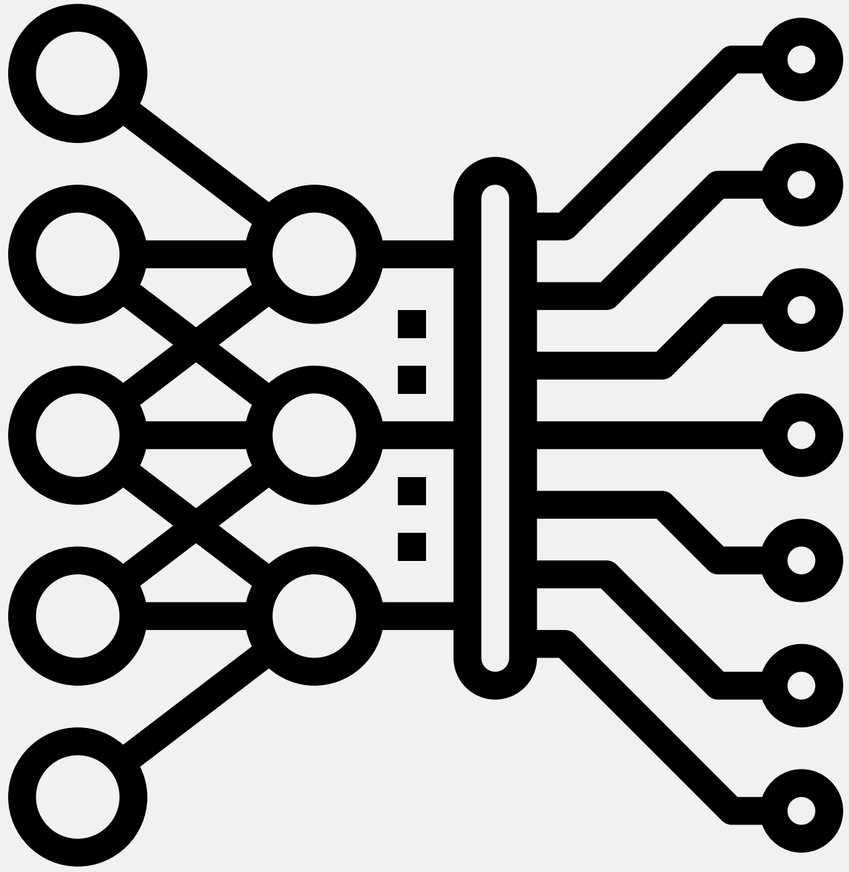
INTELIGÊNCIA COMPUTACIONAL - 2021.2

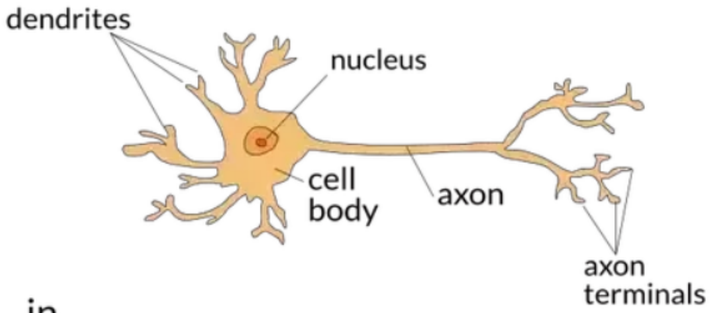
Exercício 7: Redes Neurais

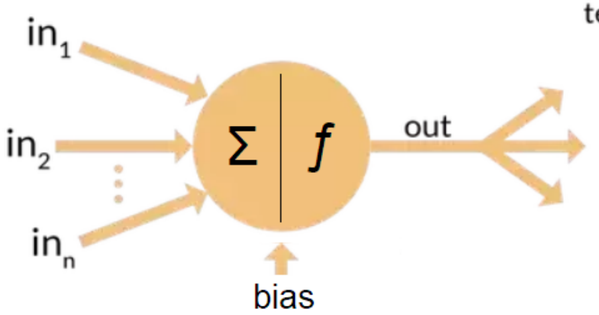
Henrique Chaves - DRE: 119025571 (Grupo 1)
Pedro Boechat - DRE: 119065050 (Grupo 1)

