Aprendizado Profundo

Autoencoders e *Variational Autoencoders*

Lucas Silveira Kupssinskü

Machine Learning Theory and Applications Laboratory
PUCRS

agosto de 2026

Visão Geral

1. Autoencoders
2. Variational Autoencoders
3. Treinamento e ELBO
4. Reparametrization Trick
5. VAE em ação
6. Panorama

Visão Geral

Autoencoders

Um *autoencoder* é uma rede neural treinada na tarefa de **reconstrução da própria entrada**.

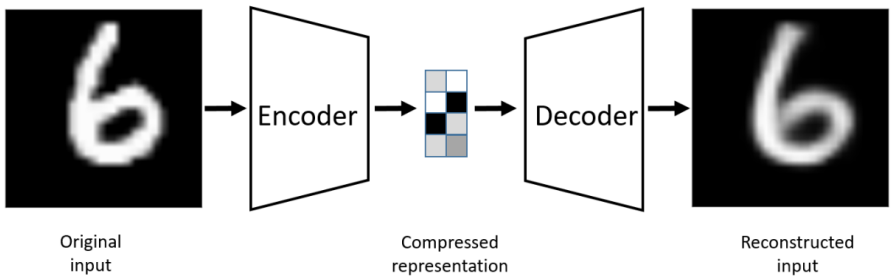
- Uma das primeiras formas de aprendizado **auto-supervisionado** / não supervisionado.
- A ideia é criar uma **representação útil** das instâncias de interesse na etapa intermediária (gargalo).

Objetivo

$$\underset{\boldsymbol{\theta},\phi}{\operatorname{argmin}} J(\mathbf{X},\ f(g(\mathbf{X},\phi),\boldsymbol{\theta}))$$

onde g é o *encoder* (parâmetros ϕ) e f é o *decoder* (parâmetros $\boldsymbol{\theta}$).

Autoencoders – Pipeline



- *Encoder* g reduz a entrada a uma representação compacta (o **latente**).
- *Decoder* f tenta reconstruir a entrada original a partir do latente.

Autoencoders – Regularização

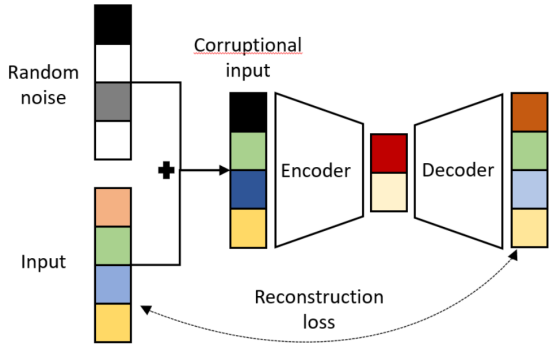
- É importante que o *autoencoder* tenha **regularização** para evitar soluções triviais.
- Se **não houver um gargalo**, o *autoencoder* pode simplesmente copiar a imagem.
- Se houver **unidades suficientes** no vetor latente, é possível indexar as imagens originais (memorização).

Formas comuns de regularização

- **Bottleneck**: dimensão latente $\ll$ dimensão da entrada.
- **Ruído** na entrada (denoising AE).
- **Esparsidade** na ativação do latente.

Denoising Autoencoders[1]

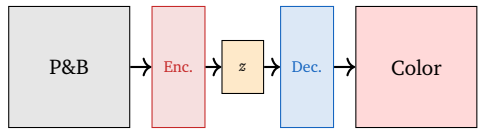
- Alguns *autoencoders* são treinados com o objetivo de **remover ruído** das instâncias de entrada.
- Tipicamente o *noise* segue uma:
  - distribuição **normal** (ruído aditivo);
  - distribuição de **Bernoulli** (ruído multiplicativo que invalida algumas *features*).
- A reconstrução é avaliada contra a entrada original (sem ruído).



[1] Pascal Vincent et al. "Extracting and Composing Robust Features with Denoising Autoencoders". Em: ICML. 2008, pp. 1096–1103.

Aplicação: Colorização

- **Entrada**: imagem P&B (1 canal). **Saída**: a mesma imagem em cores (3 canais RGB).
- *Encoder* comprime a cena em uma representação semântica; *decoder* gera as cores plausíveis.
- Perda: MSE no espaço de cor (RGB, ou *Lab* comparando apenas os canais a, b).
- Variação comum: *U-Net* convolucional com *skip connections* para preservar bordas.



Esquema: 1 canal → representação → 3 canais.

