Deep Generative Models

Variational Autoencoder

Hamid Beigy

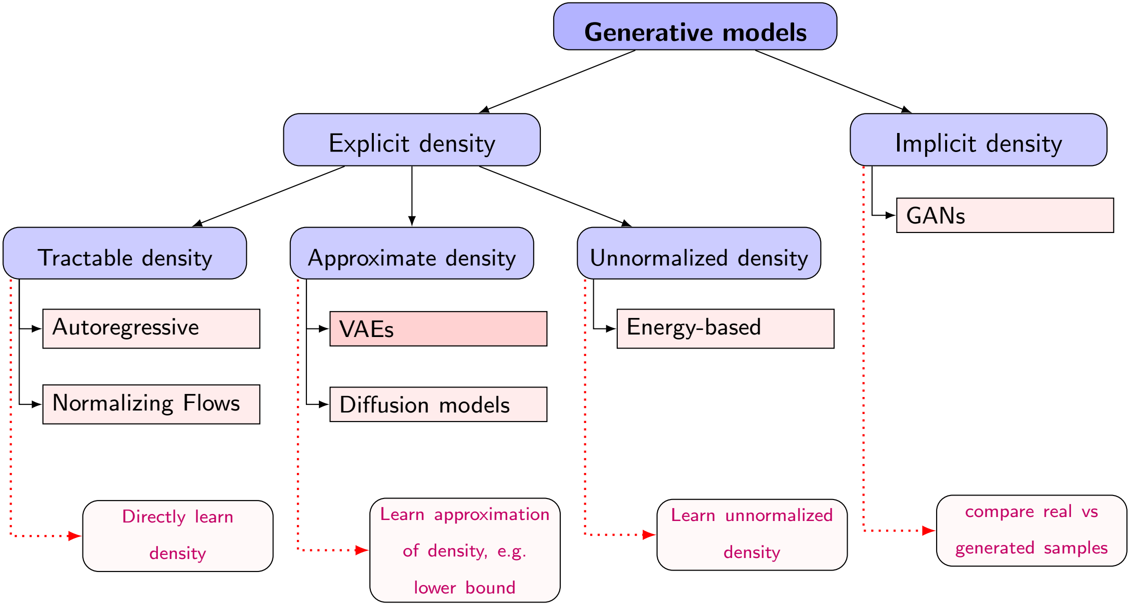
Sharif University of Technology

April 6, 2024

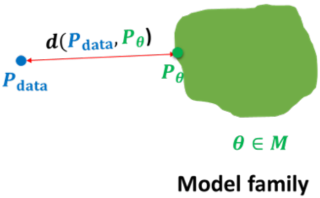
Table of contents

Introduction

Generative models categories



1. Assume that the observed variable $\mathbf{x}$ is a random sample from an underlying process, whose true distribution $p_{data}(\mathbf{x})$ is unknown.



2. We attempt to approximate this process with a chosen model, $p_\theta(\mathbf{x})$, with parameters θ such that $\mathbf{x} \sim p_\theta(\mathbf{x})$.

3. Learning is the process of searching for the parameter θ such that $p_\theta(\mathbf{x})$ well approximates $p_{data}(\mathbf{x})$ for any observed $\mathbf{x}$, i.e.

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

4. We wish $p_\theta(\mathbf{x})$ to be sufficiently flexible to be able to adapt to the data for obtaining sufficiently accurate model and to be able to incorporate prior knowledge.

Credit: Aditya Grover

Maximum Likelihood and Mini-batch SGD

1. The most common criterion for probabilistic models is maximum log-likelihood.
2. Maximization of the log-likelihood criterion is equivalent to minimization of a KL divergence between the data and model distributions.
3. We attempt to find the parameters θ that maximize the sum of the log-probabilities.

$$\theta = \arg\max_{\theta} \log \mathrm{p}_{\theta}(D) = \arg\max_{\theta} \sum_{i=1}^{n} \log \mathrm{p}_{\theta}(\mathbf{x}_i)$$

4. We can efficiently compute gradients of this objective function.
5. We can use such gradients to iteratively hill-climb to a local optimum of the the objective function.
6. If we compute such gradients using all data points, $\nabla_{\theta} \log \mathrm{p}_{\theta}(D)$, then this is known as batch gradient descent.
7. Computation of this derivative is an expensive operation for large dataset.

Maximum Likelihood and Mini-batch SGD

1. A more efficient method for optimization is stochastic gradient descent(SGD).

2. The SGD uses randomly drawn mini-batches of data $B \subseteq D$ of size n_B.

3. With mini-batches, we can form an unbiased estimator of the log-likelihood as

$$\frac{1}{n} \log \mathrm{p}_\theta(D) \approx \frac{1}{n_B} \log \mathrm{p}_\theta(B) = \frac{1}{n_B} \sum_{\mathbf{x} \in B} \log \mathrm{p}_\theta(\mathbf{x})$$

4. Symbol $\approx$ means that one of the two sides is an unbiased estimator of the other side.

5. The unbiased estimator $\frac{1}{n_B} \log \mathrm{p}_\theta(B)$ is differentiable, yielding the unbiased stochastic gradients:

$$\frac{1}{n} \nabla_\theta \log \mathrm{p}_\theta(D) \approx \frac{1}{n_B} \nabla_\theta \log \mathrm{p}_\theta(B) = \frac{1}{n_B} \sum_{\mathbf{x} \in B} \nabla_\theta \log \mathrm{p}_\theta(\mathbf{x})$$

6. These gradients can be plugged into stochastic gradient-based optimizer.

Latent Variable Model

Latent Variable Model

1. In economics, we are often interested in measuring things such as quality of life, moral, happiness, etc.
2. This things cannot be directly measured and are **latent**.
3. The idea is to link these latent variables to observed ones.
4. For example, assume that the **quality of life** can be inferred from some **linear combination** of some observed variables such as
   - wealth
   - employment
   - physical health
   - education
   - leisure time
5. Latent variables are part of model, but we cannot observe.
6. Latent variables are another way to represent the data.

Principal component analysis (PCA)

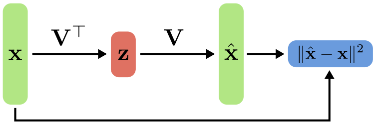
1. Let $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^\top \in \mathbb{R}^{n \times d}$ be a dataset of samples $\mathbf{x}_i \in \mathbb{R}^d$, and $\mathbf{Z} = (\mathbf{z}_1, \dots, \mathbf{z}_n)^\top \in \mathbb{R}^{n \times K}$ be the corresponding latent variables $\mathbf{z}_i \in \mathbb{R}^K$.

2. The goal of PCA is to learn a linear bidirectional mapping $\mathcal{X} \longleftrightarrow \mathcal{Z}$ such that as much information of $\mathcal{X}$ as possible is retained in $\mathcal{Z}$.

3. Let the following linear mapping maps data from latent to observation space.

$$\hat{\mathbf{x}}_i = \bar{\mathbf{x}} + \sum_{j=1}^{K} z_{ij} \mathbf{v}_j$$

where $\bar{\mathbf{x}}$ is data mean and $\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_K)$ is an orthonormal basis.

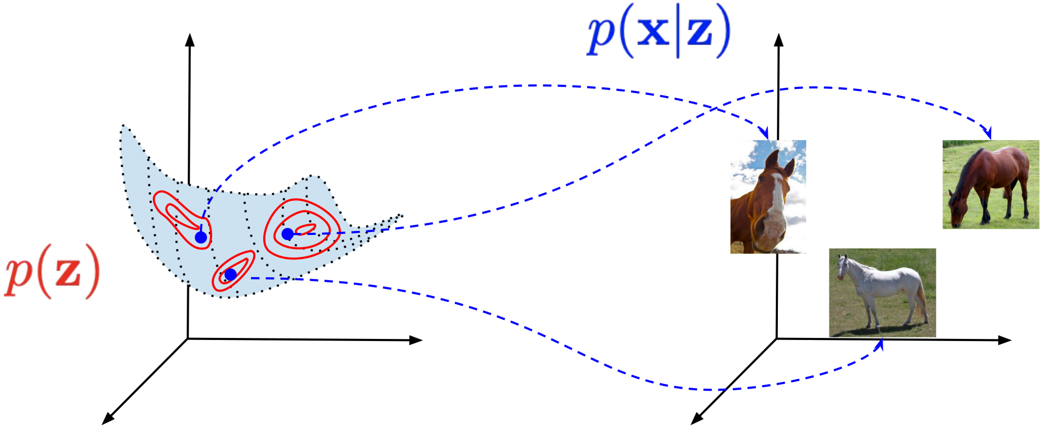


4. The goal is to minimize the L_2 reconstruction loss wrt. $\mathbf{Z}$ and $\mathbf{V}$.

$$\mathcal{L}(\mathbf{Z}, \mathbf{V}) = \sum_{i=1}^{n} \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|^2$$

1. Many probabilistic models have latent variables $\mathbf{z}$.

2. In the case of unconditional modeling of observed variable $\mathbf{x}$, the directed graphical model would then represent a joint distribution $p_\theta(\mathbf{x}, \mathbf{z})$.



