each datapoint i

$$
\begin{aligned}
\mathrm{ELBO}_i(\lambda) &= \mathbb{E}_q[\log p(\boldsymbol{x}_i, \boldsymbol{z})] - \mathbb{E}_q[\log q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)] \\
&= \mathbb{E}_q[\log p(\boldsymbol{x}_i|\boldsymbol{z})] + \mathbb{E}_q[\log p(\boldsymbol{z})] - \mathbb{E}_q[\log q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)] \\
&= \mathbb{E}_q[\log p(\boldsymbol{x}_i|\boldsymbol{z})] - KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)||p(\boldsymbol{z}))
\end{aligned}
$$

Learn θ, ϕ that maximize ELBO

$$
\mathrm{ELBO}_i(\theta, \phi) = \mathbb{E}_q\left[\log p_\phi(\boldsymbol{x}_i|\boldsymbol{z})\right] - KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)||p(\boldsymbol{z}))
$$

- Remind that $p_\phi(\boldsymbol{x}_i|\boldsymbol{z})$ is the decoder, and $q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)$ is the encoder (both are neural networks)

PGM and AE Perspective

AE perspective: minimize loss

$$l_i(\theta, \phi) = -E_{z \sim q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)}\left[\log p_\phi(\boldsymbol{x}_i|\boldsymbol{z})\right] + KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)||p(\boldsymbol{z}))$$

PGM perspective: maximize ELBO

$$\mathrm{ELBO}_i(\theta, \phi) = \mathbb{E}_q\left[\log p_\phi(\boldsymbol{x}_i|\boldsymbol{z})\right] - KL(q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)||p(\boldsymbol{z}))$$

They are equivalent $\mathrm{ELBO}_i(\theta, \phi) = -l_i(\theta, \phi)$

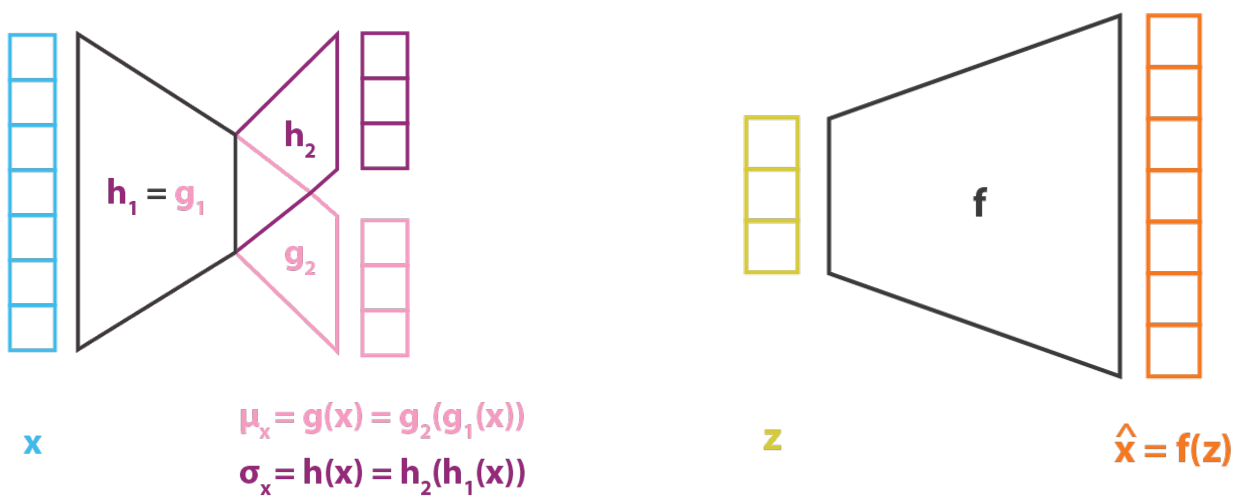
Modeling Encoder and Decoder

Encoder $q_\theta(\boldsymbol{z}|\boldsymbol{x}_i)$

- The encoder is a neural network that takes $\boldsymbol{x}_i$ and outputs the mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\sigma}_i$ of the multivariate Gaussian

Decoder $p_\phi(\boldsymbol{x}_i|\boldsymbol{z})$

- The decoder is another neural network that takes a sampled $\boldsymbol{z}$ from Gaussian and outputs a reconstructed $\hat{\boldsymbol{x}}_i$

