Variational Autoencoders (VAEs)

Jayin Khanna
CSD662: ADL

Latent Variable Models & The Learning Goal

1. What is a Latent Variable Model?

We assume our observed data $\mathbf{D = \{x_1, x_2, \dots, x_N\}}$ is generated from some true but unknown distribution $\mathbf{p_x(x)}$.
A latent variable model explains this data by introducing an unobserved (latent) variable $\mathbf{z}$.

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

or equivalently

$$\mathbf{p\theta(x) = \int_{(z)} p\theta(x, z)\, dz}$$

Intuition:

- **Discrete z (e.g., $z \in \{1, \dots, M\}$):**
  The latent variable $\mathbf{z_i}$ acts as a cluster assignment for each data point $\mathbf{x_i}$.
  → Example: Gaussian Mixture Models (GMMs)
- **Continuous z (e.g., $z \in \mathbb{R}^k$, with $k \ll d$):**
  The latent variable $\mathbf{z_i}$ is a low-dimensional feature vector or compressed representation of $\mathbf{x_i}$.
  → Foundation of Autoencoders and Variational Autoencoders (VAEs)

Example — Mixture of Gaussians (Discrete Latent Variable) 1D Mixture of Gaussians

In a **1D mixture of Gaussians**, the latent variable **z** is **discrete**, and the prior **Pr(z)** is a **categorical distribution** with one probability λ_n for each possible value of **z**.

$$\mathbf{Pr(z = n) = \lambda_n}$$

The **likelihood** of the data **x** given **z = n** is normally distributed with mean μ_n and variance σ_n^2:

$$\mathbf{Pr(x \mid z = n) = \mathcal{N}_x(\mu_n, \sigma_n^2)}$$

By marginalizing over all possible latent variable values, we get the overall data distribution:

$$\begin{aligned} Pr(x) &= \sum_{n=1}^{N} Pr(x, z = n) \\ &= \sum_{n=1}^{N} Pr(x|z = n) \cdot Pr(z = n) \\ &= \sum_{n=1}^{N} \lambda_n \cdot \mathrm{Norm}_x[\mu_n, \sigma_n^2]. \end{aligned}$$

Interpretation:
From **simple components** (a prior over z and Gaussian likelihoods), we can construct a **complex multi-modal distribution**.
Each mode corresponds to a **cluster**, and the mixture weights $\boldsymbol{\lambda_n}$ control their relative importance.