Latent Variable Model

1. The marginal distribution over the observed variables, $p_\theta(\mathbf{x})$, is

$$\begin{aligned} p_\theta(\mathbf{x}) &= \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z} \\ &= \int p_\theta(\mathbf{x} \mid \mathbf{z})\, p_\theta(\mathbf{z}) d\mathbf{z} \\ &= \mathbb{E}_{\mathbf{z}}[\, p_\theta(\mathbf{x} \mid \mathbf{z})] \end{aligned}$$

2. This is called the **marginal likelihood** or **model evidence** when taken as a function of θ.

3. Assume that we cannot calculate the integral exactly. The simplest approach would be to use the Monte Carlo approximation:

$$\begin{aligned} p_\theta(\mathbf{x}) &= \int p_\theta(\mathbf{x} \mid \mathbf{z})\, p_\theta(\mathbf{z}) d\mathbf{z} \\ &= \mathbb{E}_{\mathbf{z} \sim p_\theta(\mathbf{z})}[\, p_\theta(\mathbf{x} \mid \mathbf{z})] \\ &= \frac{1}{n_B} \sum_k p_\theta(\mathbf{x} \mid \mathbf{z}_k) \end{aligned}$$

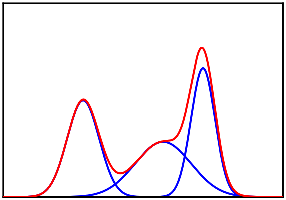
4. In the last line, we use samples from the prior over latents $\mathbf{z} \sim p_\theta(\mathbf{z})$.

1. The marginal distribution over the observed variables, $p_\theta(\mathbf{x})$, is

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}$$

2. Such an implicit distribution over $\mathbf{x}$ can be quite flexible.

   - If $\mathbf{z}$ is discrete and $p_\theta(\mathbf{x} \mid \mathbf{z})$ is a Gaussian distribution, then $p_\theta(\mathbf{x})$ is a **mixture of Gaussian**.
   - If $\mathbf{z}$ is continuous, then $p_\theta(\mathbf{x})$ can be seen as an **infinite mixture**, which are potentially more powerful than discrete mixture.



Linear Gaussian Latent Variable Model

1. Let us consider the following situation
   - We consider continuous random variables only, i.e., $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{z} \in \mathbb{R}^K$.
   - The distrubution of $\mathbf{z}$ is the standard Gaussian, i.e., $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$.
   - The dependency between $\mathbf{z}$ and $\mathbf{z}$ is linear and we assume a Gaussian additive noise:

     $$\mathbf{x} = \mathbf{W}\mathbf{z} + \mathbf{b} + \varepsilon$$

     where $\varepsilon \sim \mathcal{N}(0, \sigma^2\mathbf{I})$

   The property of the Gaussian distribution yields:

   $$p(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}\left(\mathbf{x} \mid \mathbf{W}\mathbf{z} + \mathbf{b}, \sigma^2\mathbf{I}\right)$$

2. This model is known as the **probabilistic PCA**.

1. Then, we can take advantage of properties of a linear combination of two vectors of normally-distributed random variables as

$$\begin{aligned} p(\mathbf{x}) &= \int p(\mathbf{x} \mid \mathbf{z})\, p(\mathbf{z}) d\mathbf{z} \\ &= \int \mathcal{N}(\mathbf{x} \mid \mathbf{W}\mathbf{z} + \mathbf{b}, \sigma \mathbf{I}) \mathcal{N}(\mathbf{z} \mid 0, \mathbf{I}) d\mathbf{z} \\ &= \mathcal{N}\left(\mathbf{x} \mid \mathbf{b}, \mathbf{W}\mathbf{W}^\top + \sigma^2 \mathbf{I}\right). \end{aligned}$$

2. We can calculate the logarithm of the (marginal) likelihood function $\ln p(\mathbf{x})$.
3. We can also calculate the true posterior over $\mathbf{z}$.
4. We can also calculate the true posterior over $\mathbf{z}$.

$$p(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}\left(\mathbf{M}^{-1}\mathbf{W}^\top(\mathbf{x} - \mu), \sigma^{-2}\mathbf{M}\right)$$

where $M = \mathbf{W}^\top \mathbf{W} + \sigma^2 \mathbf{I}$.

1. We use term deep latent variable model (DLVM) to denote a latent variable model, $p_\theta(\mathbf{x}, \mathbf{z})$, whose distribution are parameterized by neural networks.
2. When the prior is Gaussian, the model is called deep latent gaussian model (DLGM).
3. One advantage of DLVM is that even when each factor in the directed model is relatively simple, the marginal distribution, $p_\theta(\mathbf{x})$, can be very complex.
4. This expressiveness makes DLVMs attractive for approximating complicated underlying distribution $p_{data}(\mathbf{x})$.
5. Perhaps the simplest and the most common DLVM is specified as factorization with the following structure:

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x})\, p_\theta(\mathbf{z})$$

where $p_\theta(\mathbf{z})$ and $p_\theta(\mathbf{x} \mid \mathbf{z})$ are specified.

Maximum Likelihood Learning

1. The central object in generative models is to approximate the true underlying distribution of the data $p_{data}(\mathbf{x})$ with the distribution $p_\theta(\mathbf{x}, \mathbf{z})$.
2. Assume that we have a dataset $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ that are i.i.d. and fully-observed.
3. We maximise the probability of observing the data with respect to the parameters θ.
4. The maximum likelihood fit is
$$\theta = \arg\max_\theta \frac{1}{n} \sum_{i=1}^{n} \log\, p_\theta(\mathbf{x}_i)$$
5. In latent variable model, we have
$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x} \mid \mathbf{z})\, p_\theta(\mathbf{z}) d\mathbf{z}$$
6. The maximum likelihood fit is
$$\theta = \arg\max_\theta \frac{1}{n} \sum_{i=1}^{n} \log\left(\int p_\theta(\mathbf{x} \mid \mathbf{z})\, p_\theta(\mathbf{z}) d\mathbf{z} \right)$$
7. The main difficulty of maximum likelihood learning in DLVMs is that the marginal probability of data under the model is typically **intractable**.

Intractability

1. The intractability is due to the following have not an analytic solution or efficient estimator.

$$
\begin{aligned}
p_\theta(\mathbf{x}) &= \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z} \\
&= \int p_\theta(\mathbf{x} \mid \mathbf{z})\, p_\theta(\mathbf{z}) d\mathbf{z}
\end{aligned}
$$

2. Hence, we cannot differentiate it w.r.t θ and optimize it as in fully observable models.
3. The intractability of $p_\theta(\mathbf{x})$ is related to the intractability of posterior $p_\theta(\mathbf{x} \mid \mathbf{z})$.
4. Since $p_\theta(\mathbf{x}, \mathbf{z})$ is efficient to compute, if $p_\theta(\mathbf{x})$ is tractable then $p_\theta(\mathbf{x} \mid \mathbf{z})$ is tractable and vice versa.

$$
p_\theta(\mathbf{x} \mid \mathbf{z}) = \frac{p_\theta(\mathbf{x}, \mathbf{z})}{p_\theta(\mathbf{x})}
$$

5. Approximate inference techniques allows to approximate $p_\theta(\mathbf{x} \mid \mathbf{z})$ and $p_\theta(\mathbf{x})$ in DLVMs.

Estimating the log-likelihood

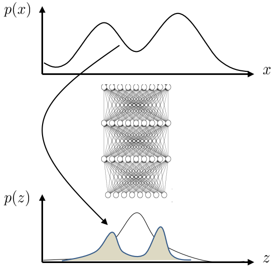
1. An alternative method is to estimate expected log-likelihood.

$$\theta = \arg\max_\theta \frac{1}{n}\sum_{i=1}^{n} \log \mathbb{E}_{\mathbf{z}\sim \mathrm{p}(\mathbf{z}\,|\,\mathbf{x}_i)}[\,\mathrm{p}_\theta(\mathbf{x}_i, \mathbf{z})]$$

2. How to calculate $\mathrm{p}_\theta(\mathbf{z} \mid \mathbf{x}_i)$?

3. Guess most likely $\mathbf{z}$ given $\mathbf{x}_i$ and pretend it is the right one.

4. But, there are many possible values of $\mathbf{z}$, so use the distribution $\mathrm{p}(\mathbf{z} \mid \mathbf{x})$.



Credit: Sergey Levine

Estimating the log-likelihood

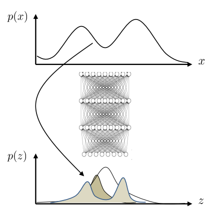
1. How to calculate $p_\theta(\mathbf{z} \mid \mathbf{x}_i)$?

