

SICO70

SISTEMAS INTELIGENTES 2

Aula 08 - Autoencoders

Prof. Rafael G. **Mantovani**

Roteiro



- 1 Introdução
- 2 AEs
- 3 Stacked AEs (SAEs)
- 4 *Denoising AEs (dAEs)*
- 5 Referências

Roteiro

- 1** Introdução
- 2** AEs
- 3** Stacked AEs (SAEs)
- 4** *Denoising AEs (dAEs)*
- 5** Referências

Introdução



Introdução



Figura de: Dagupsta (2018)

Introdução

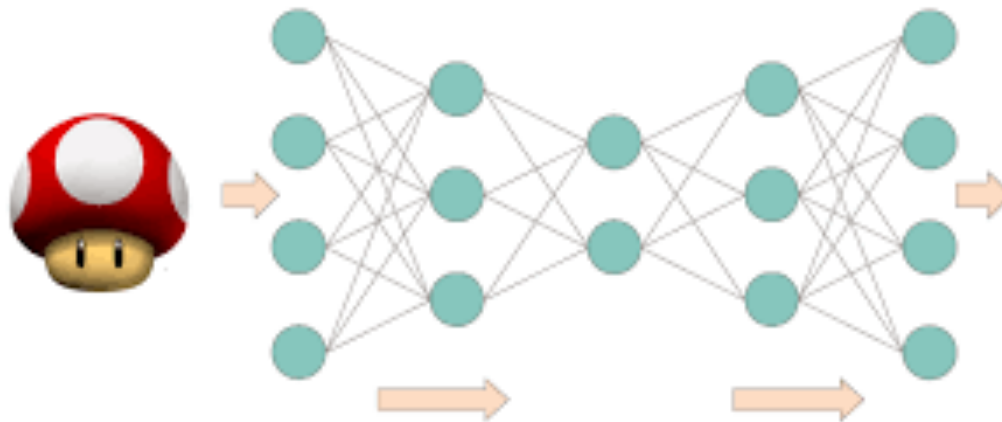


Figura de: Dagupsta (2018)

Introdução

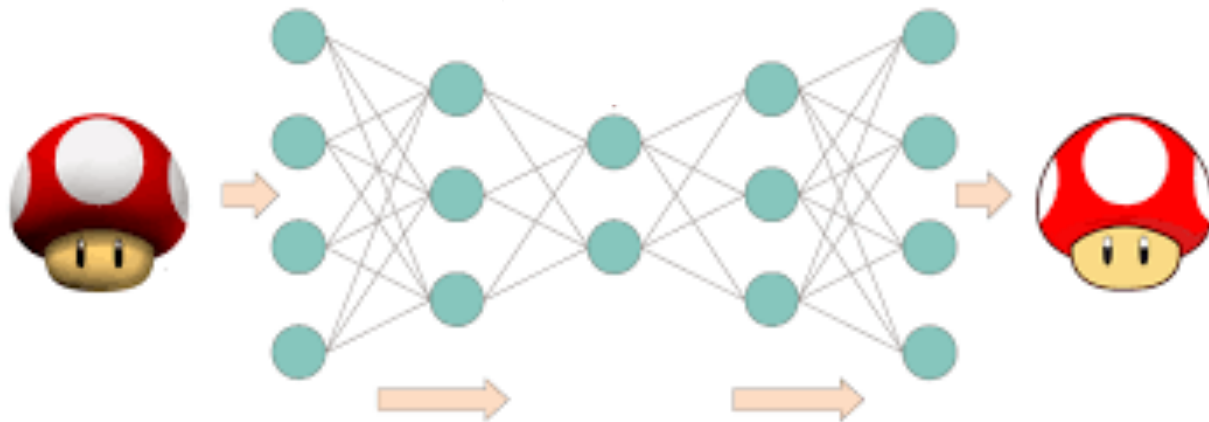


Figura de: Dagupsta (2018)

Introdução

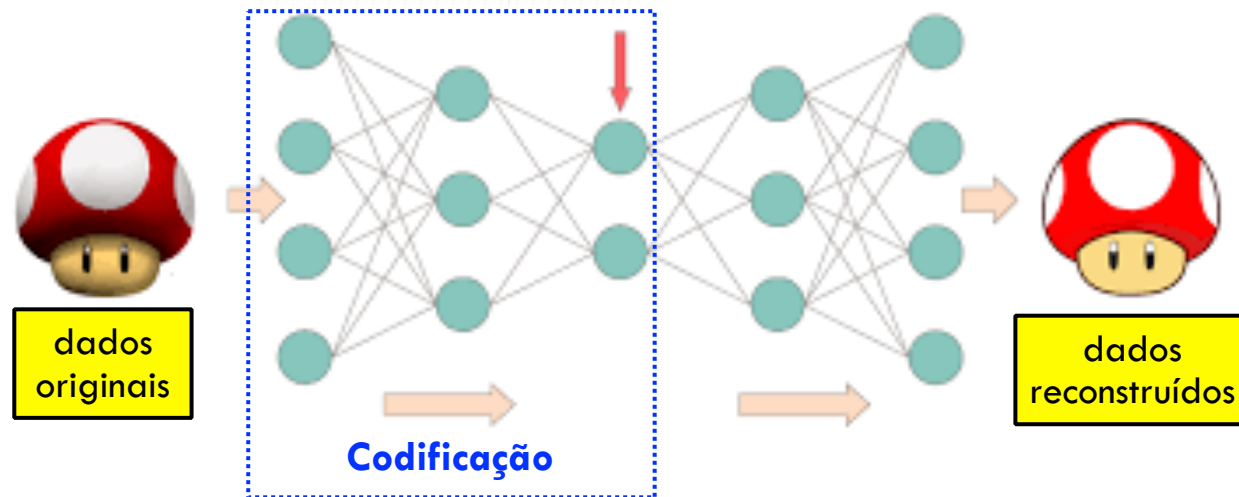


Figura de: Dagupsta (2018)

Introdução

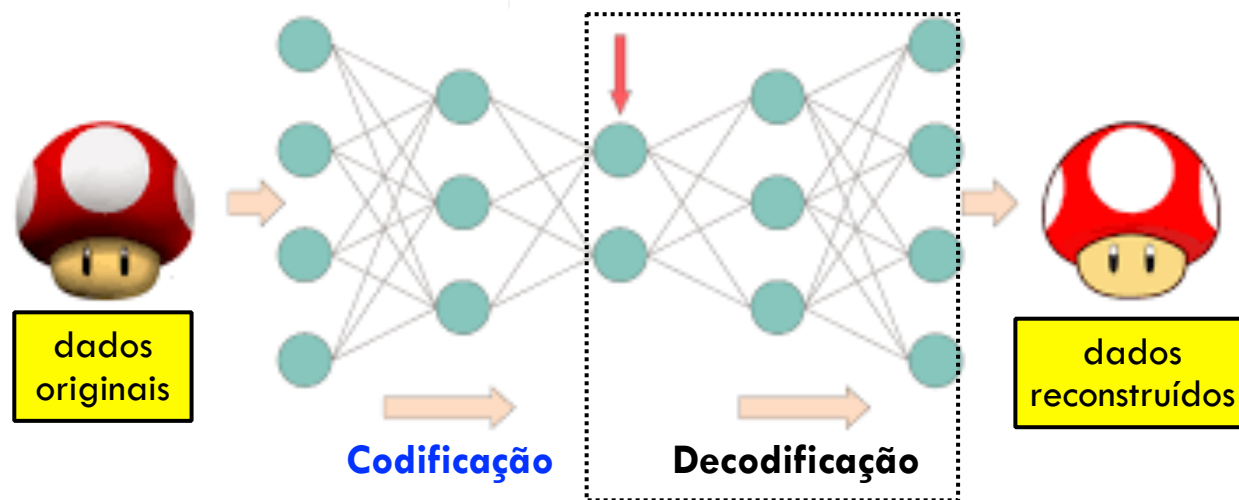


Figura de: Dagupsta (2018)

