

# Autoencoders: da motivação às variantes modernas

CPE 727 - Aprendizado de Profundo

Felipe Fink Grael, Rafael Tadeu Cardoso dos Santos, Thalles Nonato Leal  
Santos e Jefferson Osowsky

25 de novembro de 2025

# Table of Contents

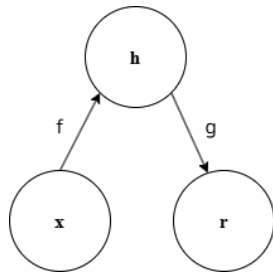
## 1 Motivação

- ▶ Motivação
- ▶ Formulação Matemática
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ Considerações sobre Arquitetura
- ▶ Autoencoders com regularização
- ▶ Encoders and Decoders Estocásticos
- ▶ Aplicações
- ▶ Referências Bibliográficas
- ▶ Anexos

# Introdução

## 1 Motivação

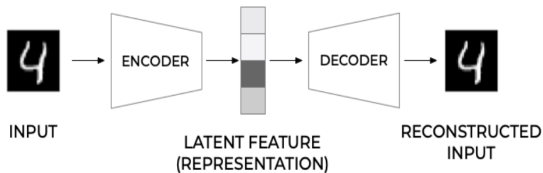
- **Definição:** Algoritmos cujo propósito principal é copiar sua entrada na saída [1]. São tipicamente construídos como redes neurais artificiais treinadas de forma não supervisionada.
- **Arquitetura Básica:**
  - Encoder: transforma entrada em representação latente
  - Representação: espaço latente de menor, maior ou igual dimensão
  - Decoder: reconstrói a entrada original
- **Objetivo:** Aprender a função identidade  $f(x) \approx x$  através de um espaço latente



**Figura:** Estrutura básica de Autoencoders.

# Componentes Fundamentais

## 1 Motivação



**Figura:** Estrutura de um autoencoder com representação latente<sup>1</sup>.

- **Encoder:**  $f_{\theta} : \mathcal{X} \rightarrow \mathcal{H}$  onde  $h = f_{\theta}(x)$
- **Decoder:**  $g_{\phi} : \mathcal{H} \rightarrow \mathcal{X}$  onde  $x' = g_{\phi}(h)$
- **Reconstrução:**  $x' = g_{\phi}(f_{\theta}(x))$
- **Espaço latente  $\mathcal{H}$ :** Representação comprimida dos dados ( $\dim(\mathcal{H}) < \dim(\mathcal{X})$ )

<sup>1</sup>Figura de Umberto Michelucci [2]

# Por que usar Autoencoders?

## 1 Motivação

### Aprendizado de Representações:

- Extrair características relevantes automaticamente dos dados
- Redução de dimensionalidade não-linear (superior ao PCA para dados complexos)
- Aprendizado não supervisionado - não requer labels

### Vantagens sobre métodos tradicionais:

- PCA: apenas transformações lineares
- Autoencoders: capturam relações não-lineares complexas
- Profundidade permite representações hierárquicas [3]

# Table of Contents

## 2 Formulação Matemática

- ▶ Motivação
- ▶ **Formulação Matemática**
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ Considerações sobre Arquitetura
- ▶ Autoencoders com regularização
- ▶ Encoders and Decoders Estocásticos
- ▶ Aplicações
- ▶ Referências Bibliográficas
- ▶ Anexos

## Definição Formal

### 2 Formulação Matemática

Seja  $\mu_{ref}$  uma distribuição de probabilidade de referência em  $\mathcal{X}$  e  $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  uma função de distância.

Função de custo do autoencoder:

$$L(\theta, \phi) = \mathbb{E}_{x \sim \mu_{ref}} [d(x, g_{\phi}(f_{\theta}(x)))]$$

Objetivo de treinamento:

$$(\theta^*, \phi^*) = \arg \min_{\theta, \phi} L(\theta, \phi)$$

# Exemplo: Autoencoder Linear de Uma Camada

## 2 Formulação Matemática

**Encoder:**

$$h = f_{W,b}(x) = \sigma(Wx + b)$$

**Decoder:**

$$x' = g_{W',b'}(h) = \sigma(W'h + b')$$

**Função Custo:**

$$L(W, b, W', b') = \frac{1}{N} \sum_{i=1}^N \|x_i - g_{W',b'}(f_{W,b}(x_i))\|_2^2$$

Parâmetros a otimizar:  $\theta = W, b, \phi = W', b'$



# Exemplo: Undercomplete Autoencoder Linear (Caso Especial)

## 2 Formulação Matemática

**Autoencoder linear:** sem função de ativação  $\sigma(z) = z$

**Teorema:** O autoencoder linear ótimo projeta os dados no subespaço gerado pelos primeiros  $k$  autovetores da matriz de covariância  $\Sigma_{XX}$  [4].

Erro mínimo:

$$\Sigma(A, B) = \text{Tr}(\Sigma) - \sum_{i=1}^k \lambda_i = \sum_{i=k+1}^n \lambda_i$$

Onde  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$  são os autovalores de  $\Sigma_{XX}$ .