2. Can bound $\log p(\mathbf{x}_i)$?

$$
\begin{aligned}
\log p(\mathbf{x}_i) &= \log \int_{\mathbf{z}} p(\mathbf{x}_i \mid \mathbf{z})\, p(\mathbf{z}) d\mathbf{z} \\
&= \log \int_{\mathbf{z}} p(\mathbf{x}_i \mid \mathbf{z})\, p(\mathbf{z}) \frac{q_i(\mathbf{z})}{q_i(\mathbf{z})} d\mathbf{z} \\
&= \log \mathbb{E}_{\mathbf{z} \sim q_i(\mathbf{z})} \left[\frac{p(\mathbf{x}_i \mid \mathbf{z})\, p(\mathbf{z})}{q_i(\mathbf{z})} \right] \\
&\geq \mathbb{E}_{\mathbf{z} \sim q_i(\mathbf{z})} \left[\log \frac{p(\mathbf{x}_i \mid \mathbf{z})\, p(\mathbf{z})}{q_i(\mathbf{z})} \right] \\
&= \mathbb{E}_{\mathbf{z} \sim q_i(\mathbf{z})} [\log p(\mathbf{x}_i \mid \mathbf{z}) + \log p(\mathbf{z})] + H(q_i)
\end{aligned}
$$

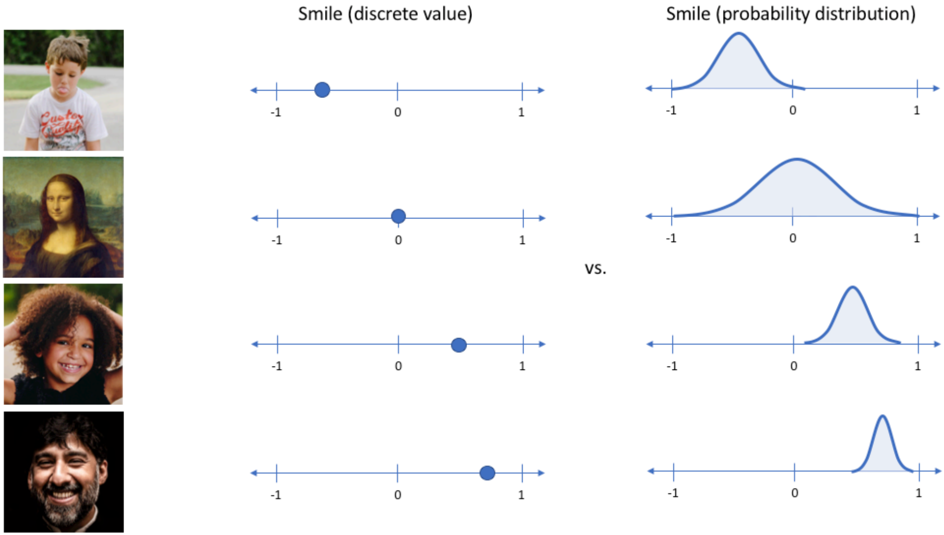
Jensen's ineq. $\log \mathbb{E}[y] \geq \mathbb{E}[\log y]$

3. Let to approximate with $q_i(\mathbf{z}) = \mathcal{N}(\mu_i, \sigma_i)$.



Credit: Sergey Levine

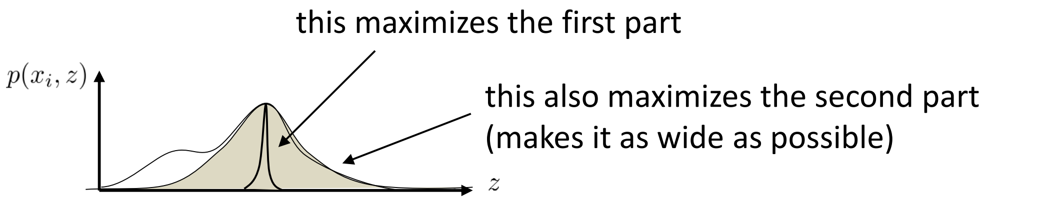
Estimating the log-likelihood



1. Can bound $\log p(\mathbf{x}_i)$?

$$\log p(\mathbf{x}_i) \geq \mathbb{E}_{\mathbf{z}\sim q_i(\mathbf{z})}\left[\log \frac{p(\mathbf{x}_i \mid \mathbf{z})\, p(\mathbf{z})}{q_i(\mathbf{z})}\right] = \underbrace{\mathbb{E}_{\mathbf{z}\sim q_i(\mathbf{z})}[\log p(\mathbf{x}_i \mid \mathbf{z}) + \log p(\mathbf{z})]}_{\text{Maximizing this maximizes } \log p(\mathbf{x}_i)} + H(q_i)$$

2. Whet do we except this to do? $\mathbb{E}_{\mathbf{z}\sim q_i(\mathbf{z})}[\log p(\mathbf{x}_i \mid \mathbf{z}) + \log p(\mathbf{z}) + \log p(\mathbf{x}_i)] + H(q_i)$



Credit: Sergey Levine

1. We bound $\log \mathrm{p}(\mathbf{x}_i)$ as

$$\log \mathrm{p}(\mathbf{x}_i) \geq \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}\left[\log \frac{\mathrm{p}(\mathbf{x}_i \mid \mathbf{z})\, \mathrm{p}(\mathbf{z})}{\mathrm{q}_i(\mathbf{z})}\right] = \underbrace{\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}[\log \mathrm{p}(\mathbf{x}_i \mid \mathbf{z}) + \log \mathrm{p}(\mathbf{z})] + H(\mathrm{q}_i)}_{\mathcal{L}_i(p,q_i)}$$

2. What makes a good $\mathrm{q}_i(\mathbf{z})$ and approximate in what sense?

$$\begin{aligned}
\mathrm{D}_{KL}(\mathrm{q}_i(\mathbf{z}) \,||\, \mathrm{p}(\mathbf{z} \mid \mathbf{x}_i)) &= \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}\left[\log \frac{\mathrm{q}_i(\mathbf{z})}{\mathrm{p}(\mathbf{z} \mid \mathbf{x}_i)}\right] = \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}\left[\log \frac{\mathrm{q}_i(\mathbf{z})\, \mathrm{p}(\mathbf{x}_i)}{\mathrm{p}(\mathbf{x}_i, \mathbf{z})}\right] \\
&= -\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}[\log \mathrm{p}(\mathbf{x}_i \mid \mathbf{z}) + \log \mathrm{p}(\mathbf{z})] \\
&\quad + \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}[\log \mathrm{q}_i(\mathbf{z})] + \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}[\log \mathrm{p}(\mathbf{x}_i)] \\
&= -\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}[\log \mathrm{p}(\mathbf{x}_i \mid \mathbf{z}) + \log \mathrm{p}(\mathbf{z})] - H(\mathrm{q}_i) + \log \mathrm{p}(\mathbf{x}_i) \\
&= -\mathcal{L}(p, q_i) + \log \mathrm{p}(\mathbf{x}_i)
\end{aligned}$$

3. Thus, we have

$$\log \mathrm{p}(\mathbf{x}_i) = \mathrm{D}_{KL}(\mathrm{q}_i(\mathbf{z}) \,||\, \mathrm{p}(\mathbf{z} \mid \mathbf{x}_i)) + \mathcal{L}(p, q_i)$$

4. Hence, we obtain $\log \mathrm{p}(\mathbf{x}_i) \geq \mathcal{L}(p, q_i)$.

Estimating the log-likelihood

1. We bound $\log \mathrm{p}(\mathbf{x}_i)$ as

$$\log \mathrm{p}(\mathbf{x}_i) \geq \underbrace{\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}[\log \mathrm{p}(\mathbf{x}_i \mid \mathbf{z}) + \log \mathrm{p}(\mathbf{z})] + H(\mathrm{q}_i)}_{\mathcal{L}_i(p,q_i)}$$

2. We also found $\mathrm{D}_{KL}(\mathrm{q}_i(\mathbf{z}) \mid\mid \mathrm{p}(\mathbf{z} \mid \mathbf{x}_i))$ as

$$\mathrm{D}_{KL}(\mathrm{q}_i(\mathbf{z}) \mid\mid \mathrm{p}(\mathbf{z} \mid \mathbf{x}_i)) = \underbrace{-\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}[\log \mathrm{p}(\mathbf{x}_i \mid \mathbf{z}) + \log \mathrm{p}(\mathbf{z})] - H(\mathrm{q}_i)}_{\mathcal{L}_i(p,q_i)} + \underbrace{\log \mathrm{p}(\mathbf{x}_i)}_{\text{Independent of } \mathrm{q}_i}$$

3. Hence maximizing $\mathcal{L}_i(p, q_i)$ with respect to q_i minimizes KL-divergence.
4. Then, we define the optimizing function as

$$\theta = \arg\max_\theta \frac{1}{n} \sum_i \mathcal{L}_i(p, q_i)$$

1. We bound $\log \mathrm{p}(\mathbf{x}_i)$ as

$$\log \mathrm{p}(\mathbf{x}_i) \geq \underbrace{\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_i(\mathbf{z})}[\log \mathrm{p}(\mathbf{x}_i \mid \mathbf{z}) + \log \mathrm{p}(\mathbf{z})] + H(\mathrm{q}_i)}_{\mathcal{L}_i(p,q_i)}$$

2. Then, we define the optimizing function as

$$\theta = \arg\max_{\theta} \frac{1}{n} \sum_i \mathcal{L}_i(p, q_i)$$

3. We use the following gradient-based algorithm to find the parameters θ.

for each $\mathbf{x}_i \in B$ **do**
  Calculate $\nabla_\theta \mathcal{L}_i(p, q_i)$.
  Sample $z \sim \mathrm{q}_i(\mathbf{z})$. ▷ $\nabla_\theta \mathcal{L}_i(p, q_i) \approx \log \mathrm{p}_\theta(\mathbf{x}_i \mid \mathbf{z})$
  $\theta \leftarrow \theta + \nabla_\theta \mathcal{L}_i(p, q_i)$ ▷ Update q_i to maximize $\mathcal{L}_i(p, q_i)$.
end for

Let $\mathrm{q}_i(\mathbf{z}) = \mathcal{N}(\mu_i, \sigma_i)$, then use $\nabla_{\mu_i} \mathcal{L}_i(p, q_i)$ and $\nabla_{\sigma_i} \mathcal{L}_i(p, q_i)$ to update μ_i and σ_i.