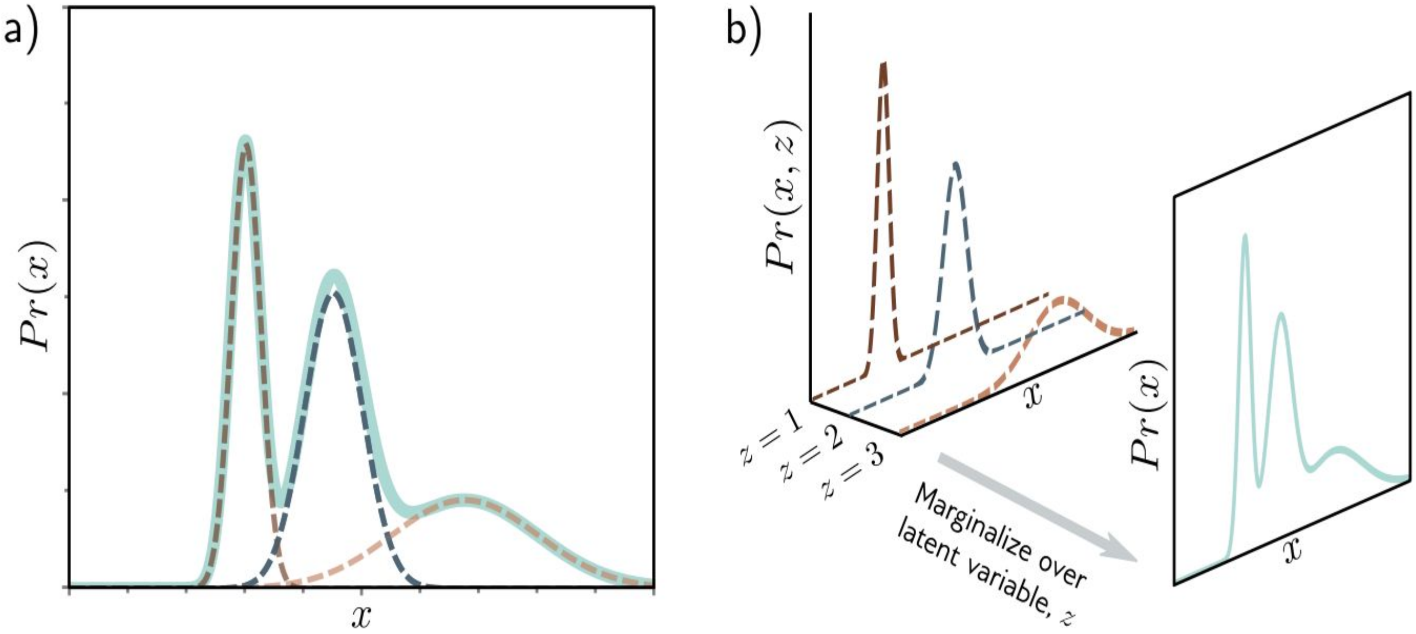


Figure 17.1 Mixture of Gaussians (MoG). a) The MoG describes a complex probability distribution (cyan curve) as a weighted sum of Gaussian components (dashed curves). b) This sum is the marginalization of the joint density $Pr(x, z)$ between the continuous observed data x and a discrete latent variable z.

Nonlinear Latent Variable Models

Continuous Multivariate Latent Variable Model

In the **nonlinear latent variable model**, both the data **x** and the latent variable **z** are **continuous and multivariate**.

Prior:

$$\mathbf{Pr(z) = \mathcal{N}\_z(0, I)}$$

Likelihood:

The likelihood of **x** given **z** and parameters **ϕ** is also Gaussian,
but its **mean** is a **nonlinear function** of **z**, modeled by a neural network **f[z, ϕ]**, and its covariance is $\boldsymbol{\sigma^2 I}$ (spherical):

$$\mathbf{Pr(x \mid z, \phi) = \mathcal{N}_x(f[z, \phi], \sigma^2 I)}$$

Marginalizing over z:

$$
\begin{aligned}
Pr(\mathbf{x}|\phi) &= \int Pr(\mathbf{x}, \mathbf{z}|\phi) d\mathbf{z} \\
&= \int Pr(\mathbf{x}|\mathbf{z}, \phi) \cdot Pr(\mathbf{z}) d\mathbf{z} \\
&= \int \mathrm{Norm}_{\mathbf{x}}\left[\mathbf{f}[\mathbf{z}, \phi], \sigma^2\mathbf{I}\right] \cdot \mathrm{Norm}_{\mathbf{z}}\left[\mathbf{0}, \mathbf{I}\right] d\mathbf{z}.
\end{aligned}
$$

Interpretation:

This can be viewed as an **infinite mixture of Gaussians**,
where each **z** contributes a Gaussian centered at **f[z, ϕ]**,
weighted by **Pr(z)**.The network **f[z, ϕ]** captures the key structure of the data, and $\boldsymbol{\sigma^2 I}$ represents residual noise.