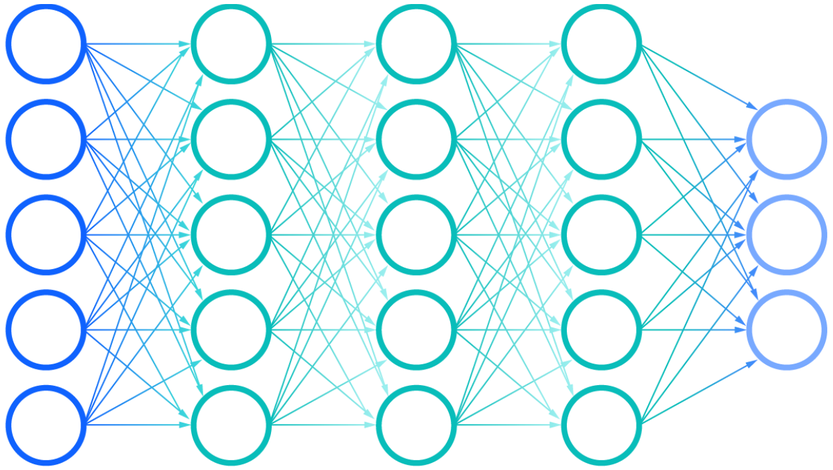
Ana Carolina Jaolino - DRE: 117230950 (Grupo 15)
Lucas Storino - DRE: 119053231 (Grupo 15)



Definição







Inspirada nos neurônios biológicos

Formadas por 4 componentes básicos

Demanda de grande desempenho computacional

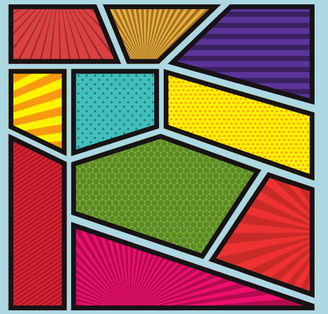
Backbone do Deep Learning

Pode ser usado em diferentes problemas

Algumas aplicações

Descobrimento de novas drogas

Previsões em séries temporais



Criação artística

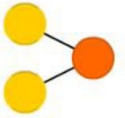
Outros: visão computacional, NLP, reconhecimento de voz, etc...

A mostly complete chart of

Neural Networks

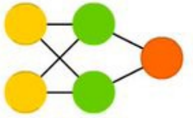
- Backfed Input Cell
- Input Cell
- Noisy Input Cell
- Hidden Cell
- Probablistic Hidden Cell
- Spiking Hidden Cell
- Output Cell
- Match Input Output Cell
- Recurrent Cell
- Memory Cell
- Different Memory Cell
- Kernel
- Convolution or Pool

Perceptron (P)

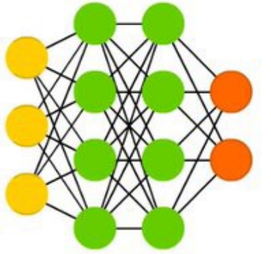


Feed Forward (FF)

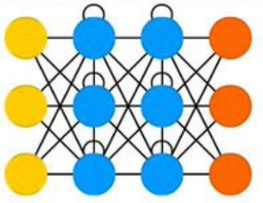
Radial Basis Network (RBF)



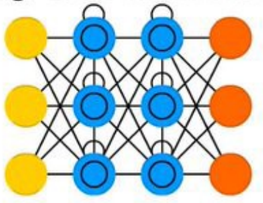
Deep Feed Forward (DFF)



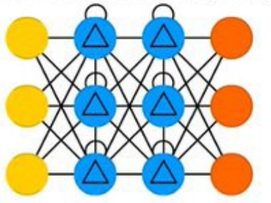
Recurrent Neural Network (RNN)



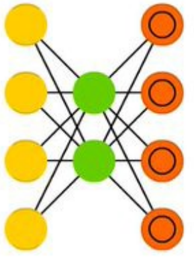
Long / Short Term Memory (LSTM)



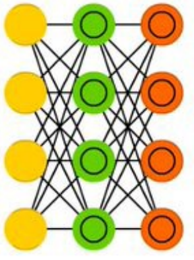
Gated Recurrent Unit (GRU)