Amortized Variational Inference

1. We bound $\log p(\mathbf{x}_i)$ as

$$\log p(\mathbf{x}_i) \geq \underbrace{\mathbb{E}_{\mathbf{z}\sim q_i(\mathbf{z})}[\log p(\mathbf{x}_i \mid \mathbf{z}) + \log p(\mathbf{z})] + H(q_i)}_{\mathcal{L}_i(p, q_i)}$$

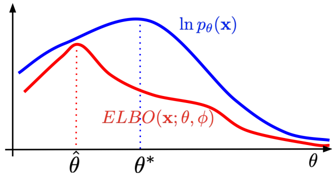
2. We want to approximate $p(\mathbf{z})$ by $q_i(\mathbf{z})$ for all $\mathbf{x}_i$, or equivalently we want to approximate $p_\theta(\mathbf{z} \mid \mathbf{x})$ by $q_\phi(\mathbf{z} \mid \mathbf{x})$ for all $\mathbf{x}$.

3. Hence, it is simple to show that variational lower bound equals to

$$\mathcal{L}(p, q) = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\log \frac{p_\theta(\mathbf{z}, \mathbf{x})}{q_\phi(\mathbf{z} \mid \mathbf{x})}\right]$$

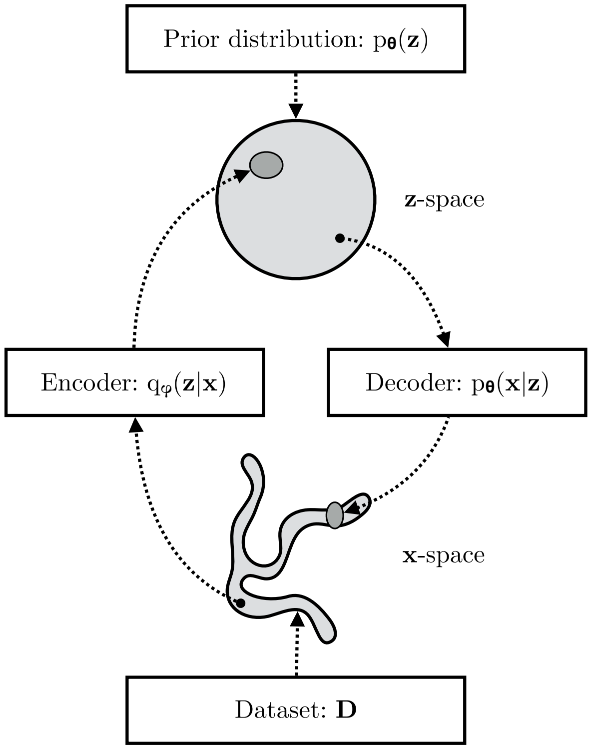
$$\mathcal{L}(p, q) = \log p_\theta(\mathbf{x}) - D_{KL}\left(p_\theta(\mathbf{z} \mid \mathbf{x}) \,||\, q_\phi(\mathbf{z} \mid \mathbf{x})\right) \leq \log p_\theta(\mathbf{x})$$

4. The KL divergence is known as the variational gap.

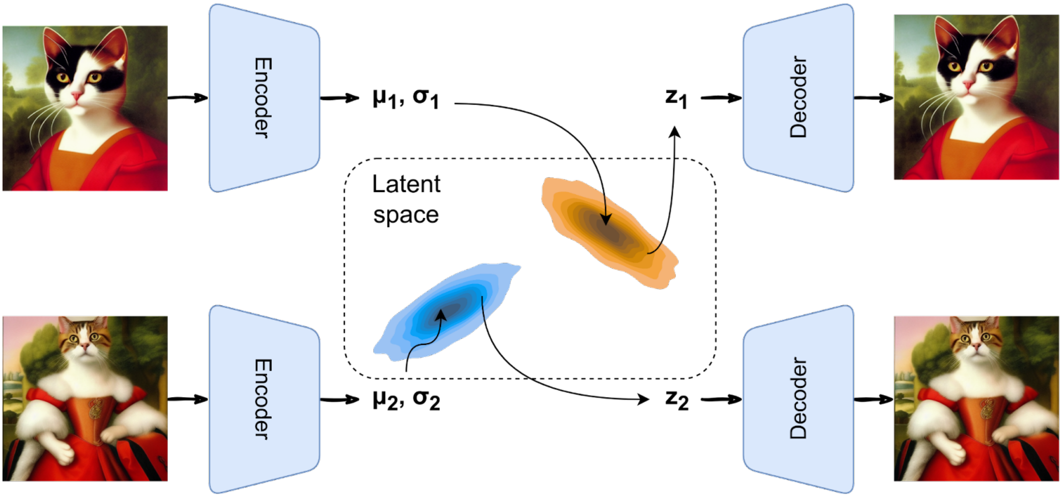


1. The problem is how to estimate $\log q(\mathbf{z} \mid \mathbf{x})$ in which the posterior distribution is different for each data point $\mathbf{x}$. This means that we need to learn different variational parameters ϕ for each data point $\mathbf{x}$.
2. To overcome this issue, we use amortized variational inference.
3. In amortized variational inference, we train an external neural network to predict the variational parameters instead of optimizing ELBO per data point.
4. This network is called the inference network and from now on, ϕ parameters will refer to the inference network weights.
5. The main model (decoder network) and the inference network are trained simultaneously by maximizing ELBO with respect to both θ and ϕ.
6. Once we train the inference network, we can compute the variational posterior for a new data point by simply feeding the data point to the network.

Amortized Variational Inference



Amortized Variational Inference



Credit: Synthesis AI

Computing the gradient of ELBO

1. We need to maximize ELBO with respect to both the model and variational parameters. This means that we need to compute the gradients of:

$$\mathcal{L}(p,q) = \log \mathrm{p}_\theta(\mathbf{x}) - \mathrm{D}_{KL}\big(\, \mathrm{p}_\theta(\mathbf{z} \mid \mathbf{x}) \,||\, \mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x})\big) \leq \log \mathrm{p}_\theta(\mathbf{x})$$

2. Unbiased gradients of the ELBO with respect to θ are:

$$\begin{aligned}
\nabla_\theta \mathcal{L}(p,q)(\mathbf{x}) &= \nabla_\theta\, \mathbb{E}_{\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x})}\big[\log \mathrm{p}_\theta(\mathbf{x},\mathbf{z}) - \log \mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x})\big] \\
&= \mathbb{E}_{\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x})}\big[\nabla_\theta\big(\log \mathrm{p}_\theta(\mathbf{x},\mathbf{z}) - \log \mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x})\big)\big] \\
&\approx \nabla_\theta\big(\log \mathrm{p}_\theta(\mathbf{x},\mathbf{z}) - \log \mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x})\big) \\
&= \nabla_\theta \log \mathrm{p}_\theta(\mathbf{x},\mathbf{z})
\end{aligned}$$

3. Although exact gradient calculation with respect to the model parameters is possible, a much better approach is to use Monte Carlo sampling.