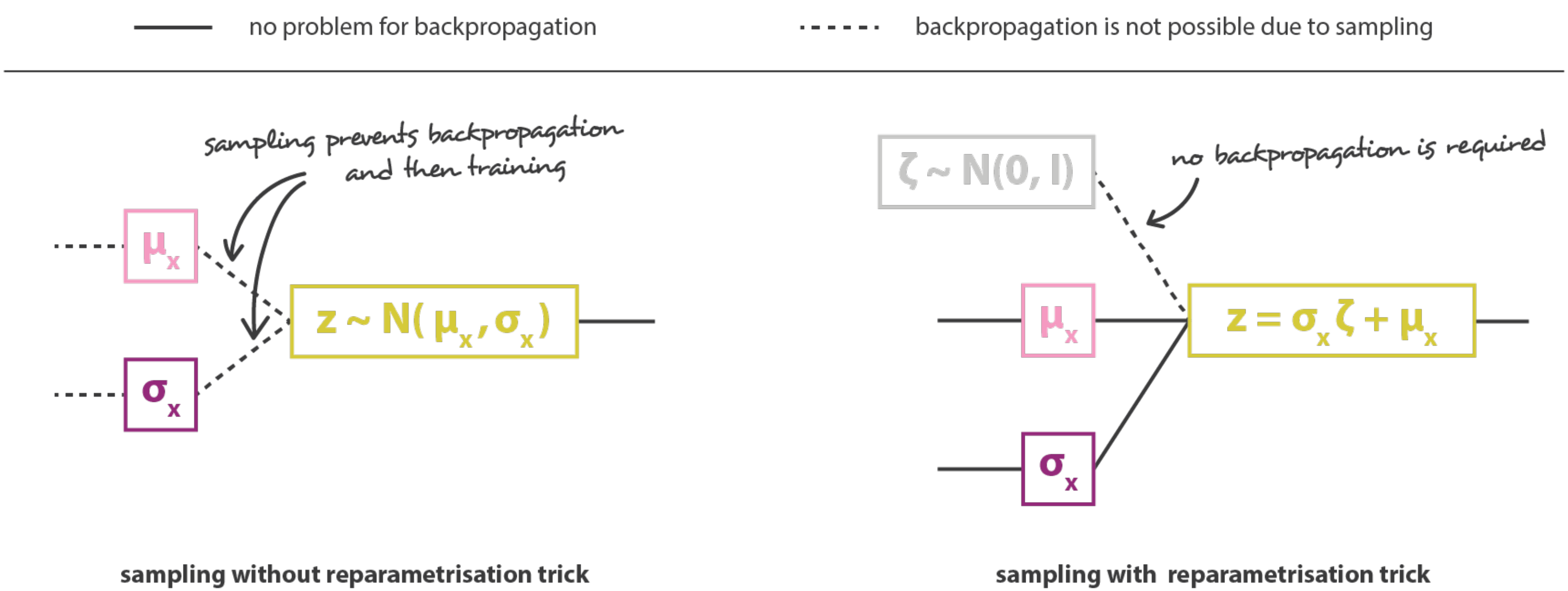


[Source: Joseph Rocca]

Reparametrization Trick

VAE is a concatenation of encoder and decoder

- Sampling is a stochastic process and therefore we cannot backpropagate the gradient
- A simple trick makes the gradient descent possible despite the random sampling for z that occurs halfway of the architecture