Introdução

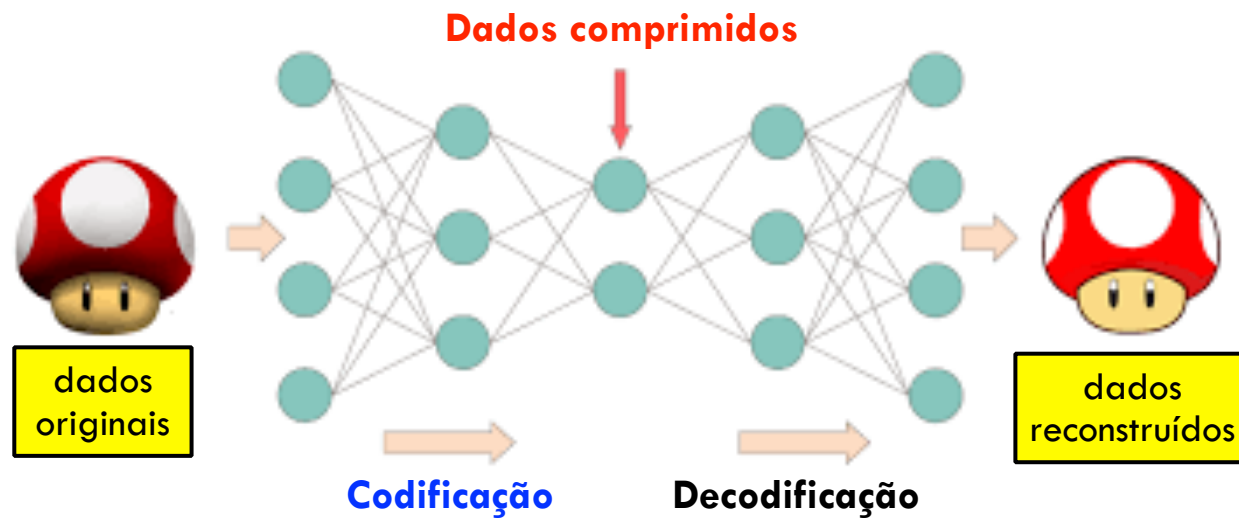


Figura de: Dagupsta (2018)

Introdução

□ **Autoencoders** (AEs)

- são capazes de aprender representações/padrões dos dados de entrada
- aprendizado é de maneira **não supervisionada**, pois o conjunto de treinamento não é rotulado
- representações latentes (**codings**): possuem uma dimensionalidade muito menor que o conjunto de entrada
- representação aprendida na camada oculta (meio)

Introdução

- Onde usamos AEs?
 - úteis para redução de dimensionalidade/visualização de dados
 - agem como **detectores** de características
 - podem ser usados para **inicialização dos pesos** sinápticos de redes neurais profundas (DNNs)
 - alguns AEs são **modelos generativos** → criam novos dados que se parecem com os dados do conjunto de treinamento

Introdução

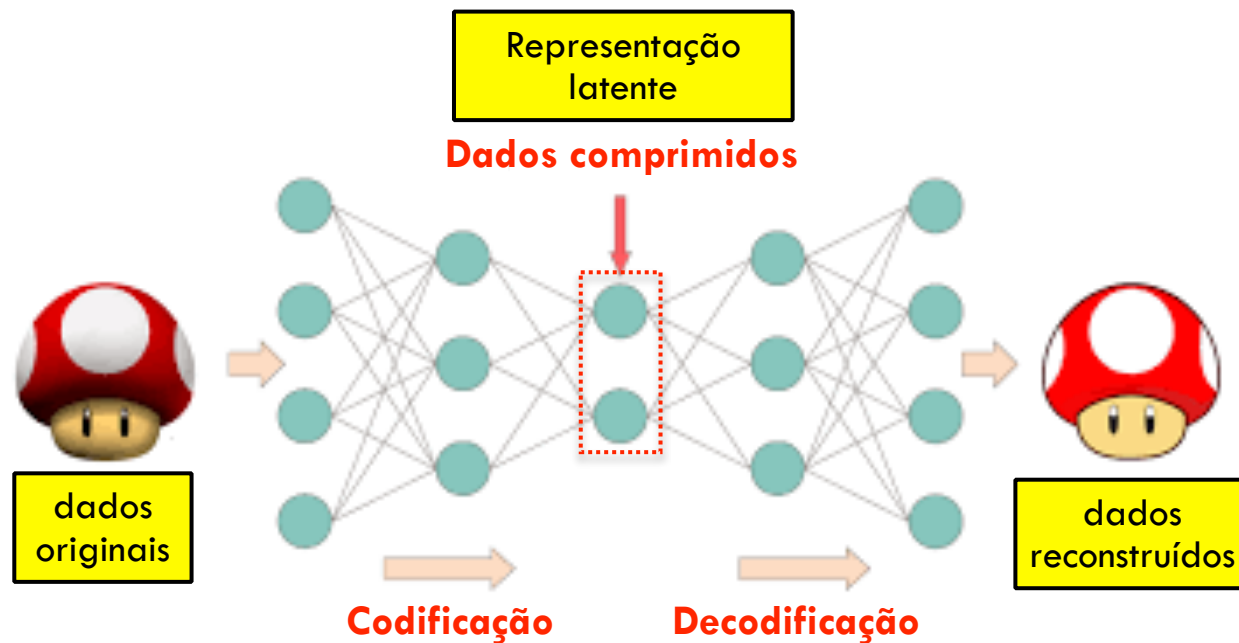


Figura de: Dagupsta (2018)

Introdução

- AEs simplesmente **aprendem** como reproduzir as entradas nas saídas do modelo
- Também podemos:
 - limitar o tamanho da **representação latente** (neurônios na camada oculta)
 - adicionar ruído às entradas e treinar a rede para recuperar os dados originais
- Esses processos forçam os AEs a aprenderem formas eficientes de representar dados

Motivação

- Duas sequências de números:
 - $s1 = 40, 27, 25, 36, 81$
 - $s2 = 50, 48, 46, 44, 42, 40, 38, 36, 34, 32$

Motivação

- Duas sequências de números:
 - $s1 = 40, 27, 25, 36, 81$
 - $s2 = 50, 48, 46, 44, 42, 40, 38, 36, 34, 32$
- Qual é a mais fácil de memorizar?

Motivação

- Duas sequências de números:
 - $s1 = 40, 27, 25, 36, 81$
 - $s2 = 50, 48, 46, 44, 42, 40, 38, 36, 34, 32$
- Qual é a mais fácil de memorizar?
 - primeiro momento, talvez $s1$
 - mas, em $s2$ podemos ver que temos os números pares de 50 até 32
 - **padrões** são mais fáceis de se memorizar do que sequências em si
 - AEs tentam descobrir/explorar padrões dos dados

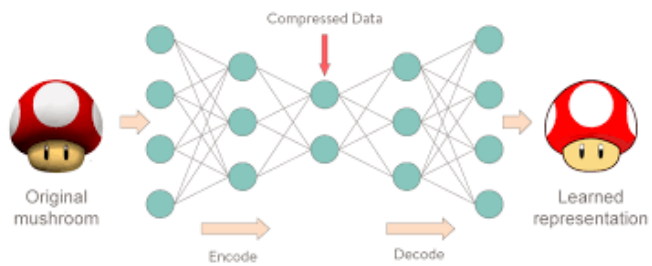
Motivação

- William Chase & Herbert Simon (70's)
 - experimentos sobre a relação entre memória, percepção e reconhecimento de padrões
 - observaram jogadores de xadrez memorizando as peças de um tabuleiro
 - jogadores precisavam de 5 segundos para memorizar a posição de todas elas