4. We generate a handful of samples for the variational posterior and average them. That way we estimate the gradients instead of calculating them in a closed form.

$$\nabla_\theta \mathcal{L}(p,q)(\mathbf{x}) = \frac{1}{K}\sum_{k=1}^{K} \nabla_\theta \log \mathrm{p}_\theta(\mathbf{x},\mathbf{z}^k) \qquad \text{where} \quad \mathbf{z}^k \sim \mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x})$$

5. Then, use back-propagation algorithm to update the model parameters.

1. Unbiased gradients with respect to the variational parameters ϕ are more difficult to obtain, since the ELBO's expectation is taken with respect to the distribution $q_\phi(\mathbf{z} \mid \mathbf{x})$, which is a function ϕ:

$$\begin{aligned}\nabla_\phi \mathcal{L}(p,q)(\mathbf{x}) &= \nabla_\theta \, \mathbb{E}_{q_\phi(\mathbf{z}\mid\mathbf{x})}\big[\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}\mid\mathbf{x})\big] \\ &\neq \mathbb{E}_{q_\phi(\mathbf{z}\mid\mathbf{x})}\big[\nabla_\theta\big(\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}\mid\mathbf{x})\big)\big]\end{aligned}$$

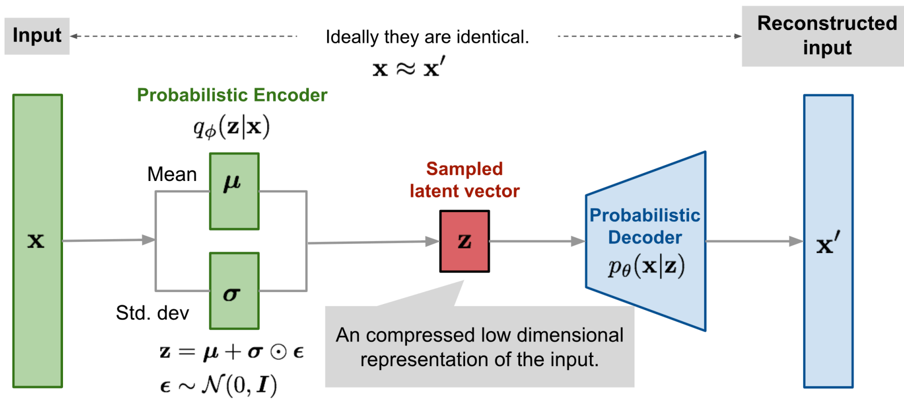
2. Why cannot we to obtain the above gradient?

3. If we can calculate such a gradient, we can use back-propagation algorithm to update the variational parameters.

1. The is transforming a sample from a fixed, known distribution to a sample from $q_\phi(\mathbf{z} \mid \mathbf{x})$.
2. If we consider the Gaussian distribution, we can express $\mathbf{z}$ with respect to a fixed ϵ, where $\epsilon \sim \mathcal{N}(0, I)$.

$$\mathbf{z} = \mu + \sigma\epsilon \qquad \textit{where } \epsilon \sim \mathcal{N}(0, I)$$

3. The ϵ term introduces the stochastic part and it is not involved in the training process.

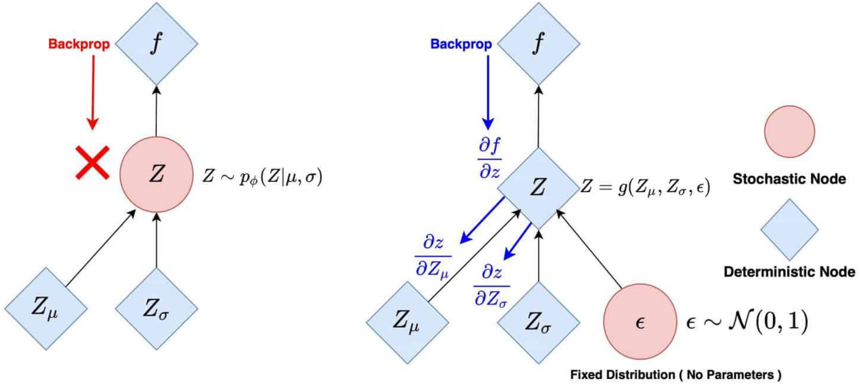


1. Given change of variable, expectations can be rewritten in terms of ϵ:

$$\mathbb{E}_{q_\phi(\mathbf{z}\,|\,\mathbf{x})}[f(\mathbf{z})] = \mathbb{E}_{p(\epsilon)}[f(\mathbf{z})]$$

2. Then, the expectation and gradient operators become commutative, and we can form a simple Monte Carlo estimator

$$\begin{aligned}\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z}\,|\,\mathbf{x})}[f(\mathbf{z})] &= \nabla_\phi \mathbb{E}_{p(\epsilon)}[f(\mathbf{z})] \\ &= \mathbb{E}_{p(\epsilon)}[\nabla_\phi f(\mathbf{z})] \\ &\approx \nabla_\phi f(\mathbf{z})\end{aligned}$$



1. Under the reparameterization, we can replace an expectation with respect to $q_\phi(\mathbf{z} \mid \mathbf{x})$ with one with respect to $p(\epsilon)$.

2. The ELBO can be rewritten as:

$$\begin{aligned}\mathcal{L}(p.q)(\mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z} \mid \mathbf{x})}\left[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z} \mid \mathbf{x})\right] \\ &= \mathbb{E}_{p(\epsilon)}\left[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z} \mid \mathbf{x})\right]\end{aligned}$$

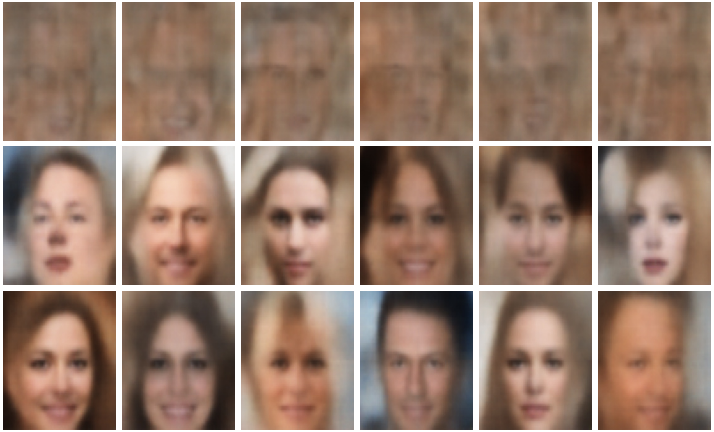
3. As a result we can form a simple Monte Carlo estimator $\widehat{\mathcal{L}}(p.q)(\mathbf{x})$ of the individual-datapoint ELBO, where we use a single noise sample ϵ from $p(\epsilon)$:

$$\widehat{\mathcal{L}}(p.q)(\mathbf{x}) = \log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z} \mid \mathbf{x})$$

4. **HW:** Show that this gradient is an unbiased estimator of the exact single-datapoint ELBO gradient.

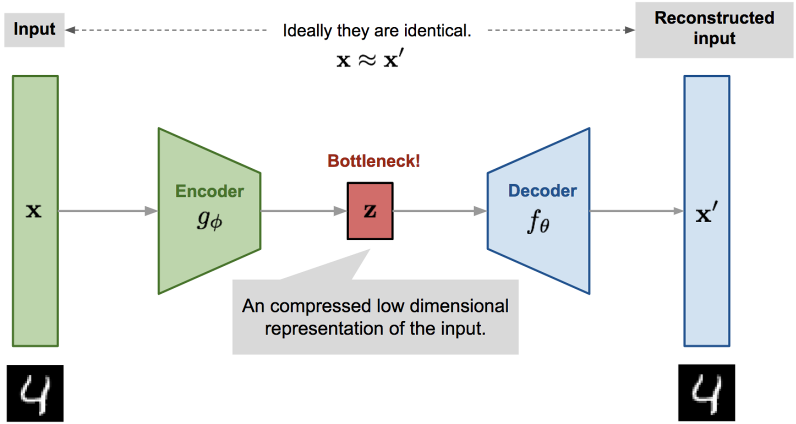
Results of VAE

1. Consider the first and third rows of the following figure:
   - The first line is deterministic autoencoder
   - The third row is VAE



2. Despite success on small scale datasets, when applied to more complex datasets such as natural images, samples tend to be unrealistic and blurry.

1. Consider an autoencoder



2. The goal of autoencoder is to minimize reconstruction loss given by

$$\min_{\theta,\phi}\{\|\mathbf{X} - \widehat{\mathbf{X}}\|^2\}$$

3. This means that a good intermediate representation not only can capture latent variables, but also benefits a full decompression process.

1. Recall that the goal of VAE is to estimate $p(\mathbf{x}, \mathbf{z})$ as

$$p(\mathbf{z})\, p(\mathbf{x} \mid \mathbf{z}) = p(\mathbf{x}) = p(\mathbf{x})\, p(\mathbf{z} \mid \mathbf{x})$$

2. Then, we learn both $p(\mathbf{x} \mid \mathbf{z})$ and $p(\mathbf{z} \mid \mathbf{x})$.
3. We need to pick one:
   - Assume that $p(\mathbf{x} \mid \mathbf{z})$ is simple and then try to find a complex $p(\mathbf{z} \mid \mathbf{x})$;
   - Assume that $p(\mathbf{z} \mid \mathbf{x})$ is simple and find a complex $p(\mathbf{x} \mid \mathbf{z})$.
4. VAEs take the first option and assume that $p(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}(f(\mathbf{z}), \sigma \mathbf{I})$.
5. Thus, $p(\mathbf{z})\, p(\mathbf{x} \mid \mathbf{z})$ is a Gaussian distribution.
6. On the other hand, what about $p(\mathbf{x})\, p(\mathbf{z} \mid \mathbf{x})$?
7. We need to find a very complex distribution $p(\mathbf{z} \mid \mathbf{x})$.
8. There are several ways, where VAE uses approximation: the encoder produces a simple distribution $q_{\phi}(\mathbf{z}) = \mathcal{N}(\mu_{\phi}, \sigma_{\phi} \mathbf{I})$.
9. This approximation states that $q_{\phi}(\mathbf{z}) \approx p(\mathbf{z} \mid \mathbf{x})$.