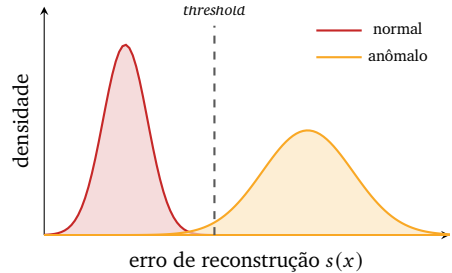
Problema **mal-posto**: várias colorações são plausíveis. MSE básico tende a cores desaturadas (média entre opções); variantes probabilísticas (VAE, GAN, difusão) geram cores mais vívidas.

Aplicação: Detecção de Anomalias

- Instâncias **normais** reconstroem bem (erro baixo); **anomalias** reconstroem mal (erro alto).
- Score de anomalia:

$$s(x) = \|x - f(g(x, \phi), \boldsymbol{\theta})\|^2$$

- Um *threshold* sobre $s(x)$ separa as duas classes.
- Casos típicos: fraude em transações, defeitos industriais (*visual inspection*), intrusão em redes, monitoramento de equipamentos.



Por que funciona?

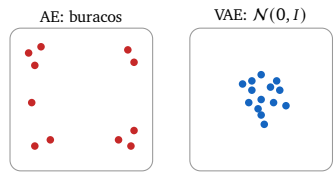
O AE só aprendeu a distribuição dos dados de treino, ele não tem como reconstruir o que nunca viu.

Visão Geral

Por que o *autoencoder* comum não gera?

Para **gerar** bastaria amostrar um código z e decodificá-lo. No AE determinístico isso **falha**:

- O encoder envia cada x a um ponto z, mas **não controla a distribuição** dos códigos.
- O latente fica com **buracos** (regiões sem dado) e aglomerados irregulares.
- Sem uma $p(z)$ conhecida de onde amostrar, um z aleatório cai num buraco → decoder gera **lixo**.



A ideia do VAE

Forçar o encoder a mapear os dados a uma distribuição **contínua e conhecida** (gaussiana): então $z \sim \mathcal{N}(0, I)$ decodificado gera dados novos plausíveis.

Variational Autoencoders[2]

- São modelos geradores **probabilísticos** cujo objetivo é aprender uma distribuição $p(x)$ sobre os dados.
- É um **modelo não linear de variável latente**.
  - Depois do treinamento podemos usar apenas o decoder (amostragem) ou apenas o encoder (inferência).

Panorama

O VAE é a **fusão** de duas ideias:

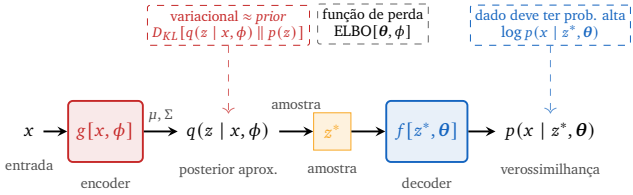
- *Autoencoder*: arquitetura encoder/decoder.
- **Inferência variacional**: aproximar a verossimilhança intratável via um limite inferior (ELBO).

Próximo slide: arquitetura completa, para servir de mapa ao longo da matemática.

[2] Diederik P. Kingma e Max Welling. "Auto-Encoding Variational Bayes". Em: ICLR. 2014.

Arquitetura VAE – Panorama

- *Encoder* $g[x, \phi]$ recebe x e produz $\mu(x)$, $\Sigma(x)$.
- Definimos $q(z \mid x, \phi) = \mathcal{N}(\mu, \Sigma)$.
- Amostramos $z^* \sim q(z \mid x, \phi)$.
- *Decoder* $f[z^*, \theta]$ gera os parâmetros de $p(x \mid z^*, \theta)$.
- Perda: $-$ELBO = reconstrução + KL.



Adaptado de Prince (2023), Fig. 17.9.

Use como mapa

Toda a matemática a seguir refere-se a esses mesmos blocos.

Modelos de variáveis latentes

- Abordagem **indireta** para modelar uma distribuição $p(x)$ sobre uma variável multidimensional x.
- Marginalização de uma probabilidade conjunta $p(x, z)$:

$$p(x) = \int p(x, z)\, dz = \int p(x \mid z)\, p(z)\, dz$$

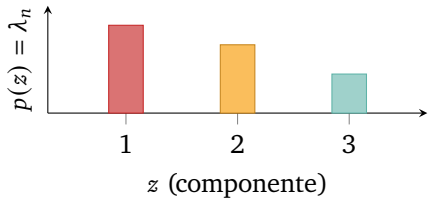
Motivação

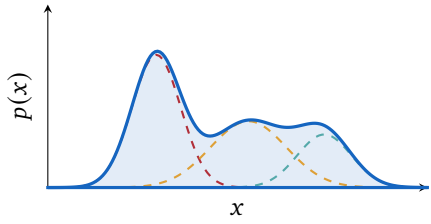
Distribuições relativamente simples $p(x \mid z)$ e $p(z)$ podem, por marginalização, definir distribuições $p(x)$ **complexas**.

