Functional Data analysis

Peida Wu

IMS

December 30, 2025

VAE Loss

ELBO Derivation Steps

$$\begin{aligned}\mathcal{L}_q &= E_{z\sim q_\varphi}[\log p_\theta(x|z) + \log p(z) - \log q_\varphi(z|x)] \\ &= E_{z\sim q_\varphi}[\log p_\theta(x|z)] - E_{z\sim q_\varphi}\left[\log \frac{q_\varphi(z|x)}{p(z)}\right] \\ &= E_{z\sim q_\varphi}[\log p_\theta(x|z)] - \mathrm{KL}(q_\varphi(z|x)\|p(z))\end{aligned}$$

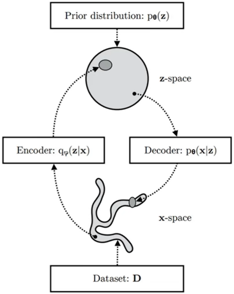


Figure 1: Progress

Why Reparameterization

$$\nabla_\phi L_{recon} = \nabla_\phi E_{q_\phi(z|x)}[\log p_\theta(x|z)]$$

Monte Carlo Sampling to compute.
Progress:

1. Encoder compute μ_ϕ, σ_ϕ
2. z $=$ sample from distribution$(mean = \mu_\phi, std = \sigma_\phi)$
3. Decoder compute reconstruction loss
   $L_{recon} = E_{q_\phi(z|x)}[\log p_\theta(x|z)]$
4. Compute $\nabla_\phi f(z) = \frac{\partial f(z)}{\partial z} \cdot \frac{\partial z}{\partial \phi}, (f(z) = f(z) = \log p_\theta(x|z))$

Since z is sampling from the latent, $\frac{\partial z}{\partial \phi}$ can't be computed.
Another problem: it'll generate great variance if we compute in math form.

The Trick: Transforming the Gradient

The Trick: $z \to g(\phi, \epsilon)$ We express z as a deterministic function g of parameters ϕ and an external noise $\epsilon \sim p(\epsilon)$ (e.g., $\mathcal{N}(0,1)$).

$$z = g(\phi, \epsilon) = \mu_\phi + \sigma_\phi \cdot \epsilon$$

The step

1. Rewrite the expectation in terms of ϵ:

$$L_{\text{recon}} = E_{q_\phi(z|x)}[f(z)] = E_{p(\epsilon)}[f(g(\phi, \epsilon))]$$

2. Now, take the gradient w.r.t. ϕ:

$$\nabla_\phi L_{\text{recon}} = \nabla_\phi E_{p(\epsilon)}[f(g(\phi, \epsilon))]$$

3. Since $p(\epsilon)$ does not depend on ϕ, we move the gradient inside E

$$= E_{p(\epsilon)}\left[\nabla_\phi f(g(\phi, \epsilon))\right]$$

4. Apply the chain rule to the term inside:

$$=_{p(\epsilon)} \left[\frac{\partial f}{\partial} \cdot \frac{\partial z}{\partial}\right]$$

Frame Title

What this means: The Gradient Path is Unblocked!

The gradient $\frac{\partial f}{\partial z}$ can now flow back to μ_ϕ and σ_ϕ:

New Graph: (Encoder $\rightarrow \mu_\phi, \sigma_\phi) + (\epsilon \sim \mathcal{N}(0,1)) \rightarrow$

$z = \mu_\phi + \sigma_\phi \cdot \epsilon \quad \leftarrow$ Deterministic & Differentiable!

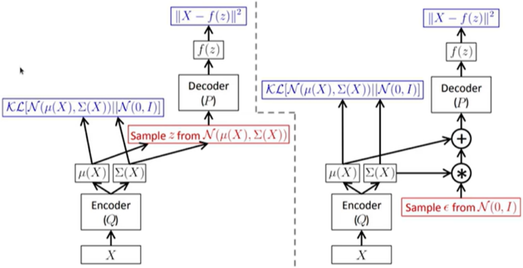
$z \rightarrow$ (Decoder $\rightarrow f(z))$

The gradients are simple and well-defined:

$$\frac{\partial z}{\partial \mu_\phi} = 1 \qquad \frac{\partial z}{\partial \sigma_\phi} = \epsilon$$

Backpropagation is now fully enabled.

Why Reparameterization



ref. 3

What is Functional Data?

In Functional Data Analysis (FDA), each observation X_i is not a single number or a vector, but a **function**, $X_i(t)$.