Motivação

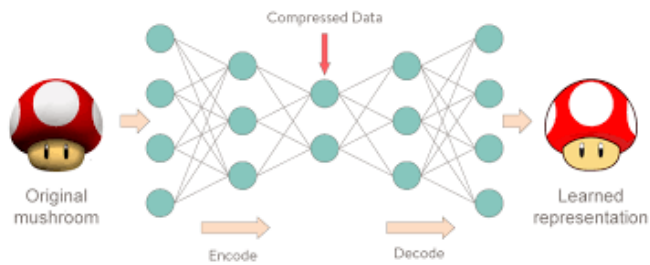
- William Chase & Herbert Simon (70's)
 - experimentos sobre a relação entre memória, percepção e reconhecimento de padrões
 - observaram jogadores de xadrez memorizando as peças de um tabuleiro
 - jogadores precisavam de 5 segundos para memorizar a posição de todas elas
- não significa que eles tem mais memória do que nós
 - eles consegue ver os **padrões** mais fácil
 - notar esses padrões os ajuda a armazenar informação de maneira mais eficiente

Motivação



Autoencoders (AEs)

Motivação

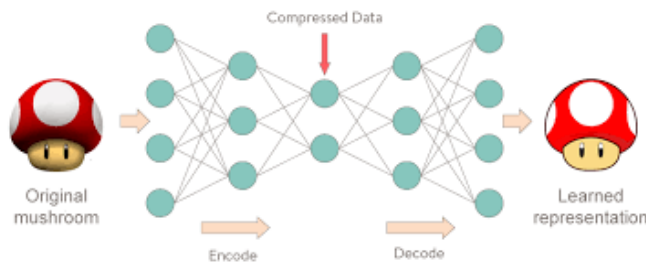


Autoencoders (AEs)



Jogador de xadrez

Motivação



Autoencoders (AEs)



Jogador de xadrez

- **AEs:**
 - agem como jogadores de xadrez
 - olham para os dados (entradas)
 - convertem em representações latentes/padrões
 - e geram algo que se parece bem próximo das entradas

Roteiro

- 1 Introdução
- 2 AEs
- 3 Stacked AEs (SAEs)
- 4 *Denoising AEs (dAEs)*
- 5 Referências

AEs

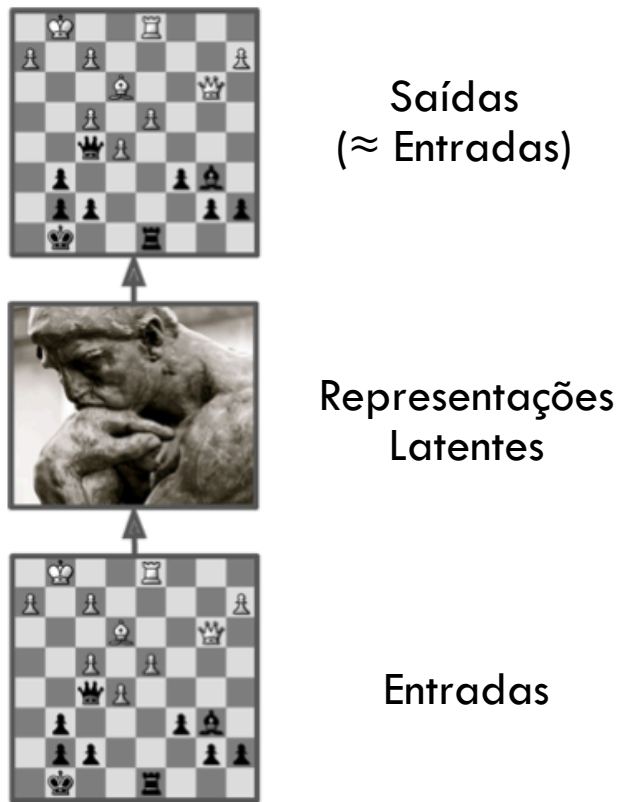


- **AEs** possuem duas partes:

AEs

- **AEs** possuem duas partes:
 - codificador - **encoder** (rede de reconhecimento): converte os padrões de entrada em representações latentes
 - decodificador - **decoder** (rede generativa): converte a representação interna e compacta nas saídas

AEs



AEs

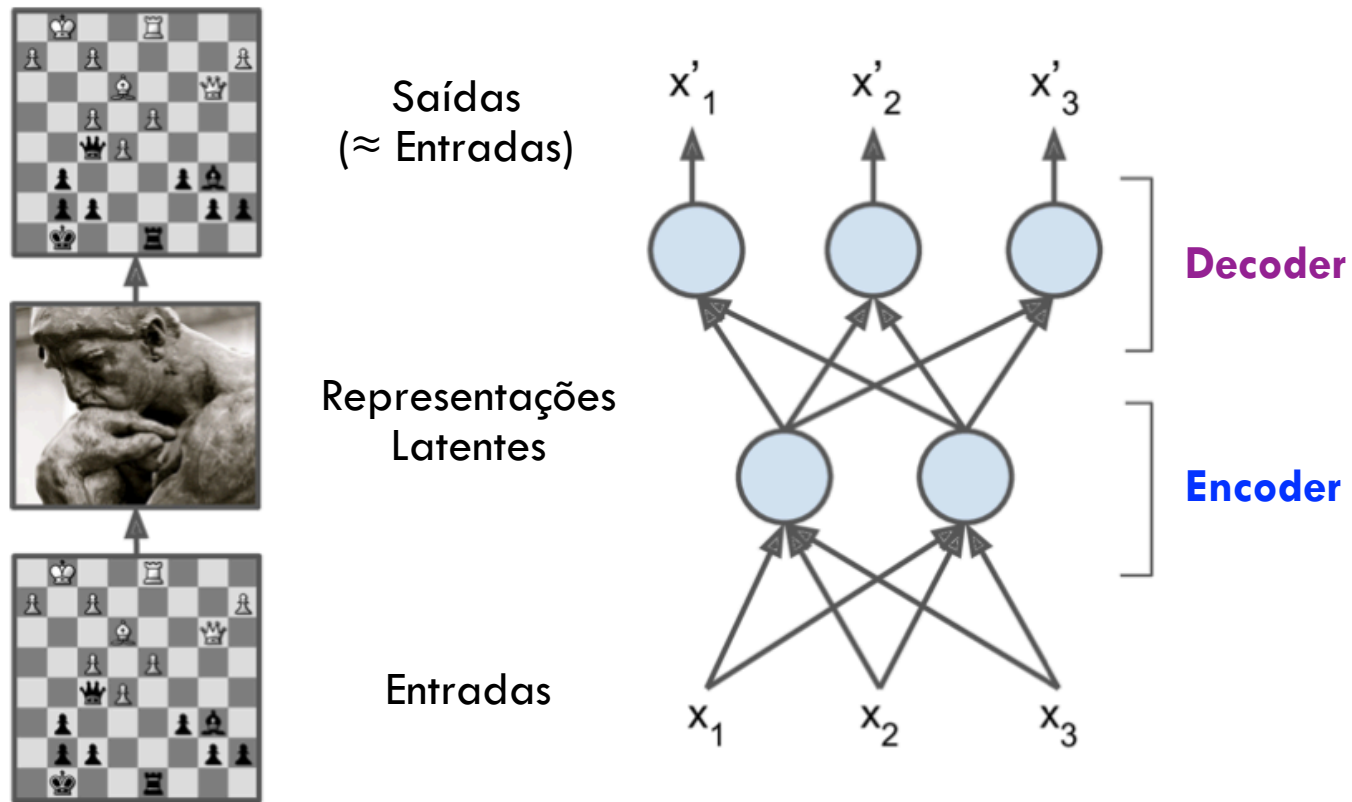
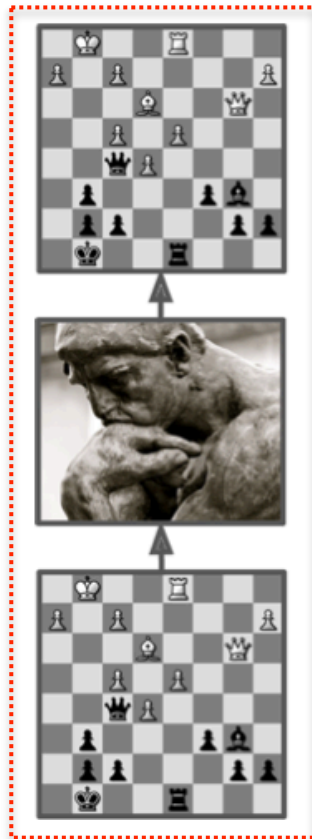