General Principle for Learning Latent Variable Models

Suppose we are given data

$$\mathcal{D} = \{x_i\}_{i=1}^n \quad \text{i.i.d.} \quad \sim p_X.$$

Let $p_\theta(x) = \int_z p_\theta(x,z)\,dz$ be a latent variable model. The goal is to estimate the model parameters θ given data $\mathcal{D}$.

$$\theta^* = \arg\min_\theta \mathrm{KL}(p_X \| p_\theta).$$

Expanding the KL divergence:

$$\theta^* = \arg\min_\theta \left[\int_x p_X(x) \log \frac{p_X(x)}{p_\theta(x)}\,dx \right].$$

Dropping the constant entropy term $H(p_X) = -\int_x p_X(x) \log p_X(x)\,dx$ (independent of θ), this becomes:

$$\theta^* = \arg\min_\theta \left[-\int_x p_X(x) \log p_\theta(x)\,dx \right].$$

- The KL minimization objective compares the true data distribution p_X with the model distribution p_θ. Minimizing it makes the model approximate the data as closely as possible.

$$\theta^* = \arg\max_\theta \mathbb{E}_{p_X}[\log p_\theta(x)].$$

Denoting the log-likelihood as $\ell(\theta) = \log p_\theta(x)$, this is precisely the **Maximum Likelihood Estimation (MLE)** principle:

$$\theta^* = \arg\max_\theta \ell(\theta) = \arg\max_\theta \mathbb{E}_{p_X}[\log p_\theta(x)].$$

In a latent variable model:

$$p_\theta(x) = \int_z p_\theta(x,z)\,dz,$$

so the log-likelihood involves integrating out the latent variable z.

Evidence Lower Bound (ELBO)

Consider $\ell(\theta) = \log p_\theta(x)$

$$= \log \int_z p_\theta(x,z)\,dz$$

Sps $q(z|x)$ denote a distn over latent variable, z

$$\ell(\theta) = \log \int_z p_\theta(x,z) \frac{q(z|x)}{q(z|x)}\,dz$$

$$= \log \int_z q(z|x) \left[\frac{p_\theta(x,z)}{q(z|x)}\right] dz$$

Can't Calculate! hence, this

$$= \underbrace{\log \left[\mathbb{E}_{q(z|x)} \left(\frac{p_\theta(x,z)}{q(z|x)} \right) \right]}$$

we can not estimate the log of an Expectation using the statistical Estimators, hence we need to pull out the expectation outside the log.

$\theta^* = \arg\max_\theta \ell(\theta)$: orignal max liklihood problem

$\theta^* = \arg\max_{\theta, q} J_\theta(q)$: Approx with ELBO

The fundamental problem solved in any latent var generative model is given below

ELBO maximization, find θ & q that maximizes the ELBO, $J_\theta(q)$

$$\boxed{\theta^*, q^* = \arg\max_{\theta, q} \mathbb{E}_{q(z|x)} \frac{p_\theta(x,z)}{q(z|x)}}$$

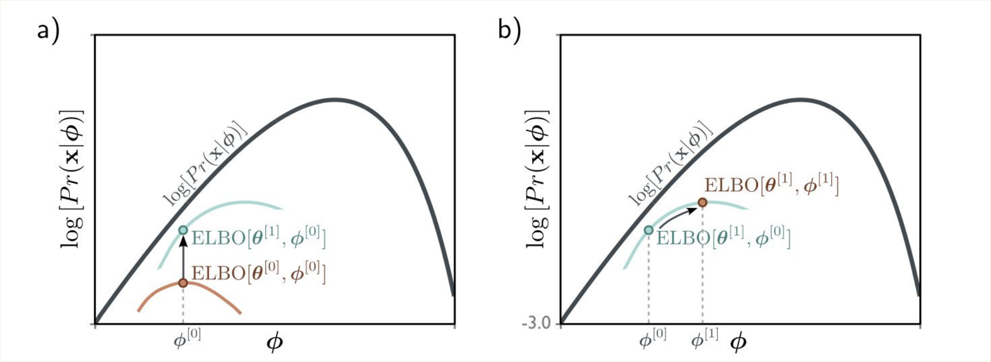
Jensen's Inequality

$$\boxed{\text{Jensen's Inequality:} \quad \log \mathbb{E}(\ ) \geq \mathbb{E} \log(\ )}$$

$$\ell(\theta) = \log \mathbb{E}_{q(z|x)} \left(\frac{p_\theta(x,z)}{q(z|x)} \right) \geq \underbrace{\mathbb{E}_{q(z|x)} \log \left(\frac{p_\theta(x,z)}{q(z|x)} \right)}_{\text{denote with } J_\theta(q)}$$