Exemplo: Mistura de Gaussianas 1D

- z é uma variável latente **discreta**, com distribuição a priori categórica: $p(z = n) = \lambda_n$.
- A verossimilhança segue uma normal: $p(x \mid z = n) = \mathcal{N}(\mu_n, \sigma_n^2)$.
- A distribuição marginal:

$$p(x) = \sum_n \lambda_n \, \mathcal{N}(\mu_n, \sigma_n^2).$$





O *prior* discreto $p(z)$ pesa cada componente, e a marginal $p(x)$ é a soma ponderada das gaussianas (tracejadas).

Modelos não lineares de variáveis latentes

Tanto z quanto x são **contínuos** e **multivariados**.

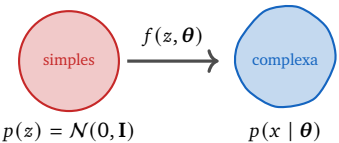
- *Prior* normal padrão: $p(z) = \mathcal{N}_z(0, \mathbf{I})$.
- Verossimilhança gaussiana com média não linear $f(z, \boldsymbol{\theta})$ e covariância esférica:

$$p(x \mid z, \boldsymbol{\theta}) = \mathcal{N}_x\Big(f(z, \boldsymbol{\theta}), \sigma^2 \mathbf{I}\Big).$$

- Marginal:

$$p(x \mid \boldsymbol{\theta}) = \int p(x \mid z, \boldsymbol{\theta})\, p(z)\, dz.$$

- Integrando gaussiano, mas f não linear $\Rightarrow$ integral **sem forma fechada**.



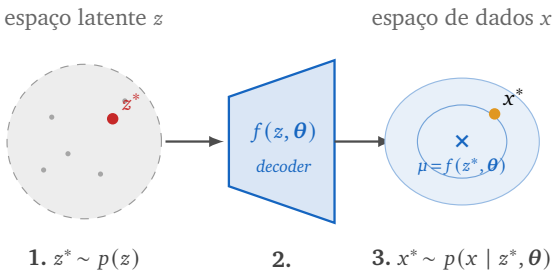
Função f não linear transforma uma gaussiana simples em uma distribuição arbitrariamente complexa.

Gerando novas instâncias x^*

Usamos **amostragem ancestral**:

1. Amostrar um latente z^* de $p(z)$ (*prior*).
2. Passar z^* pela rede $f(z^*, \boldsymbol{\theta})$ para obter a média da verossimilhança $p(x \mid z^*, \boldsymbol{\theta})$.
3. Amostrar x^* de $p(x \mid z^*, \boldsymbol{\theta})$.

A rede $f(z, \boldsymbol{\theta})$ é exatamente o *decoder* da arquitetura.



Amostramos z^* do *prior*, decodificamos com $f(z, \boldsymbol{\theta})$ e amostramos x^* da verossimilhança.
Na prática, costuma-se usar a média $f(z^*, \boldsymbol{\theta})$ diretamente, sem somar o ruído $\sigma^2\mathbf{I}$.

Visão Geral

Treinamento – O problema

Maximizar a log-verossimilhança direta é **inviável**:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left[\sum_{i=1}^{N} \int \mathcal{N}_{x^{(i)}} \left(f(z, \boldsymbol{\theta}), \sigma^2 \mathbf{I} \right) \mathcal{N}_z(0, \mathbf{I}) \, dz \right]$$

- Integral **sem forma fechada**, pois f é não linear.
- Monte Carlo direto $p(x) \approx \frac{1}{K} \sum_k p(x \mid z_k)$ com $z_k \sim p(z)$ também **falha**: para um x específico, z amostrado do prior quase nunca explica x, então $p(x \mid z)$ é minúsculo na enorme maioria das amostras.

Estratégia

Maximizamos um **limite inferior** da log-verossimilhança, usando uma proposta inteligente para amostrar z.

Por que precisamos de um *encoder*?

Ideia: ao invés de amostrar z do prior, sortear z de uma distribuição que concentra massa onde z explica x.