**Conexão com PCA:** Autoencoders lineares aprendem o mesmo subespaço que a Análise de Componentes Principais.

# Table of Contents

## 3 Undercomplete Autoencoders com redes neurais

- ▶ Motivação
- ▶ Formulação Matemática
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ Considerações sobre Arquitetura
- ▶ Autoencoders com regularização
- ▶ Encoders and Decoders Estocásticos
- ▶ Aplicações
- ▶ Referências Bibliográficas
- ▶ Anexos

# Undercomplete Autoencoder

## 3 Undercomplete Autoencoders com redes neurais

- Autoencoders são treinados para copiar a entrada para a saída, mas o objetivo real é aprender representações úteis dos dados.
- Para isso, impõe-se que a dimensão do *espaço latente* satisfaça  $\dim(h) < \dim(x)$ , formando um *autoencoder undercomplete*, o que força o modelo a capturar as características mais relevantes da distribuição dos dados.
- O treinamento consiste em minimizar a função de perda:

$$L(x, g(f(x))),$$

onde  $L$  mede quão diferente  $g(f(x))$  está de  $x$  (por exemplo, usando MSE).

# Undercomplete Autoencoders: Correspondência com PCA/SVD

## 3 Undercomplete Autoencoders com redes neurais

**PCA e Autoencoders:** Quando o decoder é linear e  $L$  é o erro quadrático médio (MSE), um autoencoder undercomplete aprende a abranger o mesmo subespaço que o PCA.

Isso ocorre porque, sob linearidade e codificação com dimensão reduzida, o autoencoder busca a melhor reconstrução no sentido dos mínimos quadrados, exatamente como o PCA.

## Formulação do Problema

### 3 Undercomplete Autoencoders com redes neurais

O objetivo de um autoencoder linear é resolver:

$$\min_{D,E} \frac{1}{2} \|X - D(E(X))\|_F^2.$$

Se escolhermos  $D$  e  $E$  como matrizes (mapeamentos lineares), o problema é:

$$\min_{D,E} \frac{1}{2} \|X - DEX\|_F^2.$$

Introduzimos uma variável auxiliar:

- $A = DE$
- $\min_{A,D,E} \frac{1}{2} \|X - AX\|_F^2$  sujeito a  $A = DE$ .
- A restrição  $A = DE$  é o que torna o problema não convexo.

# Relaxação: Ignorando a Restrição

3 Undercomplete Autoencoders com redes neurais

Ignorando por um momento a restrição  $A = DE$ :

- O problema vira mínimos quadrados clássico.
- Uma solução analítica está disponível.

A solução é:

$$A^* = (XX^T)^{-1}(XX^T).$$

Se  $(XX^T)$  é invertível:

$$A^* = I,$$

ou seja, o mapeamento ótimo (sem a restrição) seria a identidade.

## Usando a SVD de $XX^T$

### 3 Undercomplete Autoencoders com redes neurais

Como  $XX^T$  é quadrada, sua decomposição SVD é:

$$XX^T = USU^T,$$

onde  $U$  é ortogonal e  $S$  contém os autovalores.

Substituindo na solução:

$$A^* = (USU^T)^{-1}(USU^T)$$

$$= (U^T)^{-1}S^{-1}U^{-1} USU^T$$

$$= UU^T.$$

# Solução Analítica do Autoencoder Linear

## 3 Undercomplete Autoencoders com redes neurais

Reintroduzindo a restrição  $A = DE$  obtemos:

$$A^* = D^*E^* = UU^\top.$$

Portanto:

$$D^* = U, \quad E^* = U^\top.$$

- Se  $X$  tem posto  $n$ , somente as primeiras  $n$  colunas de  $U$  são necessárias.
- Assim, o autoencoder linear aprende o mesmo subespaço do PCA.



# Interpretação: Relação Explícita com PCA

## 3 Undercomplete Autoencoders com redes neurais

A matriz  $U$  contém os **autovetores** dos dados — exatamente as **direções principais** do PCA.

Portanto, a solução ótima do autoencoder linear é:

$$E^* = U^T \quad (\text{projeção para o espaço PCA})$$

$$D^* = U \quad (\text{reconstrução a partir das componentes principais})$$

- O **encoder** recupera as coordenadas PCA.
- O **decoder** reconstrói os dados a partir dessas coordenadas.

# Autoencoders Não Lineares

## 3 Undercomplete Autoencoders com redes neurais

- Autoencoders com **encoder e decoder não lineares** podem aprender uma generalização mais poderosa do PCA.
- **Risco:** se a capacidade do encoder/decoder for muito grande, o autoencoder pode apenas **copiar os dados**, sem extrair informações úteis.
- Embora não ocorra na prática, ilustra que um autoencoder treinado apenas para copiar dados **pode falhar em aprender informações úteis** se a capacidade for excessiva.