$\ell(\theta) \geq J_\theta(q)$, called the Evidence lower Bound [ELBO]

$J_\theta(q)$ is a fxn of both the model parameters θ & the densit on z, $q(z|x)$



ENCODER AND DECODER IN VAEs

*) Goal of a Neural lat. var. model:

a) Learn a lat. var. model with unknown $p_\theta(z|x)$

b.) Enable sampling / generation from $p_\theta(z|x)$

c) Enable posterior inference, Compute / estimate $q(z|x)$

Consider $J_\theta(q) = \underset{q(z|x)}{E} \log \dfrac{p_\theta(x, z)}{q(z|x)}$

$= \underset{q(z|x)}{E} \log p_\theta(x|z) - \underset{q(z|x)}{E} \log \dfrac{q(z|x)}{p_\theta(z)}$

$$\boxed{J_\theta(q) = \underset{q(z|x)}{E} \log p_\theta(x|z) - D_{KL}(q(z|x) \,||\, p_\theta(z))}$$

*) Represent $p_\theta(x|z)$ & $q(z|x)$ via Neural network

$p_\theta(x|z)$: Conditional data liklihood

$q(z|x)$: Variational latent posterior density

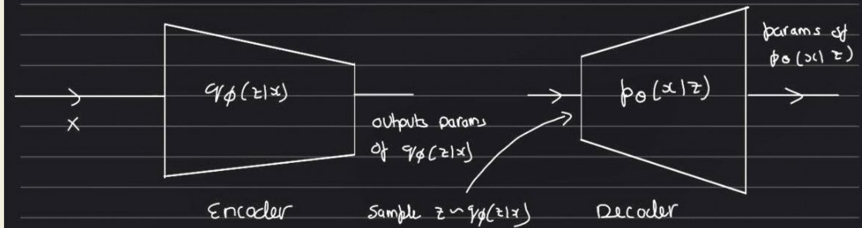
Given $D = \{x_i\}_{i=1}^{n} \sim$ iid p_x

$p_\theta(x) = \int_z p_\theta(x, z)\, dz$: model

The follow^n objective is optimized

$J_\theta(q) = \underset{q(z|x)}{E} \log p_\theta(x|z) - D_{KL}(q(z|x) \,||\, p_\theta(z))$

$q(z|x)$ & $p_\theta(x|z)$ are rep using prob NNs



There is no direct connection b/w the encoder & Dec

ϕ & θ : the weights of Enc & Dec resp.

$\phi^*, \theta^* = \underset{\theta, \phi}{\arg\max}\, J_\theta(q_\phi)$

$\phi^{t+1} \longleftarrow \phi^t + \alpha \nabla_\phi J_\theta(q_\phi)$

$\theta^{t+1} \longleftarrow \theta^t + \alpha \nabla_\theta J_\theta(q_\phi)$

The Reparameterization trick

*) To compute the gradients of $J_\theta(q_\phi)$ w.r.t ϕ:

$$\nabla_\phi \underset{q_\phi(z|x)}{\mathbb{E}} \log p_\theta(x|z) \quad , \quad z \sim q_\phi(z|x)$$

$$\nabla_\phi \underset{q_\phi(z|x)}{\mathbb{E}} \log p_\theta(x|z) = \nabla_\phi \int_z q_\phi(z|x) \cdot \log p_\theta(x|z)$$

$$= \int \nabla_\phi q_\phi(z|x) \cdot \log p_\theta(x|z)$$

[since gradient is a $\underbrace{\text{linear operator}}_{\text{Explain!!}}$]

$$= \int_z q_\phi(z|x) \left(\nabla_\phi \log p_\theta(x|z)\right) + \underbrace{\int_z \left(\nabla_\phi q_\phi(z|x)\right) \log p_\theta(x|z)}$$