- Essa distribuição ideal é a posterior $p(z \mid x)$, mas ela também é intratável.
- **Solução**: aproximamos com uma proposta parametrizada

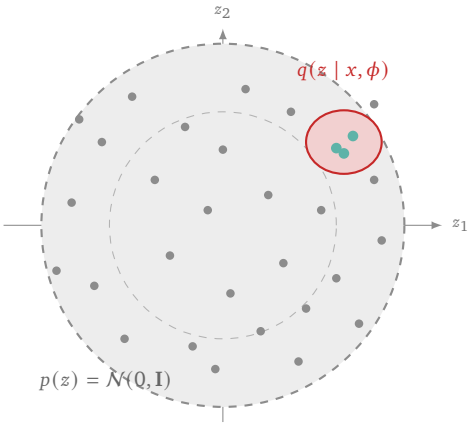
$$q(z \mid x, \phi) \approx p(z \mid x),$$

  implementada por uma **segunda rede** com parâmetros ϕ.
- Essa rede é o *encoder* do VAE: para cada x, produz uma distribuição sobre z.

O que ganhamos

Com q no lugar de $p(z)$, podemos derivar um **limite inferior** (ELBO) que é apertado quando q aproxima bem a posterior, vamos ver isso a seguir.

Amostrar do *prior* é ineficiente



- **Cinza**: amostras do *prior* $p(z)$. A massa fica espalhada por todo o espaço latente.
- **Verde**: as raras amostras que caem onde z realmente explica x, isto é, onde $p(x \mid z)$ é alto.
- Acertar essa região por sorteio do *prior* exige um número **enorme** de amostras: $p(z)$ é pouco informativo sobre um x específico.

Solução

A proposta $q(z \mid x, \phi)$ já **concentra** a massa nessa região.

Desigualdade de Jensen

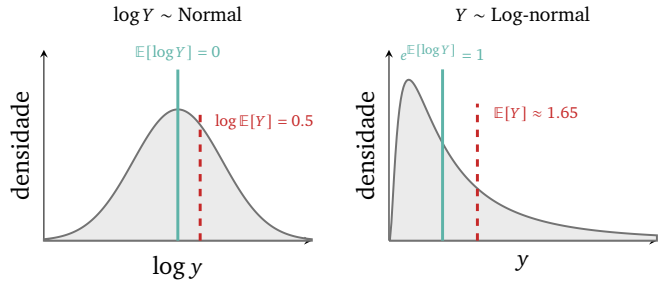
Para qualquer função **côncava** $g(\cdot)$:

$$g(\mathbb{E}[x]) \geq \mathbb{E}[g(x)]$$

Caso da função log (com $\log Y \sim \mathcal{N}(\mu, \sigma^2)$):

$$\log(\mathbb{E}[Y]) \geq \mathbb{E}[\log Y]$$

Aqui $\mu = 0$, $\sigma = 1$, então a folga $\sigma^2/2 = 0.5$.



A cauda da log-normal puxa a média aritmética $\mathbb{E}[Y]$ à direita da geométrica $e^{\mathbb{E}[\log Y]}$; logo $\log \mathbb{E}[Y] \geq \mathbb{E}[\log Y]$.

De onde vem o ELBO?

O nome **ELBO** (*Evidence Lower BOund*) é herdado de **inferência variacional**.

- **Evidência**: na linguagem bayesiana, $p(x \mid \boldsymbol{\theta}) = \int p(x, z \mid \boldsymbol{\theta})\, dz$ é a verossimilhança marginal, a "evidência" que os dados dão sobre o modelo.
- **Limite inferior**: como a integral é intratável, construímos uma função $\mathcal{L}[\boldsymbol{\theta}, \phi]$ tal que $\log p(x \mid \boldsymbol{\theta}) \geq \mathcal{L}[\boldsymbol{\theta}, \phi]$, e maximizamos $\mathcal{L}$ no lugar de $\log p(x)$.

Ingrediente que falta

Importance sampling: para qualquer q com suporte sobre z, $p(x \mid \boldsymbol{\theta}) = \mathbb{E}_{z \sim q}[p(x, z \mid \boldsymbol{\theta})/q(z)]$. O próximo slide combina esse passo com Jensen para obter o limite inferior.

Limite Inferior – Derivação

Reescrevemos a integral como uma esperança sob q e aplicamos Jensen na concavidade do log:

$$\begin{aligned}
\log p(x \mid \boldsymbol{\theta}) &= \log\left[\int p(x, z \mid \boldsymbol{\theta})\, dz\right] && \text{marginal} \\
&= \log\left[\int q(z)\, \frac{p(x, z \mid \boldsymbol{\theta})}{q(z)}\, dz\right] && \text{multiplica e divide por } q(z) \\
&= \log\left(\mathbb{E}_{z\sim q}\left[\frac{p(x, z \mid \boldsymbol{\theta})}{q(z)}\right]\right) && \text{integral ponderada por } q = \text{esperança} \\
&\geq \mathbb{E}_{z\sim q}\left[\log \frac{p(x, z \mid \boldsymbol{\theta})}{q(z)}\right] && \text{Jensen (log côncavo)} \\
&= \int q(z)\, \log\left(\frac{p(x, z \mid \boldsymbol{\theta})}{q(z)}\right) dz && \text{de volta à forma integral}
\end{aligned}$$

ELBO

O $q(z)$ externo é a medida da esperança, permanece como peso; apenas o $q(z)$ do denominador entra no log. Esse limite é o *evidence lower bound*; na prática q é parametrizado: $q(z \mid x, \phi)$.