## Comparações com outros modelos: t-SNE

### 3 Undercomplete Autoencoders com redes neurais

- O t-Distributed Stochastic Neighbor Embedding (t-SNE) reduz dimensões mapeando as distâncias do espaço de alta dimensão para um espaço de baixa dimensão por meio de uma distribuição de probabilidade.
- Objetivo: garantir que pontos próximos no espaço original permaneçam próximos após a redução.
- Usado principalmente para **visualização em 2D ou 3D**.
- Útil para obter uma visão inicial da estrutura e posicionamento dos dados.

## t-SNE x Autoencoders

### 3 Undercomplete Autoencoders com redes neurais

- Autoencoders são mais versáteis e podem ser usados tanto para extração de características quanto para redução dimensional com número livre de dimensões.
- Ambos (t-SNE e autoencoders) capturam relações **não lineares** nos dados e produzem representações de baixa dimensão.
- Entretanto, o t-SNE é frequentemente **mais custoso computacionalmente**.
- t-SNE apresenta **pouca escalabilidade** para conjuntos de dados grandes.
- Autoencoders tendem a ser **mais rápidos** em tarefas comparáveis.

# Table of Contents

## 4 Considerações sobre Arquitetura

- ▶ Motivação
- ▶ Formulação Matemática
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ **Considerações sobre Arquitetura**
- ▶ Autoencoders com regularização
- ▶ Encoders and Decoders Estocásticos
- ▶ Aplicações
- ▶ Referências Bibliográficas
- ▶ Anexos

# Autoencoders neurais: Número de camadas

## 4 Considerações sobre Arquitetura

**Teorema do Aproximador Universal:** Redes neurais com uma única camada oculta podem aproximar qualquer função contínua.

**Autoencoders profundos:** Na prática, encoder e decoders possuem pelo menos uma camada oculta cada

- Maior capacidade de modelagem
- Podem aprender representações hierárquicas
- Podem ser treinados camada a camada (greedy layer-wise pretraining)

# Dimensão do Espaço Latente

## 4 Considerações sobre Arquitetura

**Subcompletos** (undercomplete): Espaço latente tem dimensão menor que a entrada ( $\dim(\mathcal{H}) < \dim(\mathcal{X})$ )

- Forçam compactação dos dados (com perdas)
- Encoders e decoders não podem ser bons demais

**Sobrecompletos** (overcomplete): Espaço latente pode ter dimensão maior que a entrada ( $\dim(\mathcal{H}) > \dim(\mathcal{X})$ )

- Risco de aprender a função identidade
- Usam **regularização** para conferir características desejáveis

# Table of Contents

## 5 Autoencoders com regularização

- ▶ Motivação
- ▶ Formulação Matemática
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ Considerações sobre Arquitetura
- ▶ **Autoencoders com regularização**
- ▶ Encoders and Decoders Estocásticos
- ▶ Aplicações
- ▶ Referências Bibliográficas
- ▶ Anexos



# Autoencoders com regularização

## 5 Autoencoders com regularização

Técnicas que adicionam **termos de regularização** ou modificam o processo de treinamento para melhorar a qualidade das representações aprendidas. São frequentemente **sobrecompletos**.

- **Autoencoders Esparsos:** Força esparsidade na representação latente
- **Denoising Autoencoders:** Perturba a entrada com ruído e reconstrói a entrada limpa
- **Contractive Autoencoders:** Penaliza o jacobiano do encoder em relação à entrada
- **Variational Autoencoders:** Trata o espaço latente como uma variável aleatória com distribuição aprendida

# Sparse Autoencoders

## 5 Autoencoders com regularização

**Objetivo:** Forçar a representação latente a ser esparsa, ou seja, a maioria dos neurônios na camada latente deve estar inativa (valores próximos de zero).

$$L(x) = L_{\text{reconstrução}}(x) + \Omega(x)$$

**Norma  $L_1$  do espaço latente:**

$$\Omega(x) = \alpha \sum_x \|f(x)\|_1$$

- Penaliza diretamente as ativações
- Mais fácil de implementar
- Promove esparsidade indiretamente

**Divergência KL**

$$\Omega(x) = \alpha \sum_j KL(\rho \| \hat{\rho}_j), \quad \hat{\rho}_j = \frac{1}{N} \sum_x f_j(x)$$

- Divergência entre ativações e Bernoulli
- Maior controle sobre esparsidade
- $\rho$ : taxa de ativação desejada (pequeno, ex: 0.05)
- $\hat{\rho}_j$ : ativação média do neurônio  $j$

# Sparse Autoencoders: Interpretabilidade

## 5 Autoencoders com regularização

Com poucos neurônios ativos, os sparse autoencoders tendem a aprender representações mais interpretáveis e robustas.



Figure 2: Sparse autoencoders (SAEs) trained on pre-trained ViT activations discover a wide spread of features across both visual patterns and semantic structures. We show eight different features from an SAE trained on ImageNet-1K activations from a CLIP-trained ViT-B/16.