1. Now consider the loss function of VAE

$$\begin{aligned}\mathcal{L}(p,q) &= \int \mathrm{q}(\mathbf{z}) \log \frac{\mathrm{p}(\mathbf{x},\mathbf{z})}{\mathrm{q}(\mathbf{z})} = \int \mathrm{q}(\mathbf{z}) \log \frac{\mathrm{p}(\mathbf{x} \mid \mathbf{z})\,\mathrm{p}(\mathbf{z})}{\mathrm{q}(\mathbf{z})} \\ &= \int \mathrm{q}(\mathbf{z}) \log \mathrm{p}(\mathbf{x} \mid \mathbf{z}) d\mathbf{z} + \int \mathrm{q}(\mathbf{z}) \log \frac{\mathrm{p}(\mathbf{z})}{\mathrm{q}(\mathbf{z})} d\mathbf{z} \\ &= \int \mathrm{q}(\mathbf{z}) \log \mathcal{N}(f(\mathbf{z}), \sigma \mathbf{I}) d\mathbf{z} - \mathrm{D}_{KL}(\,\mathrm{q}(\mathbf{z}) \;||\; \mathrm{p}(\mathbf{z})) \\ &= -\frac{1}{\sigma}\|\mathbf{x} - f(\mathbf{z})\|^2 - \mathrm{D}_{KL}(\,\mathrm{q}(\mathbf{z}) \;||\; \mathrm{p}(\mathbf{z}))\end{aligned}$$

2. Hence, we need to maximize the above objective function:
   - The first term wants to make $f(\mathbf{z})$ as close as possible to $\mathbf{x}$.
   - The second term wants to make $\mathrm{q}(\mathbf{z})$ as close as possible to $\mathrm{p}(\mathbf{z})$.
3. In other words, we want to minimize the following objective function

$$-\mathcal{L}(p,q) = \underbrace{\frac{1}{2\sigma}\|\mathbf{x} - f(\mathbf{z})\|^2}_{\text{Reconstruction Loss: } \mathcal{L}_{rec}} + \underbrace{\mathrm{D}_{KL}(\,\mathrm{q}(\mathbf{z}) \;||\; \mathrm{p}(\mathbf{z}))}_{\text{Regularization Loss: } \mathcal{L}_{reg}}$$

4. On solution is to enrich encoder and decoder networks.

Variants of VAE

1. How can we interpret the latent vector of VAE?
2. A model trained on photos of human faces might capture
   - gentle,
   - skin color,
   - hair color,
   - hair length,
   - emotion,
   - glasses (if any),
   - many other relatively independent factors.
3. Such a disentangled representation is very beneficial to facial image generation.

1. Beta-VAE is a modification of VAE with a special emphasis to discover disentangled latent factors (Higgins et al. 2017).

2. In β-VAE, we want to maximize the probability of generating real data, while keeping the distance between the real and estimated posterior distributions small (less than a small constant δ):

$$\max_{\phi,\theta} \mathbb{E}_{\mathbf{x}\sim \mathrm{p}_{data}(\mathbf{x})}\left[\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})} \log \mathrm{p}_\theta(\mathbf{x} \mid \mathbf{z})\right] \text{ subject to } \quad \mathrm{D}_{KL}\big(\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x}) \mid\mid \mathrm{p}(\mathbf{z})\big) \leq \delta$$

3. We can rewrite it as a Lagrangian with a Lagrangian multiplier β under the KKT condition.

4. The above optimization problem with only one inequality constraint is equivalent to maximizing $\mathcal{F}(\phi, \theta, \beta)$ as :

$$\begin{aligned}
\mathcal{F}(\phi, \theta, \beta) &= \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})} \log \mathrm{p}_\theta(\mathbf{x} \mid \mathbf{z}) - \beta\big(\mathrm{D}_{KL}\big(\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x}) \mid\mid \mathrm{p}(\mathbf{z})\big) - \delta\big) \\
&= \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})} \log \mathrm{p}_\theta(\mathbf{x} \mid \mathbf{z}) - \beta\,\mathrm{D}_{KL}\big(\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x}) \mid\mid \mathrm{p}(\mathbf{z})\big) + \beta\delta \\
&\geq \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})} \log \mathrm{p}_\theta(\mathbf{x} \mid \mathbf{z}) - \beta\,\mathrm{D}_{KL}\big(\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x}) \mid\mid \mathrm{p}(\mathbf{z})\big) \qquad \text{Since } \beta, \delta \geq 0
\end{aligned}$$

Beta-VAE

1. Hence, the loss function of beta-VAE is defined as

$$\mathcal{L}_{\mathrm{BETA}}(\phi, \theta, \beta) = -\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}\,|\,\mathbf{x})} \log p_\theta(\mathbf{x} \mid \mathbf{z}) + \beta\, D_{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \,||\, p(\mathbf{z})\big)$$

   where the Lagrangian multiplier β is considered as a hyper-parameter.

2. Since $-\mathcal{L}_{\mathrm{BETA}}(\phi, \theta, \beta)$ is the lower bound of the Lagrangian, minimizing the loss is equivalent to maximizing the Lagrangian.

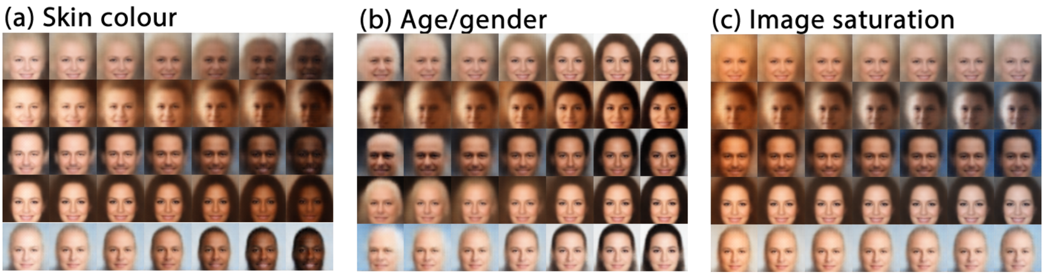
3. Considering β,
   - when $\beta = 1$, we have standard VAE, and
   - when $\beta > 1$, it applies a stronger constraint on the latent bottleneck and limits the representation capacity of $\mathbf{z}$.

4. For some conditionally independent generative factors, keeping them disentangled is the most efficient representation.

5. Therefore a higher β encourages more efficient latent encoding and hence disentanglement.

6. Hence, a higher β may create a trade-off between reconstruction quality and the extent of disentanglement.

1. Consider Latent factors learnt by β-VAE on celebA data set.



2. This experiment shows that β-VAE discovers some factors in an unsupervised manner that encode skin colour, transition from an elderly male to younger female, and image saturation.

1. Consider the loss function of Beta-VAE

$$\mathcal{L}_{\text{BETA}}(\phi, \theta, \beta) = -\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})} \log \mathrm{p}_\theta(\mathbf{x} \mid \mathbf{z}) + \beta\, \mathrm{D}_{KL}\big(\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x}) \,||\, \mathrm{p}(\mathbf{z})\big)$$

2. Assume that every pixel x_k is conditionally independent given $\mathbf{z}$.Then, the first term becomes

$$\begin{aligned}\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})} \log \mathrm{p}_\theta(\mathbf{x} \mid \mathbf{z}) &= \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})} \log \prod_k \mathrm{p}_\theta(x_k \mid \mathbf{z}) \\ &= \mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})} \sum_k \log \mathrm{p}_\theta(x_k \mid \mathbf{z})\end{aligned}$$

3. Dividing both sides of $\mathcal{L}_{\text{BETA}}(\phi, \theta, \beta)$ by n produces

$$\mathcal{L}_{\text{BETA}}(\phi, \theta, \beta) \approx -\mathbb{E}_{\mathbf{z}\sim \mathrm{q}_\phi(\mathbf{z}\,|\,\mathbf{x})}[\,\mathbb{E}_k[\log \mathrm{p}_\theta(x_k \mid \mathbf{z})]] + \frac{\beta}{n} \mathrm{D}_{KL}\big(\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x}) \,||\, \mathrm{p}_\theta(\mathbf{z})\big) \tag{1}$$

4. We design β-VAE to learn conditionally independent factors of variation in the data.

5. Hence we assume conditional independence of every latent z_m given x.

1. Since our prior $p(\mathbf{z})$ is an isotropic unit Gaussian, we can re-write the second term of $\mathcal{L}_{\text{BETA}}(\phi, \theta, \beta)$ as:

$$
\begin{aligned}
D_{KL}\left(q_\phi(\mathbf{z} \mid \mathbf{x}) \mid\mid p(\mathbf{z})\right) &= \int_{\mathbf{z}} q_\phi(\mathbf{z} \mid \mathbf{x}) \log \frac{q_\phi(\mathbf{z} \mid \mathbf{x})}{p(\mathbf{z})} \\
&= \sum_k \int_{z_k} q_\phi(z_k \mid \mathbf{x}) \log \frac{q_\phi(z_k \mid \mathbf{x})}{p(z_k)} \\
&= K\, \mathbb{E}_k \left[\int_{z_k} q_\phi(z_k \mid \mathbf{x}) \log \frac{q_\phi(z_k \mid \mathbf{x})}{p(z_k)} \right]
\end{aligned}
$$

where K is dimensionality of latent variable

2. Combining the above terms produces

$$
\mathcal{L}_{\text{BETA}}(\phi, \theta, \beta) \approx -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{x})}\left[\mathbb{E}_k[\log p_\theta(x_k \mid \mathbf{z})]\right] + \frac{K\beta}{n}\, \mathbb{E}_k \left[\int_{z_k} q_\phi(z_k \mid \mathbf{x}) \log \frac{q_\phi(z_k \mid \mathbf{x})}{p(z_k)} \right] \tag{2}
$$