Figura de: Aurélien Gerón (2019)

AEs

Jogador de xadrez

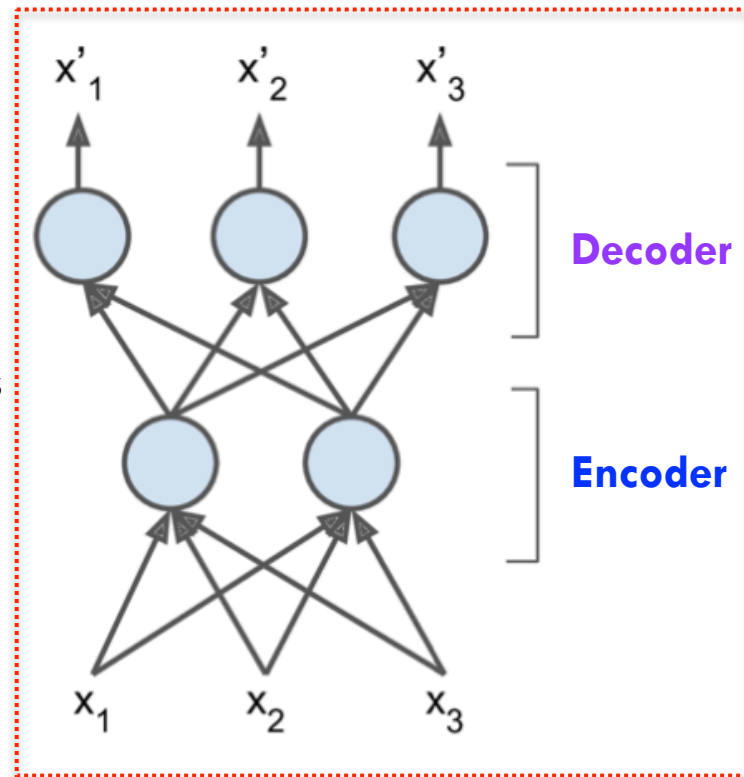


Saídas
(\approx Entradas)

Representações
Latentes

Entradas

Autoencoders (AEs)



AEs

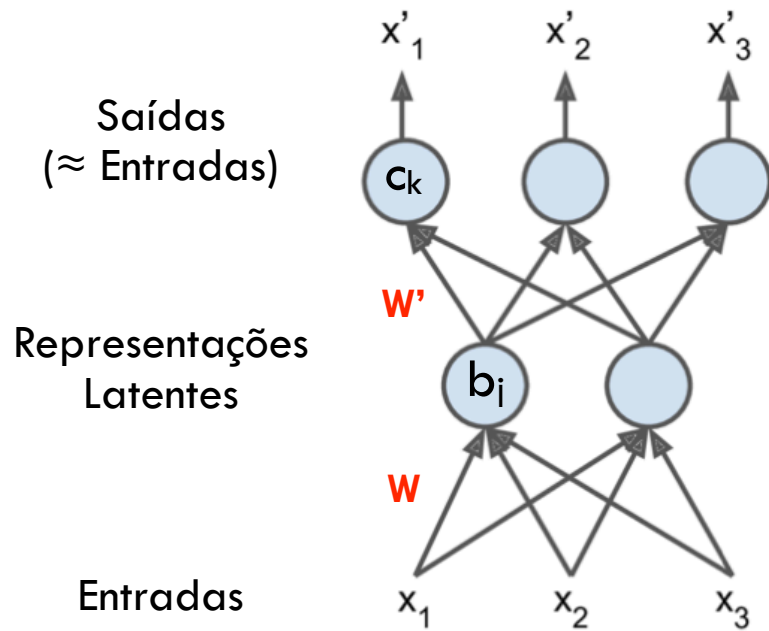


Figura adaptada de: Aurélien Gerón (2019)

AEs

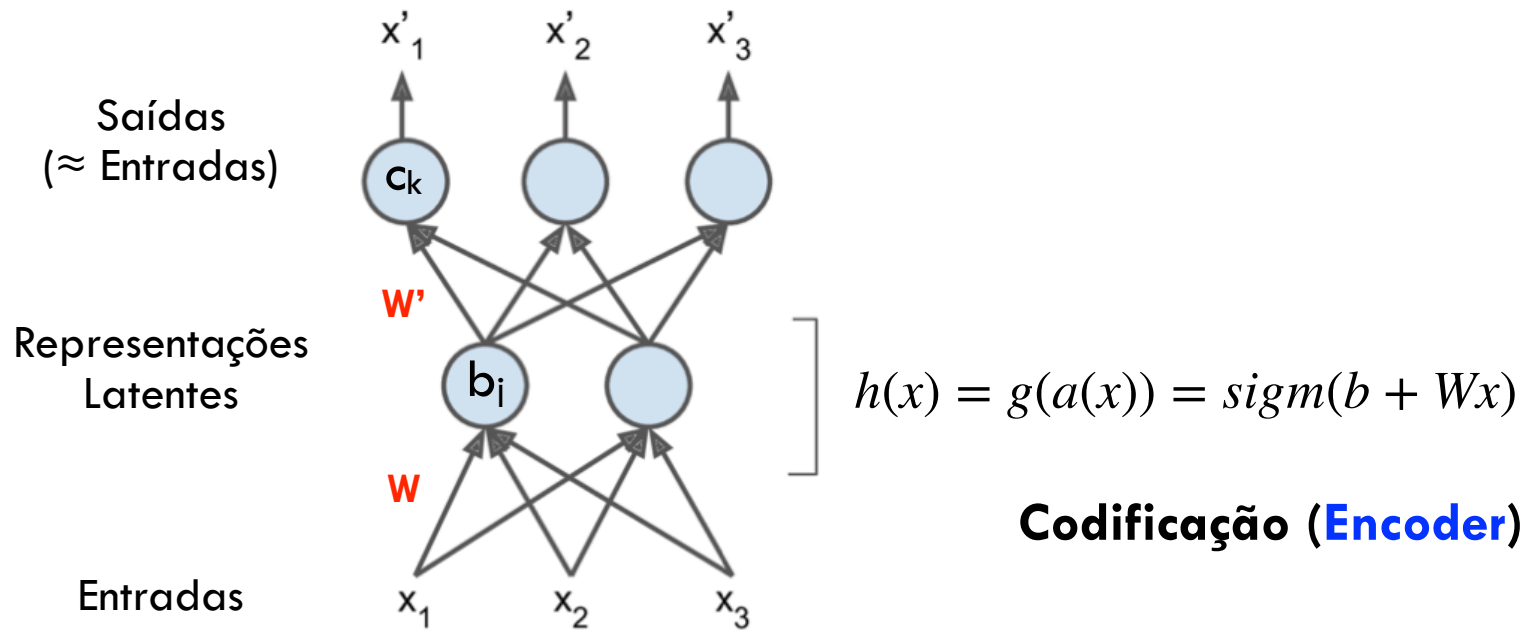


Figura adaptada de: Aurélien Gerón (2019)