Limite Inferior – Conexão com a posterior

A partir daqui escrevemos q como $q(z \mid x, \phi)$ (o encoder condiciona em x). Aplicando a regra do produto $p(x, z \mid \boldsymbol{\theta}) = p(z \mid x, \boldsymbol{\theta})\, p(x \mid \boldsymbol{\theta})$ no ELBO:

$$
\begin{aligned}
\mathrm{ELBO}[\boldsymbol{\theta}, \phi] &= \mathbb{E}_q\left[\log \tfrac{p(x,z|\boldsymbol{\theta})}{q(z|x,\phi)}\right] && \text{definição} \\
&= \mathbb{E}_q\left[\log \tfrac{p(z|x,\boldsymbol{\theta})\, p(x|\boldsymbol{\theta})}{q(z|x,\phi)}\right] && \text{regra do produto} \\
&= \mathbb{E}_q[\log p(x \mid \boldsymbol{\theta})] + \mathbb{E}_q\left[\log \tfrac{p(z|x,\boldsymbol{\theta})}{q(z|x,\phi)}\right] && \text{separa o log} \\
&= \log p(x \mid \boldsymbol{\theta}) - \mathbb{E}_q\left[\log \tfrac{q(z|x,\phi)}{p(z|x,\boldsymbol{\theta})}\right] && \log p(x) \text{ sai da } \mathbb{E}\text{; troca o sinal} \\
&= \log p(x \mid \boldsymbol{\theta}) - D_{KL}[q(z \mid x, \phi) \,\|\, p(z \mid x, \boldsymbol{\theta})] && \text{definição de KL}
\end{aligned}
$$

Rearranjando

$\log p(x \mid \boldsymbol{\theta}) = \mathrm{ELBO}[\boldsymbol{\theta}, \phi] + D_{KL}[q(z \mid x, \phi) \,\|\, p(z \mid x, \boldsymbol{\theta})]$, a identidade exata do próximo slide.

ELBO – A identidade mestre

Da derivação anterior, chega-se à **identidade exata**:

$$\log p(x \mid \boldsymbol{\theta}) = \underbrace{\int q(z \mid x, \boldsymbol{\phi}) \, \log \frac{p(x, z \mid \boldsymbol{\theta})}{q(z \mid x, \boldsymbol{\phi})} \, dz}_{\mathrm{ELBO}[\boldsymbol{\theta}, \boldsymbol{\phi}]} + \underbrace{D_{KL}[q(z \mid x, \boldsymbol{\phi}) \,\|\, p(z \mid x, \boldsymbol{\theta})]}_{\geq 0}$$

- O gap entre $\log p(x)$ e o ELBO é **exatamente** a KL entre q e a posterior verdadeira.
- Bound **apertado** $\Leftrightarrow q(z \mid x, \boldsymbol{\phi}) = p(z \mid x, \boldsymbol{\theta})$.
- Maximizar o ELBO faz **duas coisas ao mesmo tempo**: sobe $\log p(x)$ (aprende o modelo) e aperta q contra a posterior (aprende o encoder).

ELBO – Reorganizando

$$\text{ELBO}[\boldsymbol{\theta}, \boldsymbol{\phi}] = \int q(z \mid x, \boldsymbol{\phi}) \log\left(\frac{p(x, z \mid \boldsymbol{\theta})}{q(z \mid x, \boldsymbol{\phi})}\right) dz$$

Aplicando $p(x, z) = p(x \mid z)\, p(z)$ e separando a razão:

$$= \int q(z \mid x, \boldsymbol{\phi}) \log\left(\frac{p(x \mid z, \boldsymbol{\theta})\, p(z)}{q(z \mid x, \boldsymbol{\phi})}\right) dz$$

$$= \int q(z \mid x, \boldsymbol{\phi}) \log p(x \mid z, \boldsymbol{\theta})\, dz - D_{KL}[q(z \mid x, \boldsymbol{\phi}) \,\|\, p(z)]$$

ELBO – Interpretação

$$\text{ELBO}[\boldsymbol{\theta}, \boldsymbol{\phi}] = \underbrace{\int q(z \mid x, \boldsymbol{\phi}) \log p(x \mid z, \boldsymbol{\theta})\, dz}_{\text{Reconstruction Loss}} \underbrace{- D_{KL}[q(z \mid x, \boldsymbol{\phi}) \,\|\, p(z)]}_{\text{Regularização do latente}}$$

- **Primeiro termo**: log-verossimilhança esperada da reconstrução, sob z amostrados do encoder. Também chamado de *reconstruction loss*.
- **Segundo termo**: avalia quão perto a distribuição auxiliar $q(z \mid x, \phi)$ está do *prior* $p(z)$.