is not an Expectation

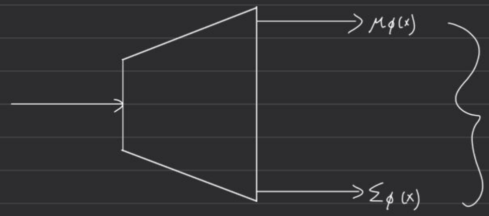
$\therefore$ can't be computed

There is one more complication; the network involves a sampling step, and it is difficult to differentiate through this stochastic component. However, differentiating past this step is necessary to update the parameters $\boldsymbol{\theta}$ that precede it in the network.

Fortunately, there is a simple solution; we can move the stochastic part into a branch of the network that draws a sample $\boldsymbol{\epsilon}^*$ from $\text{Norm}_{\epsilon}[\mathbf{0}, \mathbf{I}]$ and then use the relation:

$$\mathbf{z}^* = \boldsymbol{\mu} + \boldsymbol{\Sigma}^{1/2}\boldsymbol{\epsilon}^*, \tag{17.25}$$

let $q_\phi(z|x) = \mathcal{N}(z; \mu_\phi(x), \Sigma_\phi(x))$



Params of $q_\phi(z|x)$ for a given input x

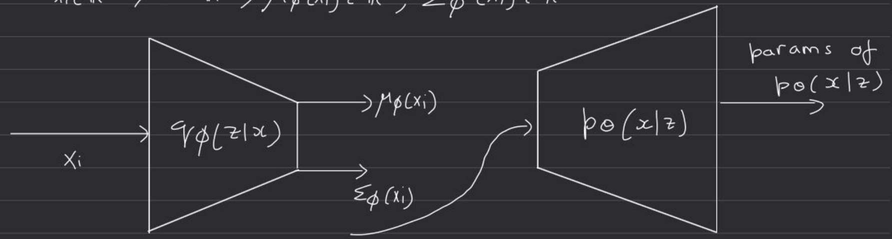
let $\varepsilon \sim N(0, I)$, then

$$z = \mu_\phi(x) + \Sigma_\phi(x) \cdot \varepsilon = g(\varepsilon)$$

$$z \sim \mathcal{N}(z; \mu_\phi(x), \Sigma_\phi(x))$$

then we have

$x_i \in \mathbb{R}^d$; $z_i \in \mathbb{R}^k$; $\mu_\phi(x_i) \in \mathbb{R}^k$; $\Sigma_\phi(x_i) \in \mathbb{R}^{k \times k}$



$\varepsilon_i \sim N(0, I)$, $z_i = \mu_\phi(x_i) + \Sigma_\phi(x_i) \cdot \varepsilon$

hence,

$$\nabla_\phi \mathop{E}_{q_\phi(z|x)} \log p_\theta(x|z) = \nabla_\phi \mathop{E}_{p_\varepsilon} \log p_\theta(x|g(\varepsilon))$$

$$= \nabla_\phi \left(\frac{1}{M} \sum_{j=1}^{M} \log p_\theta(x|g(\varepsilon_j)) \right)$$

$\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_M \sim p_\varepsilon$

To compute $\log p_\theta(x \mid g(\varepsilon_j))$:

parameterize $p_\theta(x \mid z)$ via some known distribution & use the decoder to output its params