AEs

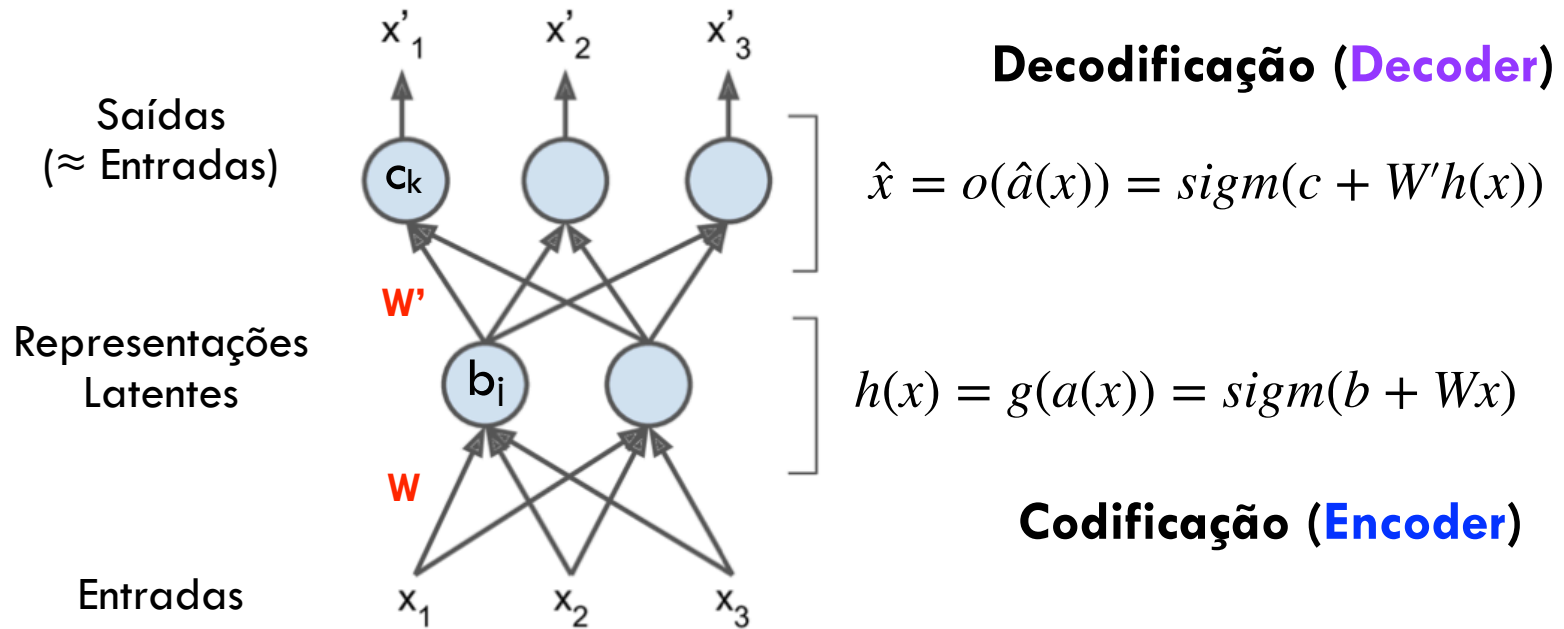


Figura adaptada de: Aurélien Gerón (2019)

□ Observações:

- AE mais simples tem a mesma arquitetura que uma MLP, exceto pela camada de saída ter o mesmo tamanho que as entradas
- camada oculta: **Encoder**
- camada de saída: **Decoder**
- saídas são chamadas de **reconstruções**, porque os AEs tentam reconstruir as entradas

Funções de custo (loss)

- Função de custo/erro:
 - contém uma **medida de reconstrução** (*loss reconstruction*) que penaliza o modelo quando as reconstruções são diferentes das entradas
 - vai comparar x com \hat{x} para medir o quão boa é a reconstrução
 - treinamos o AE para minimizar essa função por meio do gradiente descendente

Funções de custo (loss)

- Para entradas binárias

$$l(f(x)) = - \sum_k (x_k \log(\hat{x}_k) + (1 - x_k) \log(1 - \hat{x}_k))$$

- onde:
 - k é o índice da instância
 - se $x_k = 1$, tentamos “puxar” \hat{x}_k para 1
 - se $x_k = 0$, tentamos “puxar” \hat{x}_k para 0

Funções de custo (loss)

- Para entradas reais

$$l(f(x)) = \frac{1}{2} \sum_k (\hat{x}_k - x_k)^2$$

- soma das diferenças ao quadrado
 - distância euclidiana quadrática
 - função de ativação linear na camada de saída
-
- Em **ambos os casos** (saídas binárias e reais) o treinamento é feito via **Backpropagation**

Hands on



Vamos exercitar :)

Roteiro

- 1 Introdução
- 2 AEs
- 3 Stacked AEs (SAEs)
- 4 *Denoising AEs (dAEs)*
- 5 Referências

Stacked Ads

- Os AEs
 - podem também ter várias camadas empilhadas
 - nesse caso são chamados de **Stacked Autoencoders (SAEs)** ou **Deep Autoencoders (DAEs)**
 - mais camadas permitem aos AEs aprenderem padrões mais complexos
 - a arquitetura de um SAE é simétrica em relação à camada oculta das representações latentes (*coding layer*)
 - parece um “sanduíche”

Stacked Ads

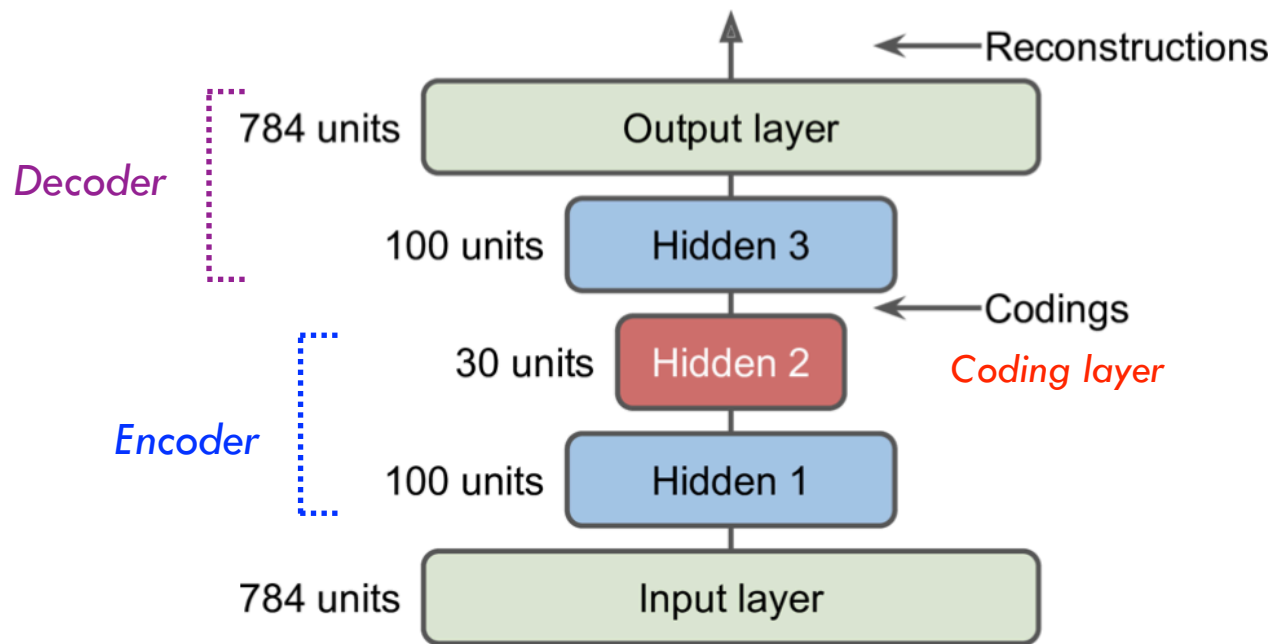


Figura de: Aurélien Géron (2019)