Sinal

O ELBO é **maximizado** (é um limite inferior da log-verossimilhança); a função de custo treinada é −ELBO, minimizada.

Termo de reconstrução = MSE (caso gaussiano)

Para $p(x \mid z, \boldsymbol{\theta}) = \mathcal{N}(f(z, \boldsymbol{\theta}),\, \sigma^2 \mathbf{I})$:

$$\begin{aligned} \log p(x \mid z, \boldsymbol{\theta}) &= -\tfrac{D_x}{2} \log(2\pi\sigma^2) - \tfrac{1}{2\sigma^2} \|x - f(z, \boldsymbol{\theta})\|^2 \\ &= -\tfrac{1}{2\sigma^2} \|x - f(z, \boldsymbol{\theta})\|^2 + \text{const} \end{aligned}$$

Ponte com o AE clássico

Maximizar $\log p(x \mid z, \boldsymbol{\theta})$ **é exatamente** minimizar o erro quadrático $\|x - f(z, \boldsymbol{\theta})\|^2$. O ELBO se decompõe em

$$-\text{ELBO} = \underbrace{\tfrac{1}{2\sigma^2} \mathbb{E}_q \left[\|x - f(z, \boldsymbol{\theta})\|^2 \right]}_{\text{reconstrução (MSE)}} + \underbrace{D_{KL}[q(z \mid x, \phi) \,\|\, p(z)]}_{\text{regularização}} + \text{const.}$$

KL entre gaussianas (1/2) – Partindo da definição

A KL entre duas distribuições quaisquer é:

$$D_{KL}[q \,\|\, p] = \int q(z) \, \log\left(\frac{q(z)}{p(z)}\right) dz = \mathbb{E}_q[\log q(z)] - \mathbb{E}_q[\log p(z)]$$

KL entre gaussianas (1/2) – Partindo da definição

A KL entre duas distribuições quaisquer é:

$$D_{KL}[q \,\|\, p] = \int q(z) \, \log\left(\frac{q(z)}{p(z)}\right) dz = \mathbb{E}_q[\log q(z)] - \mathbb{E}_q[\log p(z)]$$

Para $q(z|x, \phi) = \mathcal{N}(\mu, \Sigma)$ e $p(z) = \mathcal{N}(\mathbf{0}, \mathbf{I})$, o log de cada densidade gaussiana é uma forma quadrática:

$$\log p(z) = -\tfrac{D_z}{2} \log(2\pi) - \tfrac{1}{2} z^\top z$$

$$\log q(z) = -\tfrac{D_z}{2} \log(2\pi) - \tfrac{1}{2} \log|\Sigma| - \tfrac{1}{2}(z-\mu)^\top \Sigma^{-1}(z-\mu)$$

Os termos $-\frac{D_z}{2}\log(2\pi)$ cancelam. No próximo slide usamos a identidade $\mathbb{E}_q[zz^\top] = \Sigma + \mu\mu^\top$ (definição de covariância: $\Sigma = \mathbb{E}_q[zz^\top] - \mu\mu^\top$).

KL entre gaussianas (2/2) – Forma fechada

Tomando esperanças sob q (usando $\mathbb{E}_q[z] = \mu$ e $\mathbb{E}_q[zz^\top] = \Sigma + \mu\mu^\top$):

$$\mathbb{E}_q\left[-\tfrac{1}{2} z^\top z\right] = -\tfrac{1}{2}\left(\mathrm{Tr}[\Sigma] + \mu^\top \mu\right)$$

$$\mathbb{E}_q\left[-\tfrac{1}{2}(z-\mu)^\top \Sigma^{-1}(z-\mu)\right] = -\tfrac{1}{2}\, D_z$$

Substituindo na definição da KL:

$$D_{KL}[\mathcal{N}(\mu,\Sigma) \,\|\, \mathcal{N}(\mathbf{0},\mathbf{I})] = \tfrac{1}{2}\Bigg(\underbrace{\mathrm{Tr}[\Sigma]}_{\text{dispersão}} + \underbrace{\mu^\top \mu}_{\text{desvio da origem}} - \underbrace{D_z}_{\text{norm}} - \underbrace{\log\det\Sigma}_{\text{volume}} \Bigg)$$

