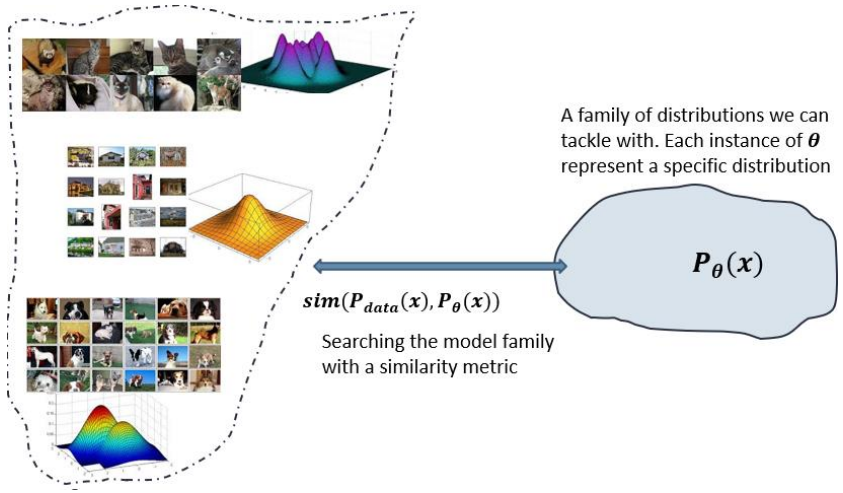
Variational autoencoder

22-808: Generative models
Sharif University of Technology
Fall 2024

Fatemeh Seyyedsalehi

Recap



- We need a framework to interact with distributions for statistical generative models.
  - Probabilistic generative models
  - Deep generative models
    - Autoregressive models
    - **Variational autoencoder**

Latent variable models

Given finite samples $x_1, \ldots, x_n \sim p$, and unlimited samples $z \sim r$.

Generative latent variable model:

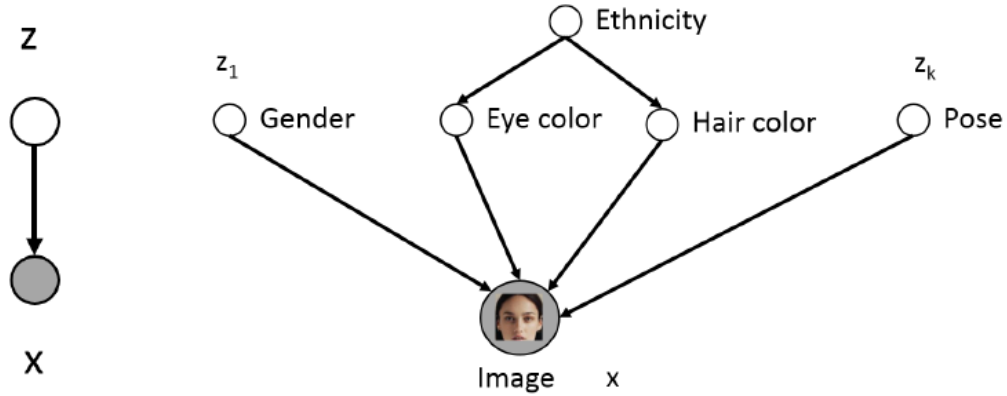
1. $z \sim r$,
2. $x \sim p_\theta(\cdot|z)$

Want to learn the marginal $p_\theta(x) \approx p(x)$ defined by

$$p_\theta(x) = \int_{\mathcal{Z}} p_\theta(x|z) r(z)\, dz.$$

Latent variable models

- Latent variables helps to discover structures behind the data.
  - Useful for downstream tasks or interpretability.
  - Latent variables can capture the variability in the data



Latent variable models

Want to learn the marginal $p_\theta(x) \approx p(x)$ defined by

$$p_\theta(x) = \int_{\mathcal{Z}} p_\theta(x|z) r(z)\, dz.$$

Fit the maximum likelihood estimator?