3. Hence, $\beta_{norm} = \frac{K\beta}{n}$, which is equivalent to optimising the original β-VAE.

Vector Quantized-Variational AutoEncoder

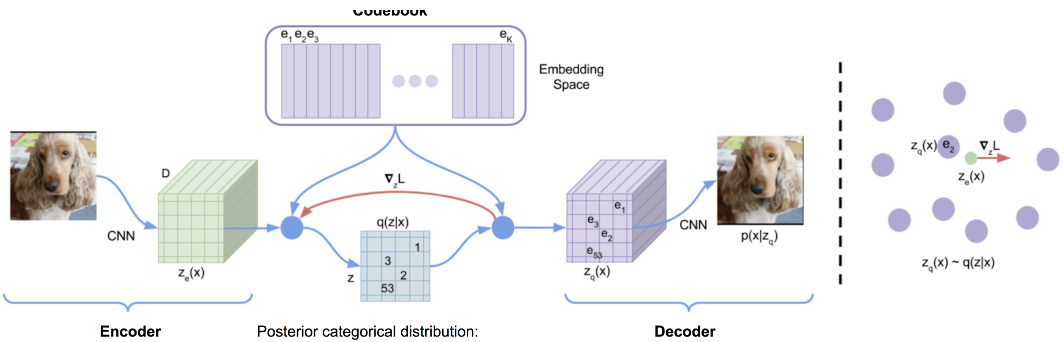
1. The VQ-VAE model learns a discrete latent variable by the encoder (Oord, Vinyals, and Kavukcuoglu 2017).
2. Discrete representations may be a more natural fit for problems like language, speech, reasoning.
3. Vector quantisation is a method to map d-dimensional vectors into a finite set of **code** vectors.
4. Let $\mathbf{E}$ be the latent embedding space, codebook in VQ-VAE.
5. An individual embedding vector is $\mathbf{e}_i$ (for $i = 1 \ldots, M$).
6. The encoder output $E(\mathbf{x}) = \mathbf{z_e}$ goes through a nearest-neighbor lookup to match to one of M embedding vectors.

$$\mathbf{z}_q(\mathbf{x}) = \text{Quantize}(E(\mathbf{x})) = \mathbf{e}_k \qquad \text{where } k = \arg\min_i \|\mathbf{x} - \mathbf{e}_i\|_2$$

7. Then this matched code vector, $E(\mathbf{x}) = \mathbf{z_e}$, becomes the input for the decoder $D(.)$.

Vector Quantised-Variational AutoEncoder

1. Note that the discrete latent variables can have different shapes in different applications; for example,
   - 1D for speech,
   - 2D for image, and
   - 3D for video
2. Hence, the VQ-VAE becomes .



1. Here, we have

$$q(\mathbf{z} = \mathbf{e}_k \mid \mathbf{x}) = \begin{cases} 1 & \text{if } k = \arg\min_i \|\mathbf{z}_e(\mathbf{x}) - \mathbf{e}_i\|_2 \\ 0 & \text{otherwise} \end{cases}$$

2. Since *argmin* is non-differentiable on a discrete space, the gradients $\mathcal{L}_{VQ-VAE}$ from decoder input $\mathbf{z}_q$ is copied to the encoder output $\mathbf{z}_\mathbf{e}$.

3. Loss function for VQ-VAE is

$$\mathcal{L}_{VQ-VAE} = \underbrace{\|\mathbf{x} - D(\mathbf{e}_k)\|_2^2}_{\text{reconstruction loss}} + \underbrace{\|\text{sg}\left[E(\mathbf{x})\right] - \mathbf{e}_k\|_2^2}_{\text{VQ loss}} + \underbrace{\beta\|\mathbf{x} - \text{sg}\left[\mathbf{e}_k\right]\|_2^2}_{\text{commitment loss}}$$

   where $\text{sg}\left[.\right]$ is the stop gradient operator.

4. The embedding vectors in the codebook is updated through EMA (exponential moving average).

Vector Quantised-Variational AutoEncoder

1. Given a code vector $\mathbf{e}_i$, say we have n_i encoder output vectors, $\{\mathbf{z}_{i,j}\}_{j=1}^{n_i}$, that are quantized to $\mathbf{e}_i$

$$N_i^{(t)} = \gamma N_i^{(t-1)} + (1-\gamma) n_i^{(t)}$$

$$\mathbf{m}_i^{(t)} = \gamma \mathbf{m}_i^{(t-1)} + (1-\gamma) \sum_{j=1}^{n_i^{(t)}} \mathbf{z}_{i,j}^{(t)}$$

$$\mathbf{e}_i^{(t)} = \mathbf{m}_i^{(t)} / N_i^{(t)}$$

where (t) refers to batch sequence in time. N_i and $\mathbf{m}_i$ are accumulated vector count and volume, respectively.

2. Left: ImageNet $128 \times 128 \times 3$ images. Right: reconstructions from a VQ-VAE with a $32 \times 32 \times 1$ and latent space, with $K = 512$.

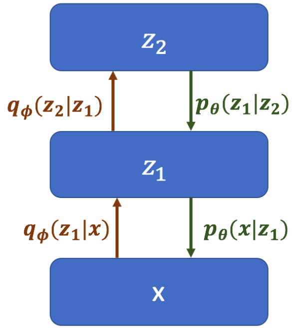
Variants of VAE

Hierarchical VAE

1. Some researchers have proposed hierarchical VAEs (Sønderby et al. 2016).

$$p(\mathbf{x} \mid \mathbf{z}) = p(\mathbf{x} \mid \mathbf{z}_1) \prod_{k=1}^{L-1} p(\mathbf{z}_k \mid \mathbf{z}_{k+1})$$

2. There are some VAEs are effectively stacked on top of each other



Hierarchical VAE

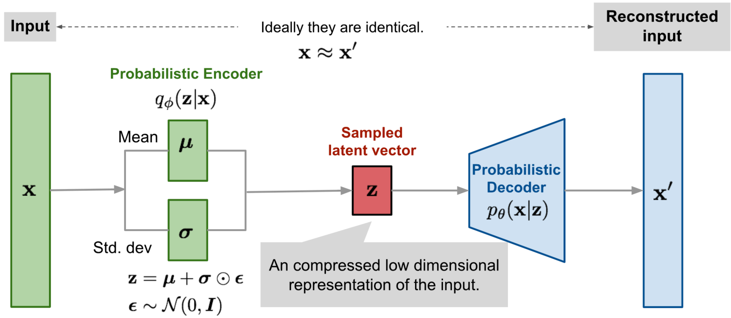
1. There are two potential advantages of using hierarchical VAEs:
   - They could improve the Evidence Lower Bound (ELBO) and decrease reconstruction error.
   - The stack of latent variables $\mathbf{z}_k$ might learn a feature hierarchy similar to those learned by convolutional neural networks.
2. It is shown that that if the purpose is to learn structured, hierarchical features, using a hierarchical VAE has limitations (Zhao, Song, and Ermon 2017).
3. **Homework:** Drive variational inference for hierarchical VAE.
4. **Homework:** Read (Sønderby et al. 2016) and (Zhao, Song, and Ermon 2017).

Variants of VAE

Conditional VAE[a]

[a] These slides are taken from Christopher Beckham's Blog

Conditional VAE



1. In a conditional VAE, we have (Kingma, Mohamed, et al. 2014)
   - a conditional generative model $\mathrm{p}_\theta(\mathbf{z}, \mathbf{x} \mid \mathbf{c})$ on latent variables $\mathbf{z}$, data $\mathbf{x}$, conditioned on $\mathbf{c}$ and parameterized by θ and
   - a conditional inference network $\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})$ conditioned on $\mathbf{c}$ and parameterized by ϕ s.t.
2. Given a true conditional data distribution $\mathrm{p}(\mathbf{x} \mid \mathbf{c})$ for all $\mathbf{c}$, we want to learn (θ, ϕ) s.t.
   - $\mathrm{p}_\theta(\mathbf{x} \mid \mathbf{c})$ approximates $\mathrm{p}(\mathbf{x} \mid \mathbf{c})$ for all $\mathbf{c}$ and
   - $\mathrm{q}_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})$ approximates $\mathrm{p}(\mathbf{z} \mid \mathbf{x}, \mathbf{c})$ for all $\mathbf{x}, \mathbf{c}$.