Características aprendidas por um sparse autoencoder aplicado nas ativações da camada 11 de um modelo ViT-Base/16 (12 camadas). O SAE tem fator de expansão de 32 (dimensão 768 na entrada e saída, 24576 no espaço latente). [5].

# Table of Contents

## 6 Encoders and Decoders Estocásticos

- ▶ Motivação
- ▶ Formulação Matemática
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ Considerações sobre Arquitetura
- ▶ Autoencoders com regularização
- ▶ **Encoders and Decoders Estocásticos**
- ▶ Aplicações
- ▶ Referências Bibliográficas
- ▶ Anexos

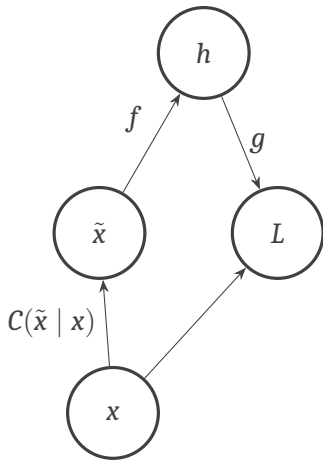
# Denoising Autoencoders (DAE)

## 6 Encoders and Decoders Estocásticos

- **Definição:** O *Denoising Autoencoder* (DAE) é um autoencoder que recebe uma entrada ruidosa e é treinado para reconstruir a versão limpa dessa entrada na saída [6].
- **Objetivo:** Minimizar a função custo:

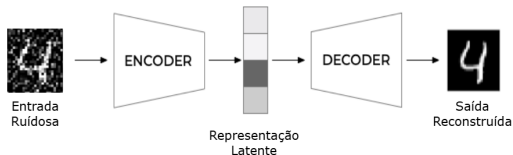
$$L(\mathbf{x}, g(f(\tilde{\mathbf{x}})))$$

- $\mathbf{x}$ : entrada original.
- $\tilde{\mathbf{x}}$ : entrada corrompida com ruído.
- $C(\tilde{\mathbf{x}} | \mathbf{x})$ : processo de corrupção que adiciona ruído à entrada.
- $f$ : função do encoder.
- $g$ : função do decoder.



# Motivação para DAE

## 6 Encoders and Decoders Estocásticos



- **Robustez a Ruído:** DAEs aprendem a extrair características robustas dos dados, ignorando variações irrelevantes causadas por ruído [7]. Isso é análogo a perspectiva humana de reconhecer objetos/formas/contextos quando os mesmos estão parcialmente ocultos ou corrompidos.
- **Melhoria na Generalização:** Ao aprender a reconstruir entradas limpas a partir de versões ruidosas, DAEs podem melhorar a capacidade de generalização do modelo.
- **Aprendizado de Representações Significativas:** DAEs incentivam o modelo a capturar estruturas subjacentes nos dados, levando a representações latentes mais informativas.

# Ruído de Entrada em DAE

## 6 Encoders and Decoders Estocásticos

Tipo de Ruído	Descrição
Ruído Gaussiano	$\tilde{x} = x + \epsilon, \epsilon \sim \mathcal{N}(0, \sigma^2 I)$
Masking Noise	$\tilde{x}_i = \begin{cases} 0 & \text{prob. } \nu \\ x_i & \text{prob. } 1 - \nu \end{cases}$
Salt-and-Pepper Noise	$\tilde{x}_i = \begin{cases} 0 & \text{prob. } \nu/2 \\ 1 & \text{prob. } \nu/2 \\ x_i & \text{prob. } 1 - \nu \end{cases}$

**Tabela:** Tipos comuns de ruído utilizados em DAE.

# Função Custo

## 6 Encoders and Decoders Estocásticos

A função custo geral para um DAE é dada por:

$$L_{DAE}(\theta, \phi) = \mathbb{E}_{x \sim \mu_{ref}, \tilde{x} \sim C(\tilde{x}|x)} [L(x, g_{\phi}(f_{\theta}(\tilde{x})))]$$

Onde:

- $x$ : entrada original amostrada de uma distribuição de referência  $\mu_{ref}$ .
- $\tilde{x}$ : entrada corrompida com ruído amostrada de uma distribuição condicional  $C(\tilde{x} | x)$ .
- $f_{\theta}$ : função do encoder parametrizada por  $\theta$ .
- $g_{\phi}$ : função do decoder parametrizada por  $\phi$ .