$$\hat{\theta}_{\text{mle}} \equiv \arg\max_\theta \mathop{\mathbb{E}}_{x \sim p} \log p_\theta(x) = \arg\max_\theta \mathop{\mathbb{E}}_{x \sim p} \log \int_{\mathcal{Z}} p_\theta(x|z) r(z)\, dz$$

This doesn't look promising…

Example of shallow latent variable model: GMM

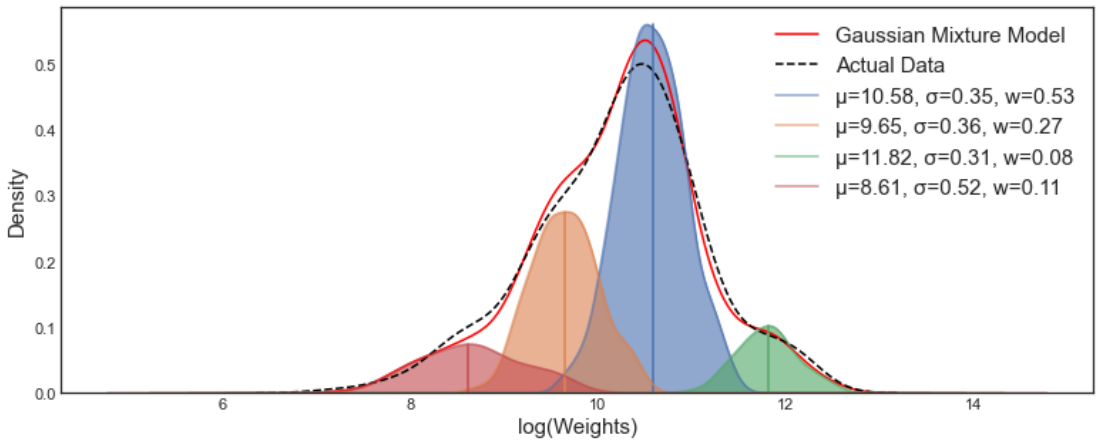
Generative model:

$$1. \quad z \sim \text{Categorical}_{\pi}(K), \qquad \pi \in \Delta^{K-1},$$
$$2. \quad x \sim \mathcal{N}(\mu_z, \Sigma_z), \qquad \mu \in \mathbb{R}^{K \times d}, \Sigma \in \mathbb{R}^{K \times d \times d}.$$

Likelihood:

$$p_\theta(x) = \int_{\mathcal{Z}} p_\theta(x|z) r(z)\, dz = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k).$$

But what if $r(z)$ is a continuous distribution over, e.g. $z \in \mathbb{R}^k$?



The ELBO lower bound

Fit the maximum likelihood estimator?

$$\hat{\theta}_{\text{mle}} \equiv \arg\max_{\theta} \mathop{\mathbb{E}}_{x\sim p} \log p_\theta(x) = \arg\max_{\theta} \mathop{\mathbb{E}}_{x\sim p} \log \int_{\mathcal{Z}} p_\theta(x|z) r(z)\, dz.$$

Use importance sampling to estimate the integral.

Construct a lower-bound on the marginal log-likelihood (the ELBO):

$$\log p_\theta(x) = \log \mathop{\mathbb{E}}_{z\sim q(\cdot|x)} \left[\frac{p_\theta(x,z)}{q(z|x)}\right] \geq \mathop{\mathbb{E}}_{z\sim q(\cdot|x)} \left[\log \frac{p_\theta(x,z)}{q(z|x)}\right].$$

ELBO estimation with Monte Carlo

Importance-sampling estimator: $\log p_\theta(x) = \log \mathop{\mathbb{E}}_{z \sim q(\cdot|x)} \left[\frac{p_\theta(x,z)}{q(z|x)} \right]$.

Cannot directly estimate the log-likelihood with samples.

Let $z_i \sim q(\cdot|x)$. Evidence lower-bound (often use $m = 1$; like "hard" EM):

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{z \sim q(\cdot|x)} \left[\log \frac{p_\theta(x,z)}{q(z|x)} \right] \approx \frac{1}{m} \sum_{i=1}^{m} \left[\log \frac{p_\theta(x,z_i)}{q(z_i|x)} \right].$$

Optimizing ELBO

Define the ELBO to be $\mathcal{L}(x, z; \theta, q) \equiv \log \frac{p_\theta(x, z)}{q(z|x)}$.