Visualizando as reconstruções

- Um jeito de garantir que o AE está propriamente treinado é comparar as entradas com as saídas
 - as diferenças não podem ser significativas

Visualizando as reconstruções

- Um jeito de garantir que o AE está propriamente treinado é comparar as entradas com as saídas
 - as diferenças não podem ser significativas



Entradas



*Reconstruções
(saídas)*

Figura de: Aurélien Gerón (2019)

Visualizando as reconstruções



*Reconstrução é perceptível,
porém não tão boa :/*

Figura de: Aurélien Gerón (2019)

- o que fazer?
 - treinar o modelo por mais épocas
 - tornar encoder/decoder mais profundo (mais camadas)
 - aumentar o tamanho da camada oculta que armazena as representações latentes (codings)

Visualizando as reconstruções

□ Mas:



- cuidado ao definir a arquitetura
- se a rede for muito “poderosa”, ela vai mapear perfeitamente as entradas nas reconstruções sem aprender nenhum padrão útil
- não irá extrair características úteis

SAEs

- Uma vez que o SAE é treinado, podemos usá-lo para **reduzir a dimensionalidade** do dataset
- Grande vantagem dos SAEs:
 - é serem capazes de lidar com **grandes quantidades de dados**
- Geralmente se usa os AEs para reduzir a dimensionalidade para um nível razoável, e depois alimentar uma técnica de visualização de dados (PCA, t-SNE)

SAEs



Figura de: Aurélien Gerón (2019)

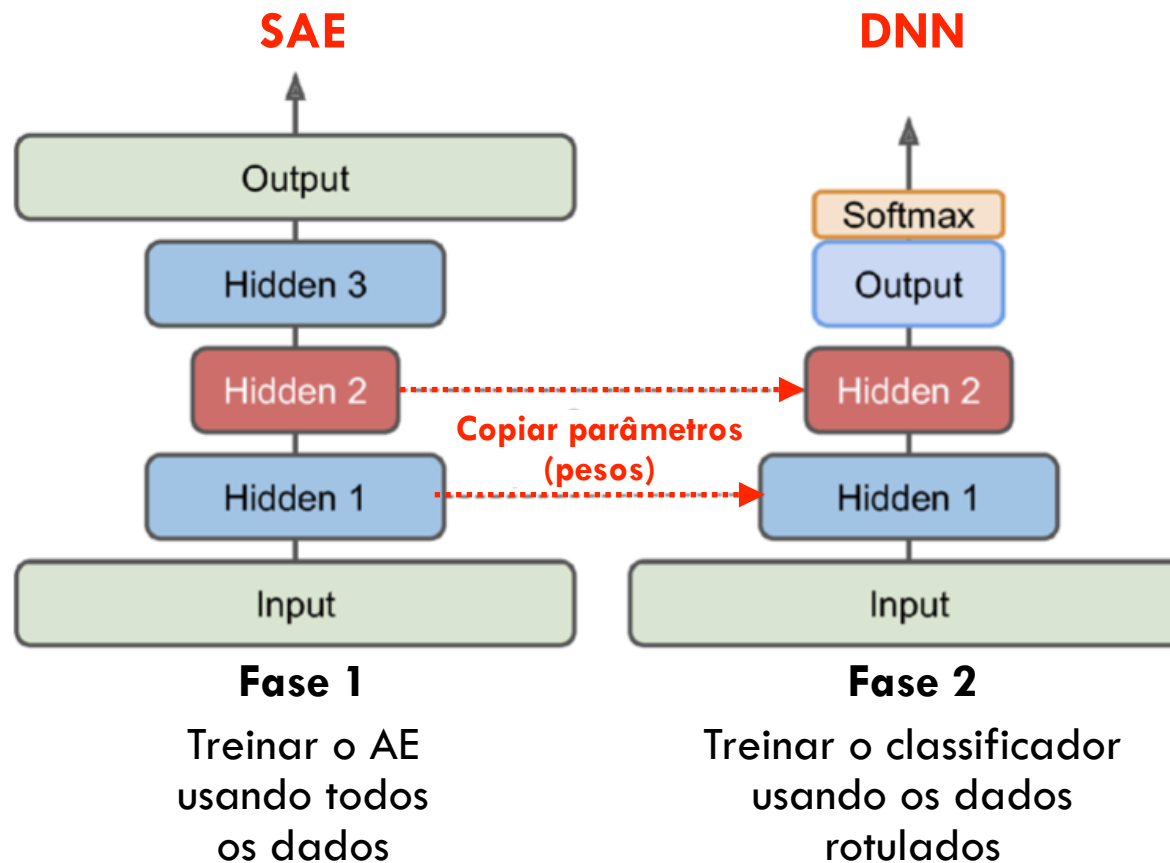
Pré-treino usando SAEs

- **Situação:** tarefa supervisionada complexa
 - mas não temos muitos exemplos rotulados
 - a maioria dos exemplos são não-rotulados

Pré-treino usando SAEs

- **Situação:** tarefa supervisionada complexa
 - mas não temos muitos exemplos rotulados
 - a maioria dos exemplos são não-rotulados
- **Solução:** encontrar uma NN que desempenhe bem em uma tarefa similar e usar os pesos sinápticos gerados por ela
 - treinar SAE com todos os dados
 - reusar os pesos sinápticos das camadas do SAE para criar uma DNN
 - treinar DNN usando apenas os dados rotulados

Pré-treino usando SAEs



Amarrando Pesos

- Quando um SAE é puramente simétrico, uma abordagem comum é “**amarrar**” os pesos de todas as camadas do decodificador (decoder) às camadas do codificador (encoder)
 - Isto reduz pela metade o número de pesos no modelo
 - acelerando o aprendizado
 - limitando o risco de overfitting

Amarrando Pesos

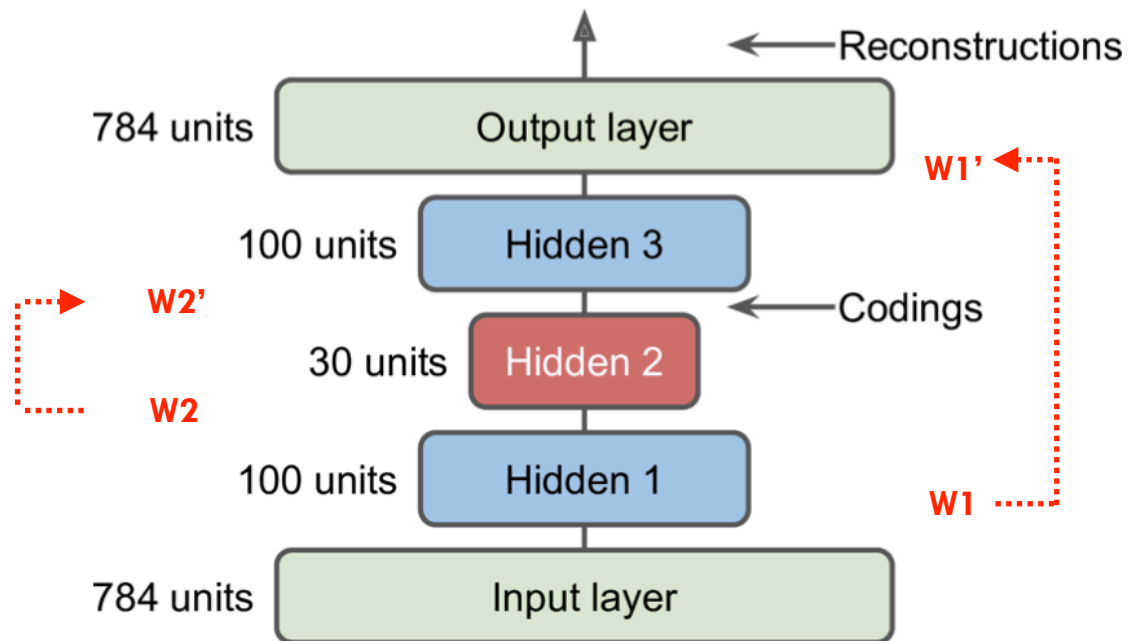


Figura de: Aurélien Géron (2019)