Conditional VAE

1. The inference process, in which latent representation is extracted from actual samples, is $q_\phi(\mathbf{x}, \mathbf{z}, \mathbf{c}) = q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})\, q(\mathbf{x}, \mathbf{c})$.

2. To obtain a sample $(\mathbf{z}, \mathbf{x}, \mathbf{c})$ from this joint we simply perform the following:

$$\begin{aligned} \mathbf{x}, \mathbf{c} &\sim q(\mathbf{x}, \mathbf{c}) && \text{ground truth} \\ \mathbf{z} &\sim q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c}) \end{aligned}$$

   where

   - $q(\mathbf{x}, \mathbf{c})$ is the ground truth data distribution,
   - $q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})$ is the learnable variational posterior.

3. The generative process, in which samples are generated, is

$$\begin{aligned} \mathbf{z}, \mathbf{c} &\sim p(\mathbf{z}, \mathbf{c}) && \text{prior} \\ \mathbf{x} &\sim p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c}) \end{aligned}$$

Conditional VAE

1. Since joint distribution for both processes are $p_\theta(\mathbf{x}, \mathbf{z}, \mathbf{c})$ and $q_\phi(\mathbf{x}, \mathbf{z}, \mathbf{c})$, we can derive their KL distribution.

2. Let $D_{KL}(q \parallel p) = D_{KL}\big(q_\phi(\mathbf{x}, \mathbf{z}, \mathbf{c}) \parallel p_\theta(\mathbf{x}, \mathbf{z}, \mathbf{c})\big)$. Thus, we have

$$
\begin{aligned}
\arg\max_{\theta,\phi} - D_{KL}(q \parallel p) &= \mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}\left[\log \frac{p_\theta(\mathbf{x}, \mathbf{z}, \mathbf{c})}{q_\phi(\mathbf{x}, \mathbf{z}, \mathbf{c})}\right] \\
&= \mathbb{E}_{q_\phi(\mathbf{z} \mid \mathbf{x},\mathbf{c})}\left[\log \frac{p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})\, p(\mathbf{x}, \mathbf{c})}{q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})}\right] - \mathbb{E}_{q(\mathbf{x},\mathbf{c})}[\log q(\mathbf{x}, \mathbf{c})] \\
&= \mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}\left[\log \frac{p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})\, p(\mathbf{z}, \mathbf{c})}{q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})}\right] - \text{const.} \\
&= \mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})] + \mathbb{E}_{q_\phi(\mathbf{z} \mid \mathbf{x},\mathbf{c})}\left[\log \frac{p(\mathbf{z}, \mathbf{c})}{q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})}\right] - \text{const.} \\
&= \mathbb{E}_{q_\phi(\mathbf{z},\mathbf{x},\mathbf{c})}[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})] - D_{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c}) \parallel p(\mathbf{z}, \mathbf{c})\big)
\end{aligned}
$$

3. This gives the typical formulation of the ELBO which we see in most VAE papers.

4. Now, we must specify $p(\mathbf{z}, \mathbf{c})$. We have two cases
   - When they are independent
   - When they are dependent

Conditional VAE (When z and c are independent)

1. When $\mathbf{z}$ and $\mathbf{c}$ are independent, we have $p(\mathbf{z}, \mathbf{c}) = p(\mathbf{z})\, p(\mathbf{c})$.

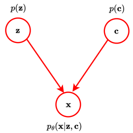
2. This means that the joint distribution of the generative process factorises into:

$$p_\theta(\mathbf{x}, \mathbf{z}, \mathbf{c}) = p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})\, p(\mathbf{z})\, p(\mathbf{c})$$

3. This leads us to the following ELBO

$$\begin{aligned} -D_{KL}\big(q_\phi(\mathbf{x}, \mathbf{z}, \mathbf{c}) \,||\, p_\theta(\mathbf{x}, \mathbf{z}, \mathbf{c})\big) &= \mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})] \\ &\quad + \mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}\left[\log \frac{p(\mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})}\right] + \log p(\mathbf{c}) \\ &= \text{likelihood} - D_{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c}) \,||\, p(\mathbf{z})\big) + \text{constants}. \end{aligned}$$

4. Here, $p(\mathbf{c})$ is prior for $\mathbf{c}$ but it falls out of the KL term since it is a constant.



- This factorisation is useful to encode if we are seeking to learn disentangled representations.
- This would make for a very controllable generative process where we could arbitrarily mix and match style and content variables from different examples to create new ones.

Conditional VAE (When z and c are dependent)

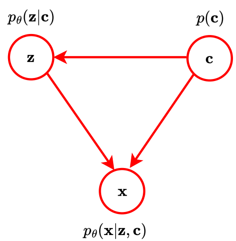
1. In general $\mathbf{z}$ and $\mathbf{c}$ may not be independent, we have $p(\mathbf{z}, \mathbf{c}) = p(\mathbf{z} \mid \mathbf{c})\, p(\mathbf{c})$.
2. This means that the joint distribution of the generative process factorises into:

$$p_\theta(\mathbf{x}, \mathbf{z}, \mathbf{c}) = p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})\, p(\mathbf{z} \mid \mathbf{c})\, p(\mathbf{c})$$

3. This leads us to the following ELBO

$$\begin{aligned} -D_{KL}\big(\, q_\phi(\mathbf{x}, \mathbf{z}, \mathbf{c}) \,||\, p_\theta(\mathbf{x}, \mathbf{z}, \mathbf{c})\big) &= \mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})] \\ &+ \mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}\left[\log \frac{p(\mathbf{z} \mid \mathbf{c})}{q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c})}\right] + \log p(\mathbf{c}) \\ &= \text{likelihood} - D_{KL}\big(\, q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c}) \,||\, p(\mathbf{z} \mid \mathbf{c})\big) + \text{constants}. \end{aligned}$$

4. Here, $p(\mathbf{c})$ is prior for $\mathbf{c}$ but it falls out of the KL term since it is a constant.



The role of the beta term

1. By looking at both versions of the ELBO, we can write them as:

$$\min_{\theta,\phi}\{-\mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})] + \beta\, D_{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c}) \mid\mid p(\mathbf{z} \mid \mathbf{c})\big)\} \quad \text{Dependent case}$$

$$\min_{\theta,\phi}\{-\mathbb{E}_{q_\phi(\mathbf{x},\mathbf{z},\mathbf{c})}[\log p_\theta(\mathbf{x} \mid \mathbf{z}, \mathbf{c})] + \beta\, D_{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{c}) \mid\mid p(\mathbf{z})\big)\} \quad \text{Independent case}$$

2. The first equation is maximising the likelihood of the data with respect to samples from the inference network.
3. In order for this to happen, $\mathbf{z}$ should encode as much information about $\mathbf{x}$ as possible through the variational posterior q_ϕ, which is our learned encoder.
4. The second term is working against the first, because it is enforcing that each per example variational posterior must be close to the prior distribution.
5. Since the prior is not a function of $\mathbf{x}$, it implies that some information about $\mathbf{x}$ in the encoding pathway has to be lost.
6. **Homework:** Please above optimization functions from mutual information perspective, and describe what happens.
7. **Homework:** Please read (Rathakumar et al. 2023), (Guo et al. 2024) and (Harvey, Naderiparizi, and Wood 2022).

The results for conditional VAE

References

1. Peper An Introduction to Variational Autoencoders (Kingma and Welling 2019).
2. Chapter 21 of Probabilistic Machine Learning: Advanced Topics (Murphy 2023).
3. Chapter 4 of Deep Generative Modeling (Tomczak 2022).

- Guo, Zhiqiang et al. (2024). *DualVAE: Dual Disentangled Variational AutoEncoder for Recommendation*.
- Harvey, William, Saeid Naderiparizi, and Frank Wood (2022). "Conditional Image Generation by Conditioning Variational Auto-Encoders". In: *The Tenth International Conference on Learning Representations*.
- Higgins, Irina et al. (2017). "Beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: *International Conference on Learning Representations*.
- Kingma, Diederik P., Shakir Mohamed, et al. (2014). "Semi-supervised Learning with Deep Generative Models". In: *Advances in Neural Information Processing Systems*, pp. 3581–3589.
- Kingma, Diederik P. and Max Welling (2019). "An Introduction to Variational Autoencoders". In: *Foundations and Trends in Machine Learning* 12.4, pp. 307–392.
- Murphy, Kevin P. (2023). *Probabilistic Machine Learning: Advanced Topics*. The MIT Press.
- Oord, Aäron van den, Oriol Vinyals, and Koray Kavukcuoglu (2017). "Neural Discrete Representation Learning". In: *Advances in Neural Information Processing Systems*, pp. 6306–6315.
- Rathakumar, Keerth et al. (2023). *DualVAE: Controlling Colours of Generated and Real Images*.

- Sønderby, Casper Kaae et al. (2016). "Ladder Variational Autoencoders". In: *Advances in Neural Information Processing Systems*, pp. 3738–3746.
- Tomczak, Jakub M. (2022). *Deep Generative Modeling*. Springer.
- Zhao, Shengjia, Jiaming Song, and Stefano Ermon (2017). "Learning Hierarchical Features from Deep Generative Models". In: *International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70, pp. 4091–4099.

Questions?