# Função Custo

## 6 Encoders and Decoders Estocásticos

Algumas escolhas comuns para a função de perda incluem:

- **Mean Squared Error (MSE):**

$$L(x, z) = \|x - z\|_2^2$$

- **Cross-Entropy Loss:**

$$L(x, z) = - \sum_i [x_i \log(z_i) + (1 - x_i) \log(1 - z_i)]$$

## Score Matching e DAE

### 6 Encoders and Decoders Estocásticos

**Score Matching:** Técnica de estimação de densidade que visa aprender o gradiente do logaritmo da densidade de probabilidade dos dados, conhecido como *score function* [8]. Aprender o gradiente do logaritmo da densidade de probabilidade é uma maneira alternativa de aprender a estrutura da densidade de probabilidade dos dados [6].

$$\nabla_x \log p_{data}(x)$$

**Resultado Teórico [9]:** Para ruído Gaussiano pequeno ( $\sigma \rightarrow 0$ ), treinar um DAE é equivalente a estimar o score da distribuição de dados:

$$\nabla_x \log p_{data}(x) \approx \frac{g(f(x)) - x}{\sigma^2}$$

**Implicações:** Isso sugere que DAEs não apenas aprendem a reconstruir entradas limpas, mas também capturam informações sobre a estrutura subjacente da distribuição dos dados.

# Interpretação Geométrica - Manifold

## 6 Encoders and Decoders Estocásticos

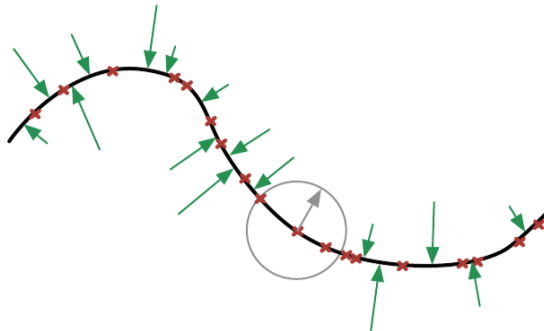
Mas o que exatamente o DAE está aprendendo?

- **Hipótese:** Dados reais residem em um **variedade** (manifold)  $\mathcal{M}$  de dimensão  $d' \ll d$   
 $\mathbb{R}^d$
- **Exemplos:** Imagens naturais, sinais de áudio, etc.
- **Ruído:** Adiciona pequenas perturbações que movem os pontos para fora da variedade  $\mathcal{M}$

Podemos dizer então que o DAE está aprendendo a projetar pontos ruidosos de volta para essa variedade  $\mathcal{M}$  [7][10].

# Interpretação Geométrica - Manifold

## 6 Encoders and Decoders Estocásticos



**Figura:** Em verde, o vetor resultante de  $g(f(\tilde{x})) - \tilde{x}$ . Em vermelho, os dados originais sem corrupção por ruído. Em cinza, o dado corrompido gerado a partir de uma perturbação equiprovável gerando uma amostra de  $\mathcal{C}(\tilde{x}|x)$ .

# Treino vs. Teste em DAE

6 Encoders and Decoders Estocásticos

Na fase de treino:

- Entrada corrompida com ruído:  $\tilde{x} \sim \mathcal{C}(\tilde{x} | x)$
- Alvo:  $x$
- Objetivo: reconstruir a entrada limpa  $x$  a partir de  $\tilde{x}$

## Treino vs. Teste em DAE

### 6 Encoders and Decoders Estocásticos

Na fase de teste/produção, tipicamente, temos duas possibilidades:

- Usar o DAE como um autoencoder padrão:
  - Entrada:  $x$
  - Saída de interesse:  $h = f(x)$  (tipicamente)
  - Objetivo: obter uma representação robusta dos dados de entrada, já que o modelo foi treinado para aprender mais características relevantes do que numa aplicação padrão.
- Usar o DAE para denoising:
  - Entrada:  $\tilde{x}$
  - Saída de interesse:  $g(f(\tilde{x}))$
  - Objetivo: remover o ruído da entrada, aproveitando a capacidade do DAE de aprender a reconstruir entradas limpas a partir de versões ruidosas.

# Variational Autoencoder

## 6 Encoders and Decoders Estocásticos

- Diferentemente dos **Autoencoders tradicionais**, que replicam principalmente os dados de entrada, os **VAEs podem gerar novos resultados**.
- Eles funcionam **identificando e aprendendo características ocultas** dos conjuntos de dados de treinamento.
- Essas características aprendidas são usadas como **representação latente** para **gerar novos dados**.

# Variational Autoencoder

## 6 Encoders and Decoders Estocásticos

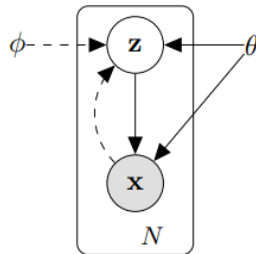
- *Criação dos VAEs surgiram da seguinte pergunta*
  - “Como podemos realizar inferência e aprendizado eficientes em modelos probabilísticos direcionados, na presença de variáveis latentes contínuas com distribuições posteriores intratáveis e grandes conjuntos de dados?” [11]
- Criado modelo para derivar um estimador de limite inferior (uma função objetivo estocástica) para uma variedade de modelos gráficos dirigidos com variáveis latentes contínuas.