Estimate the marginal log-likelihood with by $\log p_\theta(x) \geq \mathop{\mathbb{E}}_{z \sim q(\cdot|x)} \mathcal{L}(x, z; \theta, q)$.

Equality holds when $q(z|x) = p_\theta(z|x) = \frac{p_\theta(x|z) r(z)}{p_\theta(x)}$.

Jointly optimize over θ, q:

$$\hat{\theta}_{\text{mle}} \equiv \arg\max_\theta \mathop{\mathbb{E}}_{x \sim p} \log p_\theta(x) = \arg\max_\theta \sup_q \mathop{\mathbb{E}}_{\substack{x \sim p \\ z \sim q(\cdot|x)}} \mathcal{L}(x, z; \theta, q).$$

Optimizing ELBO

How to estimate the posterior $q(z|x) \approx p_\theta(z|x)$?

For GMM's this was easy. We can compute the posterior exactly:

$$q(z|x) = p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)} = \frac{\pi_z \mathcal{N}(x; \mu_z, \Sigma_z)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \Sigma_k)}.$$

What if the model isn't so simple?

- What if the likelihood $p_\theta(x|z)$ isn't just a Gaussian?
- What if the prior $r(z)$ is a continuous distribution on $z \in \mathbb{R}^k$?

Optimizing ELBO

How to estimate the posterior $q(z|x) \approx p_\theta(z|x)$?

Learn a model that approximates the posterior!

Let $q_\phi(z|x)$ be a family of density estimators with parameters ϕ.

People sometimes call this amortized inference.

Stochastic Backpropagation

Jointly optimize over θ, ϕ:

$$\hat{\theta}_{\text{mle}} \equiv \arg\max_{\theta} \mathop{\mathbb{E}}_{x\sim p} \log p_\theta(x) = \arg\max_{\theta} \sup_{\phi} \mathop{\mathbb{E}}_{\substack{x\sim p \\ z\sim q_\phi(\cdot|x)}} \mathcal{L}(x, z; \theta, \phi).$$

Let's use SGD, given a sample $x_i \sim p, z_i \sim q_\phi(\cdot|x_i)$.

Estimate the gradient w.r.t θ : $\nabla_\theta \mathop{\mathbb{E}}_{\substack{x\sim p \\ z\sim q_\phi(\cdot|x)}} \mathcal{L}(x, z; \theta, \phi) \approx \nabla_\theta \log \frac{p_\theta(x_i, z_i)}{q_\phi(z_i|x_i)}$.

But we're in trouble computing $\nabla_\phi \mathop{\mathbb{E}}_{\substack{x\sim p \\ z\sim q_\phi(\cdot|x)}} \mathcal{L}(x, z; \theta, \phi)$.

The Reparameterization Trick

Need to construct a Monte Carlo estimate of $\nabla_\phi \mathop{\mathbb{E}}_{\substack{x\sim p \\ z\sim q_\phi(\cdot|x)}} \mathcal{L}(x, z; \theta, \phi)$.

Suppose $q_\phi(z|x)$ is defined by a pushforward distribution, e.g.

$$z = f_\phi(x, \varepsilon), \text{ where } \varepsilon \sim \mathcal{N}(0, I).$$

Then $\nabla_\phi \mathop{\mathbb{E}}_{\substack{x\sim p \\ z\sim q_\phi(\cdot|x)}} \mathcal{L}(x, z; \theta, \phi) = \mathop{\mathbb{E}}_{\substack{x\sim p \\ \varepsilon\sim\mathcal{N}(0,I)}} \nabla_\phi \mathcal{L}(x, f_\phi(x, \epsilon); \theta, \phi)$.

This is an example of Monte Carlo gradient estimation.