Exemplo Numérico: Calculando o Termo KL

Imagine um VAE treinado em MNIST com latente 2D. Para a imagem de um "7" específico, o encoder produz uma distribuição $q(z \mid x, \phi) = \mathcal{N}(\mu, \Sigma)$ sobre o latente, compacta o suficiente para "codificar 7", mas com alguma incerteza para suavizar o espaço. O termo KL mede quão longe essa distribuição está do prior $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

Encoder produz $\mu = (0.5, -0.8)$ e $\Sigma = \text{diag}(0.25, 1.5)$ (latente 2D, $D_z = 2$).

Calculando cada termo:

$$\begin{aligned} \text{Tr}[\Sigma] &= 0.25 + 1.5 = 1.75 \\ \mu^\top \mu &= 0.25 + 0.64 = 0.89 \\ \log\det\Sigma &= \log(0.25) + \log(1.5) \\ &\approx -1.386 + 0.405 \\ &= -0.981 \\ D_z &= 2 \end{aligned}$$

Substituindo

$$\begin{aligned} D_{KL} &= \tfrac{1}{2}\big(1.75 + 0.89 - 2 + 0.981\big) \\ &= \tfrac{1}{2}\,(1.621) \approx \mathbf{0.81} \end{aligned}$$

Os dois termos que dominam:
$\mu^\top \mu = 0.89$ (desvio da origem) e
$-\log\det\Sigma = +0.98$ (variância pequena $\sigma^2 = 0.25$ é penalizada para evitar colapso).

VAE – Aproximação por *Monte Carlo*

O primeiro termo do ELBO ainda é uma integral intratável. Porém, como é uma **esperança**, podemos aproximá-lo por *Monte Carlo*:

$$\int q(z \mid x, \phi)\, \log p(x \mid z, \boldsymbol{\theta})\, dz \approx \log p(x \mid z^*, \boldsymbol{\theta}), \quad z^* \sim q(z \mid x, \phi)$$

ELBO aproximado

$$\text{ELBO}[\boldsymbol{\theta}, \phi] \approx \log p(x \mid z^*, \boldsymbol{\theta}) - D_{KL}[q(z \mid x, \phi) \,\|\, p(z)]$$

Visão Geral

Reparametrization Trick

- Como temos um **termo estocástico** no meio da rede neural (a amostragem $z^* \sim q(z \mid x, \phi)$), é inviável computar *backpropagation* diretamente.
- **Solução**: reparametrizar a saída do *encoder* de modo a tirar a amostragem do caminho do gradiente:

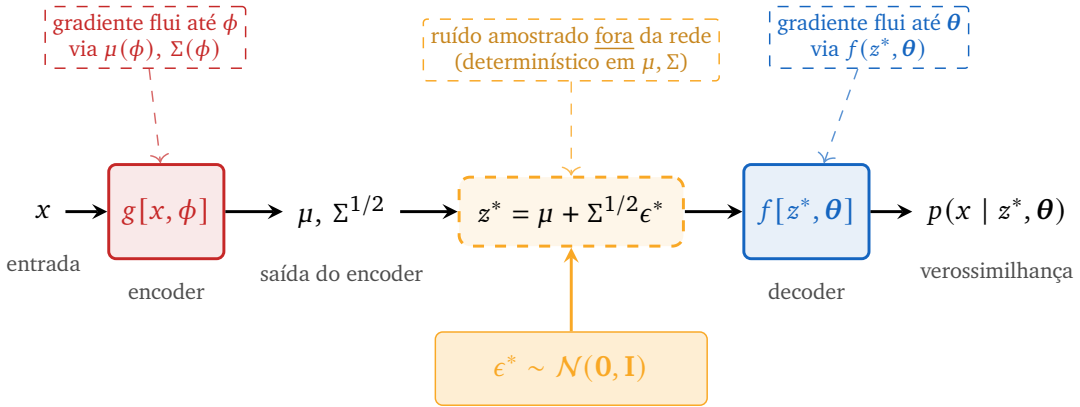
$$z^* = \mu + \Sigma^{1/2} \epsilon^*, \qquad \epsilon^* \sim \mathcal{N}(0, \mathbf{I})$$

Para Σ diagonal (caso usual), $\Sigma^{1/2} \epsilon^* = \sigma \odot \epsilon^*$ (forma elemento a elemento).

Ideia chave

O ruído ϵ^* é amostrado fora da rede; μ e Σ permanecem no caminho diferenciável.