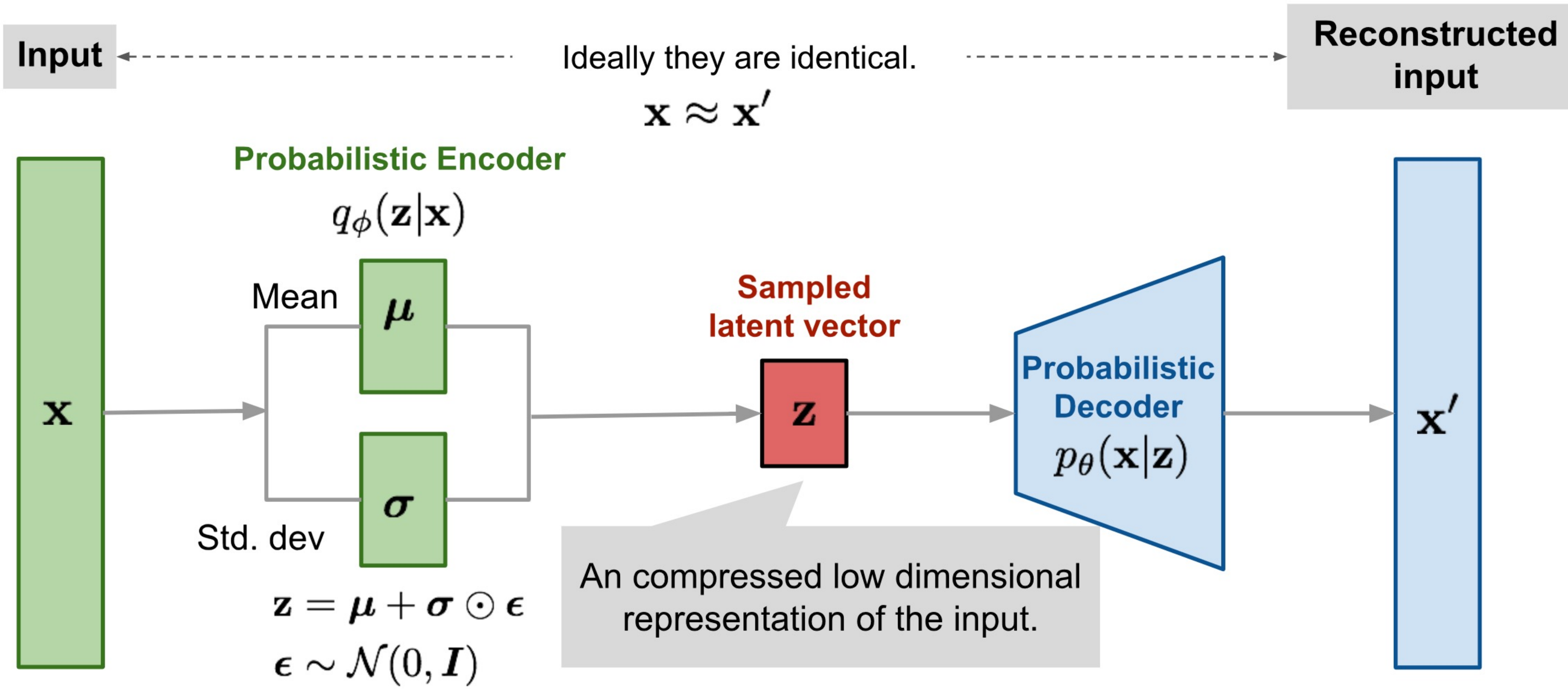


[Source: Joseph Rocca]

The Overall Architecture

End-to-end training is possible

- Use L2 distance between $\boldsymbol{x}_i$ and $\hat{\boldsymbol{x}}_i$ for reconstruction loss



[Source: Lilian Weng]

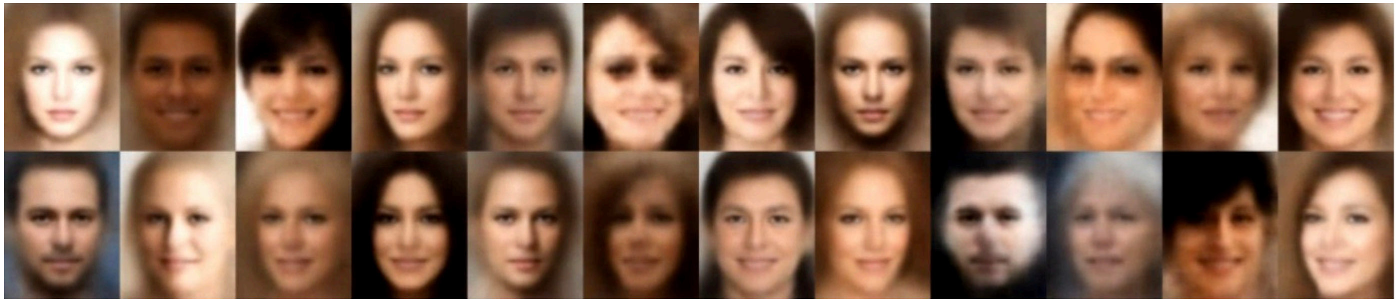
VAE vs. GAN

Downside of GAN

- Images are generated off some arbitrary noise
  - It is not straightforward to generate with specific features
- GAN only discriminates between real and fake images
  - No constraints that an image of a cat must look like a cat
  - No actual object in a generated image, but the style just looks like a cat picture

Downside of VAE

- Output images are blurry; it uses direct mean squared errors



Summary

AE is good for dimensionality reduction

- AE is a neural network composed of an encoder and a decoder
- Create a bottleneck to go through for data
- Trained to lose a minimal quantity of information during the encoding-decoding process (i.e., reduce the reconstruction error)
- Due to overfitting, the latent space can be extremely irregular
- Hard to use for a generative process

VAE tackles the AE’s problem of latent space irregularity

- Encoder returns a distribution over the latent space instead of a single point
- Loss function additionally includes a regularization for better organization of the latent space
- Derived from the technique of variational inference