The Gaussian VAE

Use a prior $r(z) = \mathcal{N}(0, I)$ where $z \in \mathbb{R}^k$ (k is a hyper-parameter).

With a Gaussian likelihood $p_\theta(x|z) = \mathcal{N}(x; g_\theta(z), \sigma^2_\theta(z)I)$.

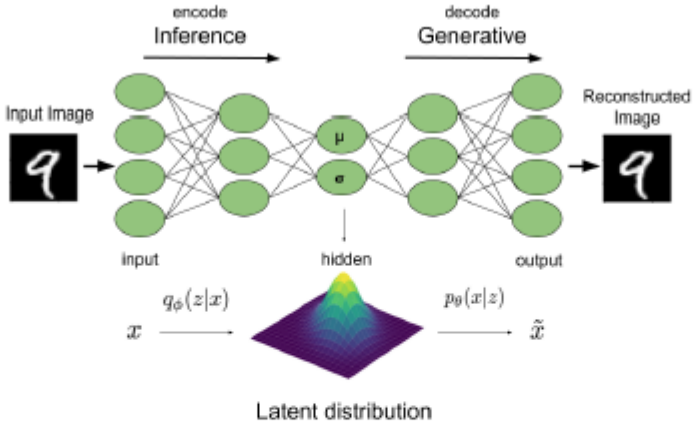
Where $g_\theta : \mathcal{Z} \to \mathcal{X}$ (the "decoder") and $\sigma^2_\theta : \mathcal{Z} \to \mathbb{R}$ are neural nets.

Use a posterior approximation $q_\phi(z|x) = \mathcal{N}(z; f_\phi(x), \Sigma_\phi(x))$

Where $f_\phi : \mathcal{X} \to \mathcal{Z}$ (the "encoder") and $\Sigma_\phi : \mathcal{X} \to \mathcal{Z} \otimes \mathcal{Z}$ are neural nets.

Think of it like a Gaussian mixture model with infinitely many components!

The Gaussian VAE



$$
\begin{aligned}
\mathcal{L}(\mathbf{x};\theta,\phi) &= E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{z},\mathbf{x};\theta) - \log q_\phi(\mathbf{z}|\mathbf{x}))] \\
&= E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{z},\mathbf{x};\theta) - \log p(\mathbf{z}) + \log p(\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x}))] \\
&= E_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z};\theta)] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))
\end{aligned}
$$

Reconstruction and Divergence

The ELBO of the Gaussian VAE is:

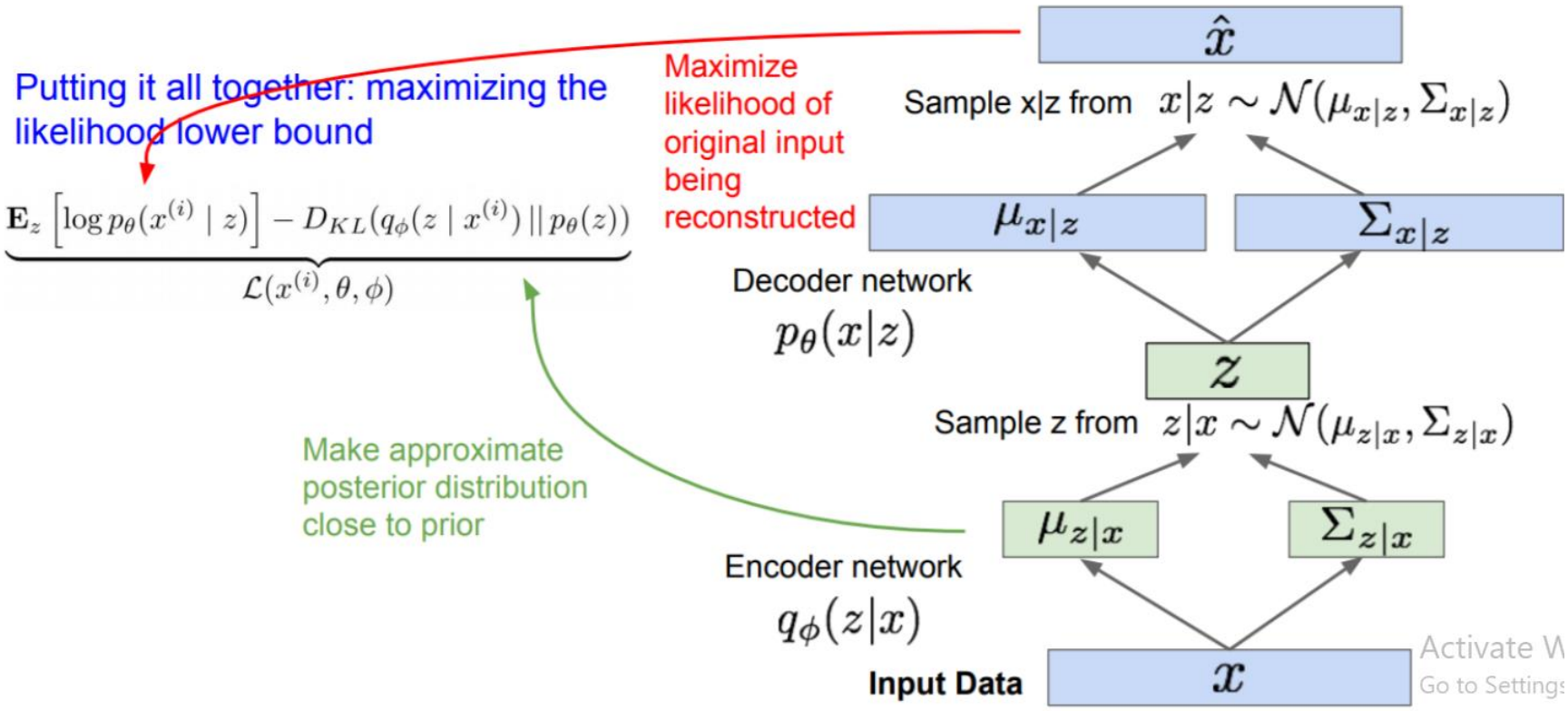
$$-\frac{\dim(\mathcal{X})}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\mathop{\mathbb{E}}_{z\sim q_\phi(\cdot|x)}\|x - g_\theta(z)\|^2 - D(q_\phi(z|x)\parallel r(z)).$$

If σ^2 is held constant, then

$$\hat{\theta}_{\text{mle}} = \arg\min_\theta \inf_\varphi \mathop{\mathbb{E}}_{\substack{x\sim p \\ z\sim q_\varphi(\cdot|x)}}\left[\frac{1}{2\sigma^2}\|x - g_\theta(z)\|^2 + D(q_\varphi(z|x)\parallel r(z))\right].$$