Reparametrization Trick – Diagrama



Adaptado de Prince (2023), Fig. 17.11.

Visão Geral

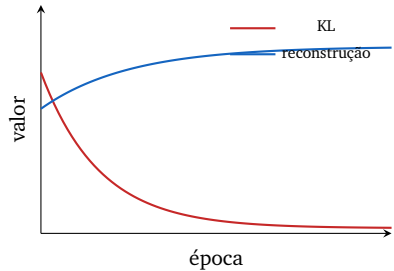
VAE em ação – Amostras e interpolação

Amostragem: $z \sim p(z)$, depois $x = f(z, \boldsymbol{\theta})$. Resultado reconhecível mas borrado (limite inferior não é apertado).

Interpolação: transição suave entre dois dígitos confirma que o latente é contínuo e semanticamente organizado.

Patologia: *posterior collapse*

- Quando o decoder f é **poderoso demais**, ele consegue modelar x ignorando z.
- O termo KL pressiona $q(z \mid x, \phi) \to p(z)$: encoder colapsa para o prior. Latente vira inútil.
- Sintomas durante o treino:
  - $D_{KL} \to 0$;
  - $\mu(x) \to 0$, $\Sigma(x) \to \mathbf{I}$ para todo x.
- Mitigações:
  - **KL annealing**: rampa do peso do KL de 0 a 1;
  - **Free bits**: troca D_{KL} por $\sum_i \max(\lambda, D_{KL,i})$. Cada dimensão tem λ nats "grátis" antes do KL pressionar;
  - reduzir capacidade do decoder.

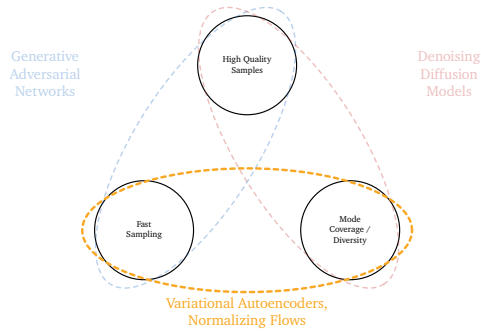


KL cai a zero enquanto a reconstrução estagna: o decoder ignora z.

Visão Geral

Onde VAEs ficam no trilema?

- **Amostragem rápida**: ✓ (1 forward do decoder).
- **Cobertura / diversidade**: ✓ (maximização do ELBO cobre toda a distribuição).
- **Qualidade visual**: × (imagens tipicamente borradas; o limite inferior não é apertado).



Posicionamento

VAEs ocupam o canto velocidade + cobertura do trilema, ao lado dos *Normalizing Flows*. Para qualidade, combine VAEs com modelos difusores, base da arquitetura *Latent Diffusion*.

Resumo

- *Autoencoders*: encoder g e decoder f reconstruindo a entrada; latente útil para representação.
- *Denoising AE*: entrada corrompida; rede aprende a remover ruído.
- Aplicação de AE: detecção de anomalias via erro de reconstrução.
- *VAE*: *autoencoder* probabilístico, latente é uma distribuição $q(z \mid x, \phi)$.
- Identidade mestre: $\log p(x) = \text{ELBO} + D_{KL}[q \| p(z \mid x)]$; bound apertado quando q aproxima a posterior.
- ELBO = reconstrução $-$ $\text{KL}[q(z \mid x, \phi) \| p(z)]$ (logo $-$ELBO = reconstrução + KL); para $p(x \mid z)$ gaussiana, reconstrução = MSE.
- *Reparametrization trick* permite backprop através da amostragem.
- *Posterior collapse*: $\text{KL} \to 0$, latente vira inútil; mitigado com KL annealing ou free bits.

Leituras Recomendadas

- Capítulo 17: Simon J. D. Prince. Understanding Deep Learning. MIT Press, 2023
- Diederik P. Kingma e Max Welling. “Auto-Encoding Variational Bayes”. Em: ICLR. 2014
- Pascal Vincent et al. “Extracting and Composing Robust Features with Denoising Autoencoders”. Em: ICML. 2008, pp. 1096–1103
- Irina Higgins et al. “β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework”. Em: ICLR. 2017

Referências I

[1] Irina Higgins et al. “β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework”. Em: ICLR. 2017.

[2] Diederik P. Kingma e Max Welling. “Auto-Encoding Variational Bayes”. Em: ICLR. 2014.

[3] Simon J. D. Prince. Understanding Deep Learning. MIT Press, 2023.

[4] Pascal Vincent et al. “Extracting and Composing Robust Features with Denoising Autoencoders”. Em: ICML. 2008, pp. 1096–1103.