Outros modelos de AEs

- **Convolutional AEs (CAEs)**
 - AEs puros tendem a não funcionar bem com imagens (a menos que as imagens sejam pequenas)
 - CNNs são nitidamente superiores para tarefas com imagens
 - **Encoder** é uma **CNN comum**, composta de camadas convolucionais e pooling
 - reduz a dimensionalidade espacial dos dados mas aumenta a profundidade (filtros)
 - **Decoder** faz o inverso
 - aumenta os dados, e reduz a profundidade
 - usar camadas **convolucionais transpostas**

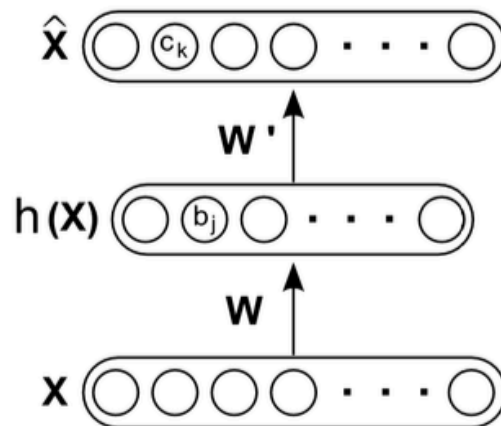
Outros modelos de AEs

- **AEs Recorrentes (RAEs)**

- AEs para sequências: séries temporais ou textos
- RNN tem desempenho melhor do que DNN
- RAE:
 - **Encoder:** é uma rede *sequence-to-vector*, que comprime a sequência de entrada em apenas um único vetor
 - **Decoder:** é uma rede *vector-to-sequence*, que reconstrói a sequência de entrada a partir da representação intermediária (vector)

Camada subcompleta

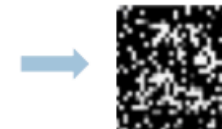
- AE subcompleto \rightarrow camada oculta é **menor** que a camada de entrada
 - comprime as entradas
 - comprime bem apenas para as entradas do conjunto de treinamento



Aprende bem a distribuição dos dados de treino

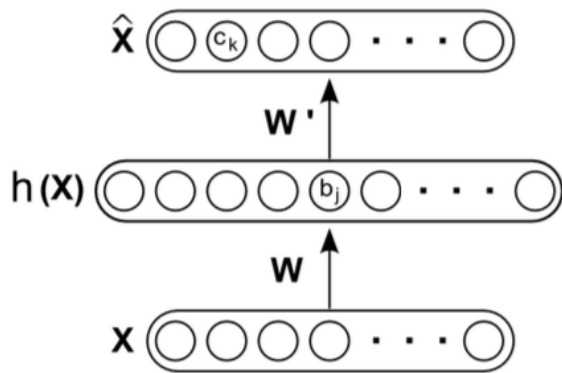


Não generaliza bem para dados diferentes



Camada sobrecompleta

- AE sobrecompleto \rightarrow camada oculta é **maior** que a camada de entrada



- não há compressão na camada oculta
- cada unidade oculta pode apenas copiar um diferente componente da entrada
- não há garantia de que as unidades ocultas irão extrair uma estrutura que tem significado

Roteiro

- 1 Introdução
- 2 AEs
- 3 Stacked AEs (SAEs)
- 4 *Denoising AEs (dAEs)*
- 5 Referências

Denoising AEs (dAEs)

□ **Ideia:**

- Adicionar ruídos às entradas
- treinar o AE para recuperar os dados originais, livre de ruídos
- ruído gerado pode ser Gaussiano ou aleatório (dropout)

Denoising AEs (dAEs)

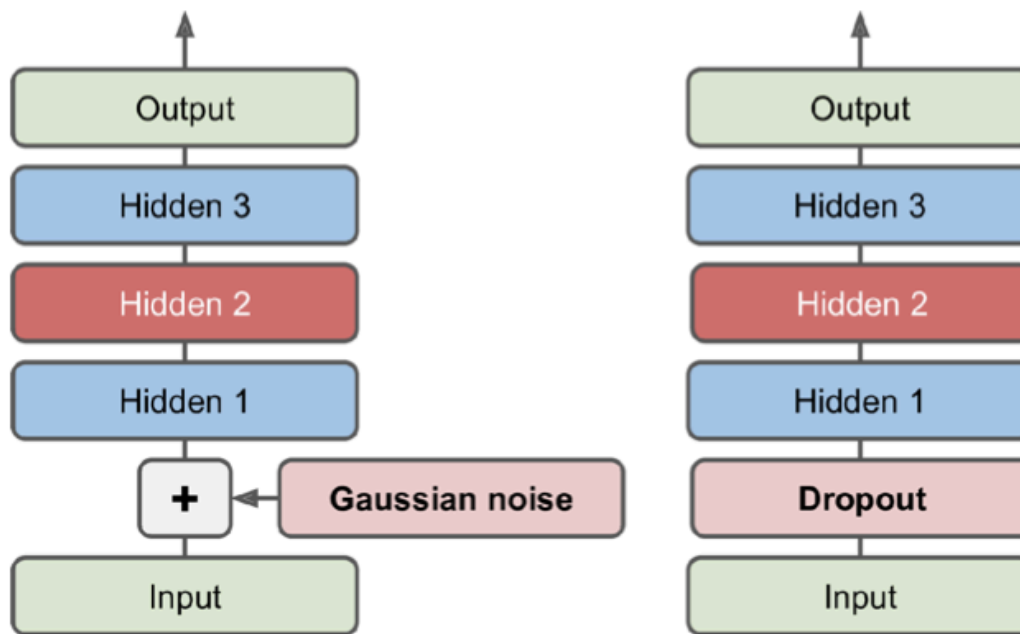


Figura de: Aurélien Géron (2019)

Denoising AEs (dAEs)