# Variational Autoencoder: Encoder e Decoder

## 6 Encoders and Decoders Estocásticos

- Esta figura apresenta dois DAGs interconectados: o **encoder** e o **decoder**.
- O fluxo do **encoder** (linha pontilhada) mostra a compressão de uma imagem de entrada  $x$  em uma representação latente  $z$ . O símbolo  $\phi$  representa os parâmetros do encoder.
- O fluxo do **decoder** (linha sólida) ilustra o processo de reconstrução, transformando  $z$  em uma imagem próxima de  $x$ . O símbolo  $\theta$  representa os parâmetros do decoder.
- Tanto o encoder quanto o decoder são **probabilísticos**, e suas distribuições de probabilidade apresentam **dependências condicionais**, conectadas via  $z$ .



# Variational Autoencoder: Encoder e Decoder

## 6 Encoders and Decoders Estocásticos

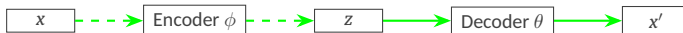
- O **encoder** modela  $P(z|x)$ : recebe  $x$  e produz  $z$ .



- O **decoder** modela  $P(x|z)$ : recebe  $z$  e reconstrói  $x'$  próximo de  $x$ .



- Fluxo completo:  $x \rightarrow z \rightarrow x'$



# Variáveis Latentes Contínuas

## 6 Encoders and Decoders Estocásticos

- Se conseguirmos determinar a **distribuição das variáveis latentes** e amostrar do espaço latente:
  - Podemos usar o decoder para gerar novas imagens.
  - As imagens geradas seguem a mesma estrutura e características do conjunto de treinamento.
  - Não é necessário fornecer a imagem de entrada ou usar o encoder.
- Objetivo principal: **usar o decoder como gerador de imagens.**
- Desafio: **determinar com precisão a distribuição das variáveis latentes  $z$  é intratável.**

# Distribuições Posteriores Intratáveis

## 6 Encoders and Decoders Estocásticos

- O **encoder** captura a essência da imagem de entrada  $x$  e a representa no **espaço latente**.
- Ele modela a **distribuição posterior**  $P(z \mid x)$ , ou seja, a probabilidade das variáveis latentes  $z$  dado os dados observados  $x$ .
- Determinar essa distribuição com precisão é **complexo** devido às relações intrincadas entre dados e variáveis latentes.
- Essa complexidade torna a **posterior intratável**.

# Distribuições Posteriores Intratáveis

## 6 Encoders and Decoders Estocásticos

$$P(z | x) = \frac{P(x | z) P(z)}{P(x)}$$

- O decoder modela a verossimilhança  $P(x | z)$ , ou seja, a probabilidade de observar uma imagem  $x$  dado as variáveis latentes  $z$ .
- A distribuição a priori  $P(z)$  captura nossas crenças ou suposições sobre as variáveis latentes  $z$  antes de observar qualquer imagem  $x$ .
- No contexto dos VAEs, nossa escolha do prior (geralmente uma distribuição normal padrão) é motivada pela conveniência computacional.

# Distribuições Posteriores Intratáveis

## 6 Encoders and Decoders Estocásticos

- O denominador  $P(x)$  representa a verossimilhança marginal ou evidência.
- Ele quantifica a probabilidade de observar os dados da imagem  $x$  sem condicionar a qualquer valor específico de  $z$ .
- Para determiná-lo, precisamos calcular a densidade de probabilidade da imagem  $x$  para cada valor possível de  $z$ , e então integrar todos esses valores:

$$P(x) = \int P(x | z)P(z) dz$$

- Se pudéssemos calcular  $P(x)$  com precisão, usaríamos a distribuição posterior  $P(z | x)$  para amostrar as características latentes.
- No entanto, as complexidades de dados de alta dimensão, estruturas de modelo e a necessidade de integração sobre o espaço latente tornam o cálculo direto de  $P(x)$  praticamente impossível.
- Esse desafio inerente torna a **posterior**  $P(z | x)$  **intratável**.

# Inferência Variacional

## 6 Encoders and Decoders Estocásticos

- Inferência Variacional (VI): método usado para aproximar distribuições posteriores complexas e intratáveis por distribuições mais simples e tratáveis.
- Ideia principal:
  1. Escolher uma distribuição aproximadora: selecionar uma família de distribuições, tipicamente mais simples que a posterior verdadeira, com parâmetros ajustáveis.
  2. Otimizar para minimizar a diferença: ajustar os parâmetros da distribuição aproximadora com base nos dados observados, tornando-a o mais próxima possível da posterior verdadeira.
- A medida de proximidade geralmente é a divergência de Kullback-Leibler (KL).
- A inferência variacional transforma o problema intratável em otimização, maximizando o **Evidence Lower Bound (ELBO)**:

$$\log P(\mathbf{x}) \geq \mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log P_{\theta}(\mathbf{x} | \mathbf{z})] - D_{KL}(q_{\phi}(\mathbf{z} | \mathbf{x}) || P(\mathbf{z}))$$

- Em VAEs, isso permite treinar o encoder ( $q_{\phi}$ ) e o decoder ( $P_{\theta}$ ) de forma eficiente, mesmo com variáveis latentes contínuas e dados de alta dimensão.