- We analyze a sample of n curves (or functions):

$$X_1(t), X_2(t), \ldots, X_n(t)$$

- The function is observed over a continuous domain, such as time $t \in [a, b]$.
- In practice, we observe discrete, often noisy points: $Y_i(t_j)$. FDA tools help us treat them as the smooth functions they represent.

Data Type	Single Observation X_i	Sample Data (n obs.)
Multivariate	Vector $x_i \in \mathbb{R}^p$	$n \times p$ Matrix
Functional	**Function $X_i(t)$**	**A set of n curves**

Why FDA? The Challenges

Question: Why not just treat $X_i(t)$ as a high-dimensional vector $X_i = [X_i(t_1), \ldots, X_i(t_m)]$?

- **The Curse of Dimensionality:** Often, the number of sampling points m is much larger than the number of samples n ($m \gg n$).
- **Smoothness:** Standard methods ignore that $X_i(t_j)$ and $X_i(t_{j+1})$ are highly correlated. FDA *exploits* this smoothness.
- **Irregularity:** Data can be **sparse** or **irregular**. The t_j points may be different for each X_i. A vector representation is impossible.

The Representation of Functional Data

Problem: We have discrete points $Y_i(t_j)$, but we want to analyze the smooth function $X_i(t)$.
We approximate $X_i(t)$ as a weighted sum (linear combination) of K known basis functions $\psi_k(t)$.

$$X_i(t) \approx \sum_{k=1}^{K} c_{ik}\psi_k(t)$$

- $\psi_k(t)$: A set of pre-defined basis functions (e.g., B-Splines, Fourier basis).
- c_{ik}: The coefficient (or "score") for the k-th basis on the i-th sample.

Functional PCA

Define:

$$\text{mean:}\bar{x}(t) = \frac{1}{n}\sum_{i=1}^{n} x_i(t)$$

$$\text{covariance:}\frac{1}{n}\sum_{i=1}^{n}(x_i(s) - \bar{x}(s))(x_i(t) - \bar{x}(t))$$

$$\text{Correlation:}\frac{\frac{1}{n}\sum_{i=1}^{n}(x_i(s) - \bar{x}(s))(x_i(t) - \bar{x}(t))}{\sqrt{\sum_{i=1}^{n}(x_i(s) - \bar{x}(s))^2}\sqrt{\sum_{i=1}^{n}(x_i(t) - \bar{x}(t))^2}}$$

1. Top K FPCs $w_1(t), \ldots, w_K(t)$
2. $X_i(t)$ is projected to $s_{i1}, \ldots, s_{iK}$: $X_i(t) = \sum_{i=1}^{K} s_{ik} w_k(t)$
3. Top FPCs are represented by basis functions. (e.g. Fourier, B-spline)

FPCA

1. $x(t) = (x_1(t), x_2(t), \ldots, x_n(t))^T$, (centered)
2. $\phi(t) = (\phi_1(t), \ldots, \phi_J(t))$
3. Centered curves $x(t) = \mathbf{C}\phi(t)$
4. $\sigma(s, t) = n^{-1}x(t)x^T(t) = n^{-1}\phi^T(s)\mathbf{C}^T\mathbf{C}\phi(t)$
5. $w(t) = \phi^T(t)\mathbf{b}$
6. maximize:
$$\int \sigma(s, t)w(t)dt = \rho w(s)$$
$$\text{subject to} \int [w(t)]^2 dt = \mathbf{b}^T \int \phi(t)^T\phi(t)dt\mathbf{b} = 1$$
7. $\mathbf{W} = \int \phi(t)\phi(t)^T dt$

FPCA Derivation via Basis Expansion

- Centered curves: $x(t) = C\phi(t)$
- Covariance function: $\sigma(s,t) = n^{-1}x(s)^T x(t) = n^{-1}\phi(s)^T C^T C\phi(t)$
- Eigenfunction: $w(t) = \phi(t)^T b$
- Gram matrix: $W = \int \phi(t)\phi(t)^T dt$

Want to maximize (solve the integral equation):

$$\int \sigma(s,t)w(t)dt = \rho w(s)$$

$$\int n^{-1}\phi(s)^T C^T C\phi(t)\phi(t)^T b dt = \rho\phi(s)^T b$$

$$n^{-1}\phi(s)^T C^T C\left(\int \phi(t)\phi(t)^T dt\right) b = \rho\phi(s)^T b$$

$$n^{-1}C^T CWb = \rho b; \quad \text{s.t. } b^T Wb = 1$$