Eg: $p_\theta(x \mid z) = \mathcal{N}(x; \hat{x}_\theta(z), I)$

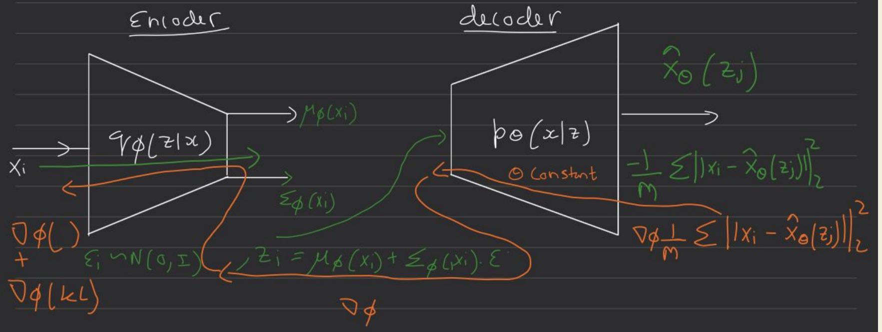
$$\Rightarrow \log p_\theta(x \mid z) = \log \left(\frac{1}{(2\pi)^{d/2}} \exp - \left[\| x - \hat{x}_\theta(z) \|_2^2 \right] \right.$$

$$= - \| x - \hat{x}_\theta(z) \|_2^2$$

hence,

$$\boxed{\begin{aligned} \nabla_\phi \mathop{\mathbb{E}}_{q_\phi(z|x)} \log p_\theta(x \mid z) &= \nabla_\phi \mathop{\mathbb{E}}_{p_\varepsilon} \log p_\theta(x \mid g(\varepsilon)) \\ &= \nabla_\phi \left[-\frac{1}{m} \sum_{j=1}^{m} \| x_i - \hat{x}_\theta(z_j) \|_2^2 \right. \\ \text{where } z_j &= \mu_\phi(x_i) + \varepsilon_j \Sigma_\phi(x_i), \\ \varepsilon_1, \varepsilon_2, \ldots, \varepsilon_j &\sim N(0, I) \end{aligned}}$$

Training the Network:
1. The Encoder



Forward pass: i) Given an $x_i \in D$, pass it through the Encoder, to obtain $\mu_\phi(x_i)$ & $\Sigma_\phi(x_i)$

ii) Sample $\varepsilon_1, \dots, \varepsilon_m \sim N(0, I)$ outside NNs

iii) Compute $z_1, \dots, z_m$ via reparametrization

iv) pass all of $z_1, \dots, z_j$ through the Decoder to compute $\hat{x}_\theta(z_j)$

v) Compute $-\frac{1}{m}\sum_{j=1}^{m}\|x_i - \hat{x}_\theta(z_j)\|_2^2$

training for Encoder (one training step)

$$\phi^{t+1} \longleftarrow \phi^t + \alpha \nabla_\phi J_\theta(q_\phi)$$

Backward pass:

i) Compute the grad $\nabla_\phi\left(-\frac{1}{m}\sum_{j=1}^{m}\|x_i - \hat{x}_\theta(z_j)\|_2^2\right)$

ii) Backprop through
→ Decoder, with fixed param θ
→ Reparam, Id of φ
→ Encoder

Thus we compute $\nabla_\phi \mathbb{E}_{q_\phi(z|x)} \log p_\theta(x|z)$

To compute $\nabla_\phi D_{KL}[q_\phi(z|x) \| p_\theta(z)]$:

$p_\theta(z)$: Latent prior, typically assumed to be $\mathcal{N}(0, I)$

$$\therefore D_{KL}[q_\phi(z|x) \| p_\theta(z)]$$

$$= D_{KL}\left[\mathcal{N}(\mu_\phi(x), \Sigma_\phi(x)) \| \mathcal{N}(0, I)\right]$$

$$= -\frac{1}{2} \underbrace{\log|\Sigma_\phi(x)| - d + tr[\Sigma_\phi(x)]}_{\text{Can be Estimated}} + \|\mu_\phi(x)\|_2^2$$

hence,

$$\nabla_\phi D_{KL}[q_\phi(z|x) \| p_\theta(z)]$$

$$\nabla_\phi\left[-\frac{1}{2}\log|\Sigma_\phi(x)| - d + tr[\Sigma_\phi(x)] + \|\mu_\phi(x)\|_2^2\right]$$

$$= \nabla_\phi(KL)$$

2. The Decoder