# Table of Contents

## 7 Aplicações

- ▶ Motivação
- ▶ Formulação Matemática
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ Considerações sobre Arquitetura
- ▶ Autoencoders com regularização
- ▶ Encoders and Decoders Estocásticos
- ▶ **Aplicações**
- ▶ Referências Bibliográficas
- ▶ Anexos



# RBM e Deep Autoencoder[12]

## 7 Aplicações

- Objetivo: reduzir a dimensionalidade de dados complexos.
- Abordagem:
  - Treinar uma pilha de RBMs camada a camada (pretraining).
  - Cada RBM aprende features da camada anterior.
  - O bottleneck code na camada central representa a versão compacta dos dados.
- Unrolling: conecta todas as RBMs em uma rede única formando um deep autoencoder.
- Fine-tuning: ajuste de todos os pesos com backpropagation para minimizar erro de reconstrução.
- Resultado: código de baixa dimensão que preserva estrutura não-linear dos dados, superando PCA.

# Arquitetura do Deep Autoencoder

## 7 Aplicações

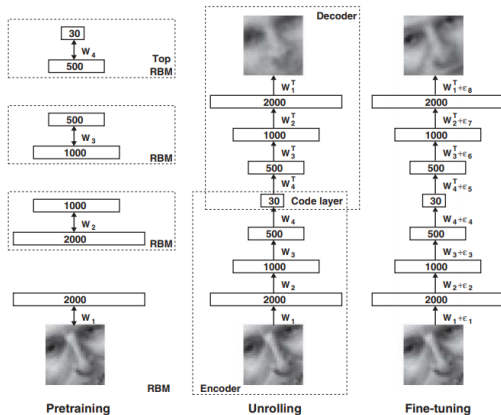
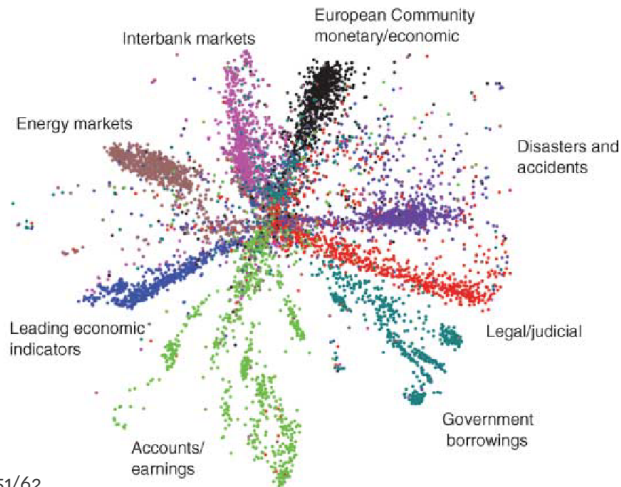


Figura: Exemplo de arquitetura: RBMs empilhadas são "unrolled" em um deep autoencoder.

# Deep Autoencoder Aplicado a Documentos[12]

## 7 Aplicações



- 804k documentos do corpus Newswire
- Frequencia dos 2000 radicais mais frequentes
- Autoencoder  
2000-500-250-125-2

# U-Net[13]

7 Aplicações

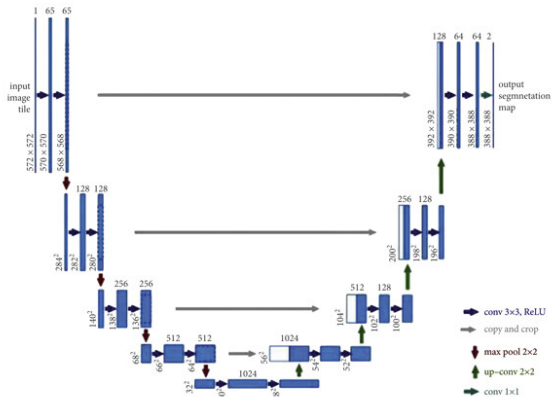


Figura: Arquitetura U-Net típica usada para tarefas de segmentação de imagens médicas.

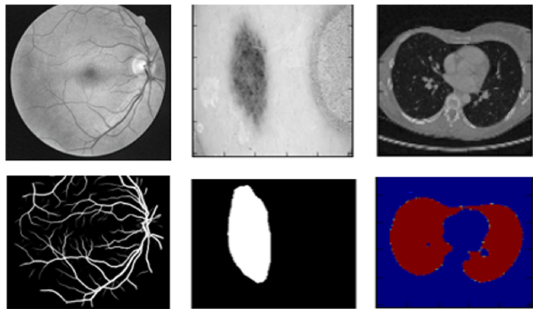
# U-Net

## Características Principais

- **Arquitetura em U:** Composta por um caminho de contração (encoder) e um caminho de expansão (decoder), formando uma estrutura em U.
- **Camadas Convolucionais:** Utiliza camadas convolucionais para capturar características espaciais das imagens.
- **Skip Connections:** Conecta diretamente camadas correspondentes do encoder e decoder, preservando informações espaciais detalhadas.