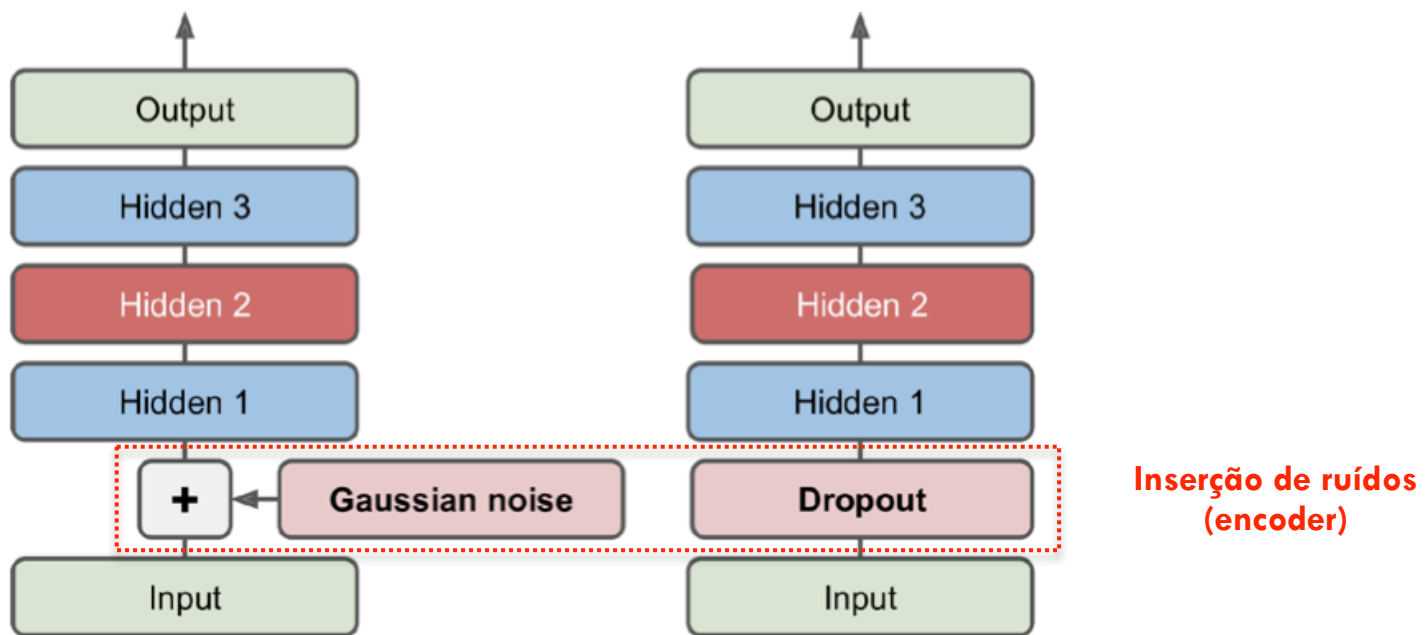
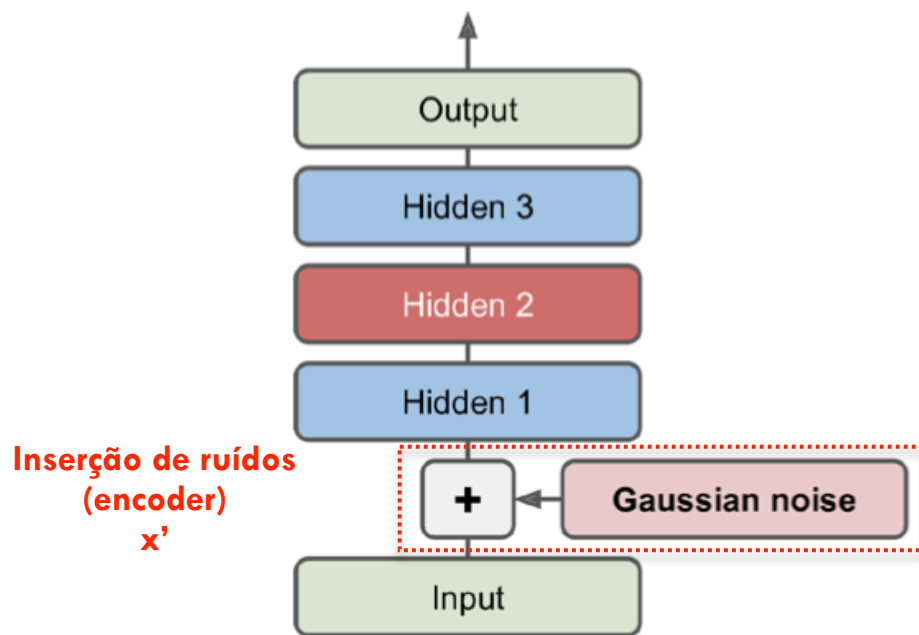


Figura de: Aurélien Géron (2019)

Denoising AEs (dAEs)



- representação deve ser **robusta** à introdução de **ruídos** nos padrões
- a **reconstrução** é calculada com base na **entrada corrompida** (x')
- função de **erro** é a mesma, usando os valores das reconstruções (**outputs**) e entradas originais (**inputs**)

Figura de: Aurélien Géron (2019)

Denoising AEs (dAEs)

- A implementação é direta/simples
 - SAE com uma camada adicional de Dropout ou ruído ligada às entradas
 - ruídos são aplicados apenas no encoder
 - dAEs podem ser usados para remover ruídos de dados em geral (textos, imagens, etc)

Denoising AEs (dAEs)

- A implementação é direta/simples
 - SAE com uma camada adicional de Dropout ou ruído ligada às entradas
 - ruídos são aplicados apenas no encoder
 - dAEs podem ser usados para remover ruídos de dados em geral (textos, imagens, etc)



Entradas com ruídos

Reconstruções (saídas)

Figura de: Aurélien Gerón (2019)

Hands on

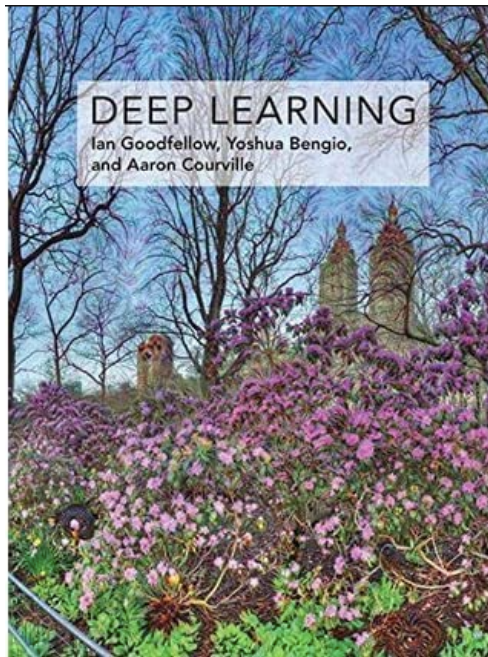


Vamos exercitar :)
[[Google Colab](#)]

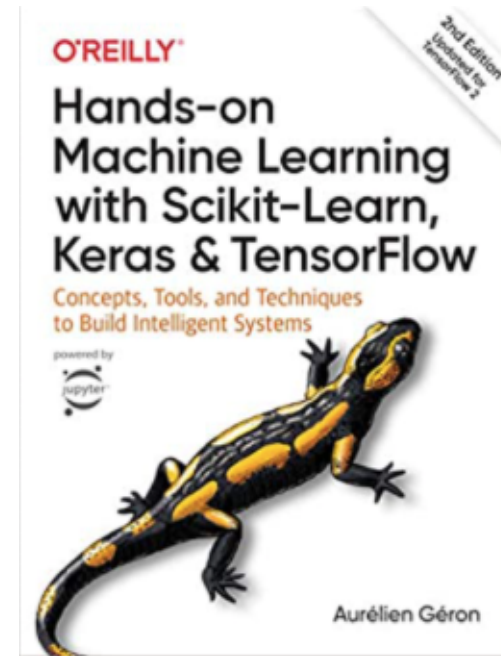
Roteiro

- 1 Introdução
- 2 AEs
- 3 Stacked AEs (SAEs)
- 4 *Denoising AEs (dAEs)*
- 5 Referências

Literatura Sugerida



(Goodfellow, Bengio, Courville; 2015)



(Géron, 2019)

Artigos



- William G. Chase & Herbert A. Simon, “Perception in chess”, Cognitive Psychology v4, no. 1 (1973): 55-81.



Obrigado :)

Prof. Rafael G. Mantovani
rafaelmantovani@utfpr.edu.br