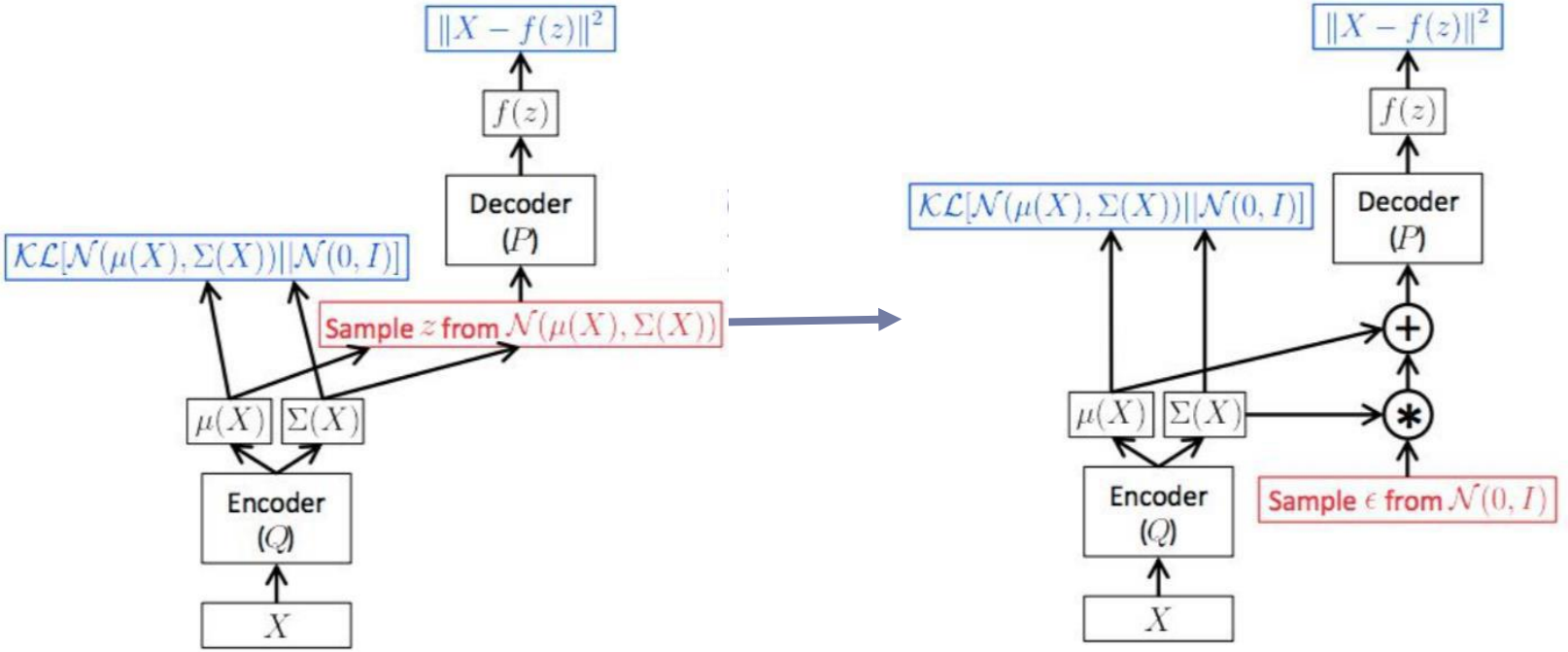
People refer to these to terms as “reconstruction” and “divergence.”

The Gaussian VAE



Reparameterization in VAE

- We can easily backpropagate the loss to the Encoder.



VAE vs. AE (autoencoder)

- The main advantage of a VAE over a deterministic autoencoder is that it defines a proper generative model, that can create sensible-looking novel images by decoding prior samples.

- An autoencoder only knows how to decode latent variables derived from the training set and performs poorly when fed random inputs.

VAE vs. AE (autoencoder)

- In the case of reconstruction task, both models can reconstruct a given input image reasonably well, although the VAE reconstructions are somewhat blurry.

- Row 1: main image
- Row 2: reconstructed with AE
- Row 3,4: reconstructed with VAE

β-VAE

- VAEs often generate somewhat blurry images
- This is not the case for models that optimize the exact likelihood, such as pixelCNNs
- Considering the reconstruction term of the VAE loss function:

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) = -\frac{1}{2\sigma^2}||\boldsymbol{x} - d_{\boldsymbol{\theta}}(\boldsymbol{z})||_2^2 + \text{const}$$

- The VAE encoder maps a set of different inputs to a same latent variable.
- By the above reconstructing penalty, decoder should predict the average of all possible inputs mapped to this latent code.

β-VAE

- We can solve this problem by increasing the expressive power of the posterior approximation (avoiding the merging of distinct inputs into the same latent code), or of the generator (by adding back information that is missing from the latent code), or both.

- However, an even simpler solution is to reduce the penalty on the KL term, making the model closer to a deterministic autoencoder:

$$\mathcal{L}_\beta(\boldsymbol{\theta}, \boldsymbol{\phi}|\boldsymbol{x}) = \underbrace{-\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right]}_{\mathcal{L}_E} + \beta \underbrace{D_{\mathbb{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{z})\right)}_{\mathcal{L}_R}$$

Next topics

- Generative adversarial networks (GAN)