# U-Net

Exemplos



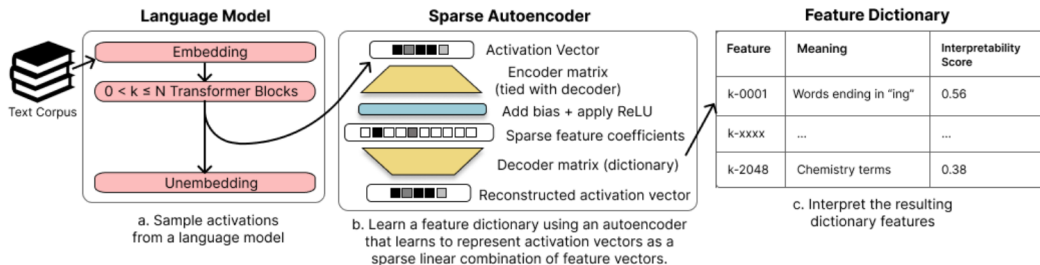
**Figura:** Imagem de entrada (em cima). Imagem segmentada (em baixo).



**Figura:** Saída de camadas intermediárias da U-Net.

# Sparse Autoencoders para Interpretabilidade de LLMs

## 7 Aplicações



- Pesquisadores da EleutherAI, apresentado como poster na ICLR2023
- Usa autoencoder com esparsidade por  $L_1$
- Espaço latente esparso a partir de estado intermediário de LLM
- Usa figura de mérito de interpretabilidade de features feita com outra LLM

# Table of Contents

## 8 Referências Bibliográficas

- ▶ Motivação
- ▶ Formulação Matemática
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ Considerações sobre Arquitetura
- ▶ Autoencoders com regularização
- ▶ Encoders and Decoders Estocásticos
- ▶ Aplicações
- ▶ **Referências Bibliográficas**
- ▶ Anexos



## Referências Bibliográficas

### 8 Referências Bibliográficas

- [1] Rumelhart, E. David, M. James, and L. James, *Parallel distributed processing: explorations in the microstructure of cognition. Volume 1. Foundations*. O1 1986.
- [2] U. Michelucci, “An introduction to autoencoders,” CoRR, vol. abs/2201.03898, 2022.
- [3] M. Tschannen, O. Bachem, and M. Lucic, “Recent advances in autoencoder-based representation learning,” CoRR, vol. abs/1812.05069, 2018.
- [4] E. Oja, “Simplified neuron model as a principal component analyzer,” *Journal of Mathematical Biology*, vol. 15, pp. 267–273, 1982.
- [5] S. Stevens, W.-L. Chao, T. Berger-Wolf, and Y. Su, “Sparse autoencoders for scientifically rigorous interpretation of vision models,” 2025.

## Referências Bibliográficas

### 8 Referências Bibliográficas

- [6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.  
<http://www.deeplearningbook.org>.
- [7] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” pp. 1096–1103, 01 2008.
- [8] A. Hyvarinen, “Estimation of non-normalized statistical models by score matching,” *Journal of Machine Learning Research*, vol. 6, pp. 695–709, 04 2005.
- [9] P. Vincent, “A connection between score matching and denoising autoencoders,” *Neural Computation*, vol. 23, no. 7, pp. 1661–1674, 2011.

## Referências Bibliográficas

### 8 Referências Bibliográficas

- [10] G. Alain and Y. Bengio, “What regularized auto-encoders learn from the data generating distribution,” 2014.
- [11] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” 2022.
- [12] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [13] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, (Cham), pp. 234–241, Springer, 2015.

# Table of Contents

9 Anexos

- ▶ Motivação
- ▶ Formulação Matemática
- ▶ Undercomplete Autoencoders com redes neurais
- ▶ Considerações sobre Arquitetura
- ▶ Autoencoders com regularização
- ▶ Encoders and Decoders Estocásticos
- ▶ Aplicações
- ▶ Referências Bibliográficas
- ▶ Anexos

# Divergência Kullback-Leibler

9 Anexos

**Entropia de  $P$ :** Custo ótimo para codificar dados de  $P$

$$H(P) = \mathbb{E}_{x \sim P}[-\log P(x)] = - \sum_x P(x) \log P(x)$$

**Entropia Cruzada:** Custo de usar código ótimo para  $Q$  em dados de  $P$

$$H_{\text{cross}}(P, Q) = \mathbb{E}_{x \sim P}[-\log Q(x)] = - \sum_x P(x) \log Q(x)$$

**Divergência KL:** o quão pior é  $Q$  comparado ao ótimo  $P$ .

$$D_{KL}(P \| Q) = H_{\text{cross}}(P, Q) - H(P) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

# Autoencoders: da motivação às variantes modernas

*Obrigado pela Atenção!*

*Alguma Pergunta?*

*Felipe Fink Grael, Rafael Tadeu C. dos Santos,  
Thalles Nonato Leal Santos e Jefferson Osowsky*