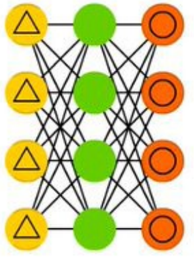
Auto Encoder (AE)



Variational AE (VAE)



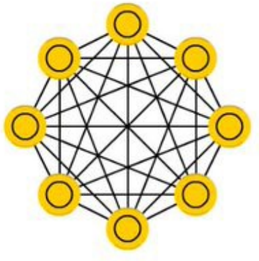
Denoising AE (DAE)



Sparse AE (SAE)

Markov Chain (MC)

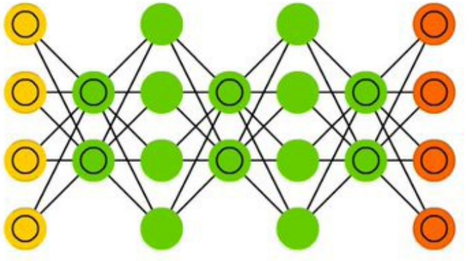
Hopfield Network (HN)



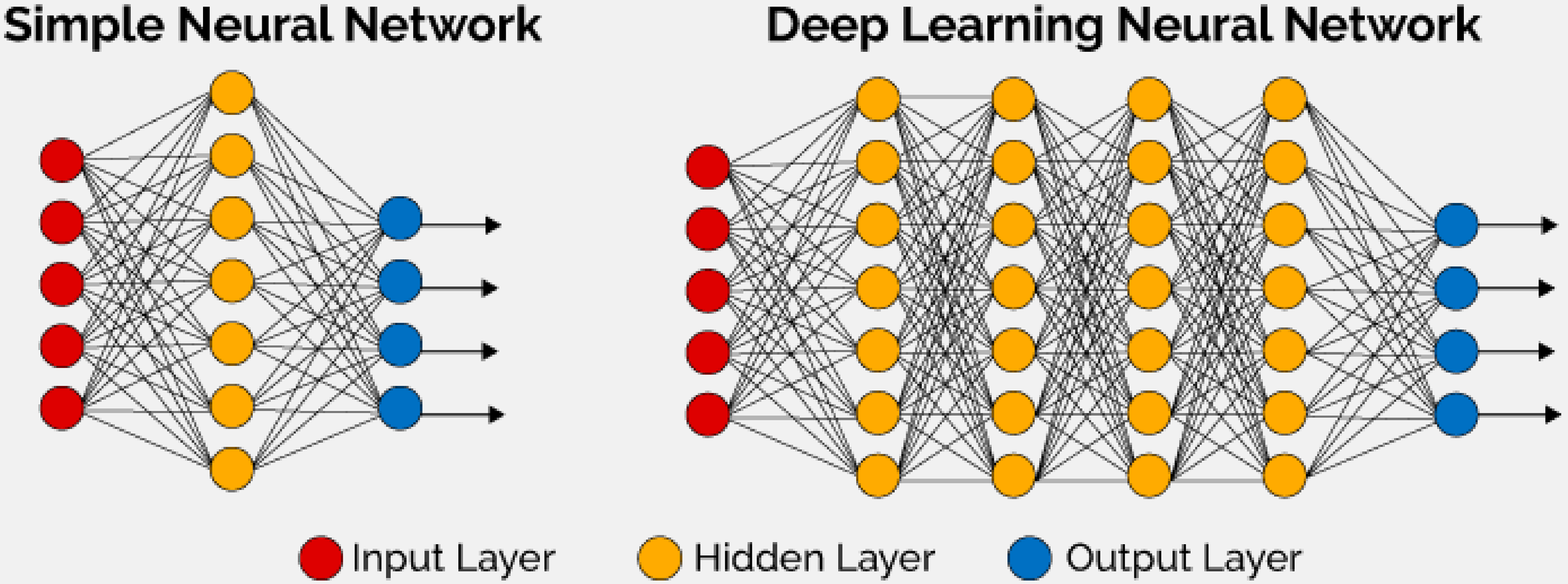
Boltzmann Machine (BM)

Restricted BM (RBM)

Deep Belief Network (DBN)



Simple NN vs Deep Learning NN

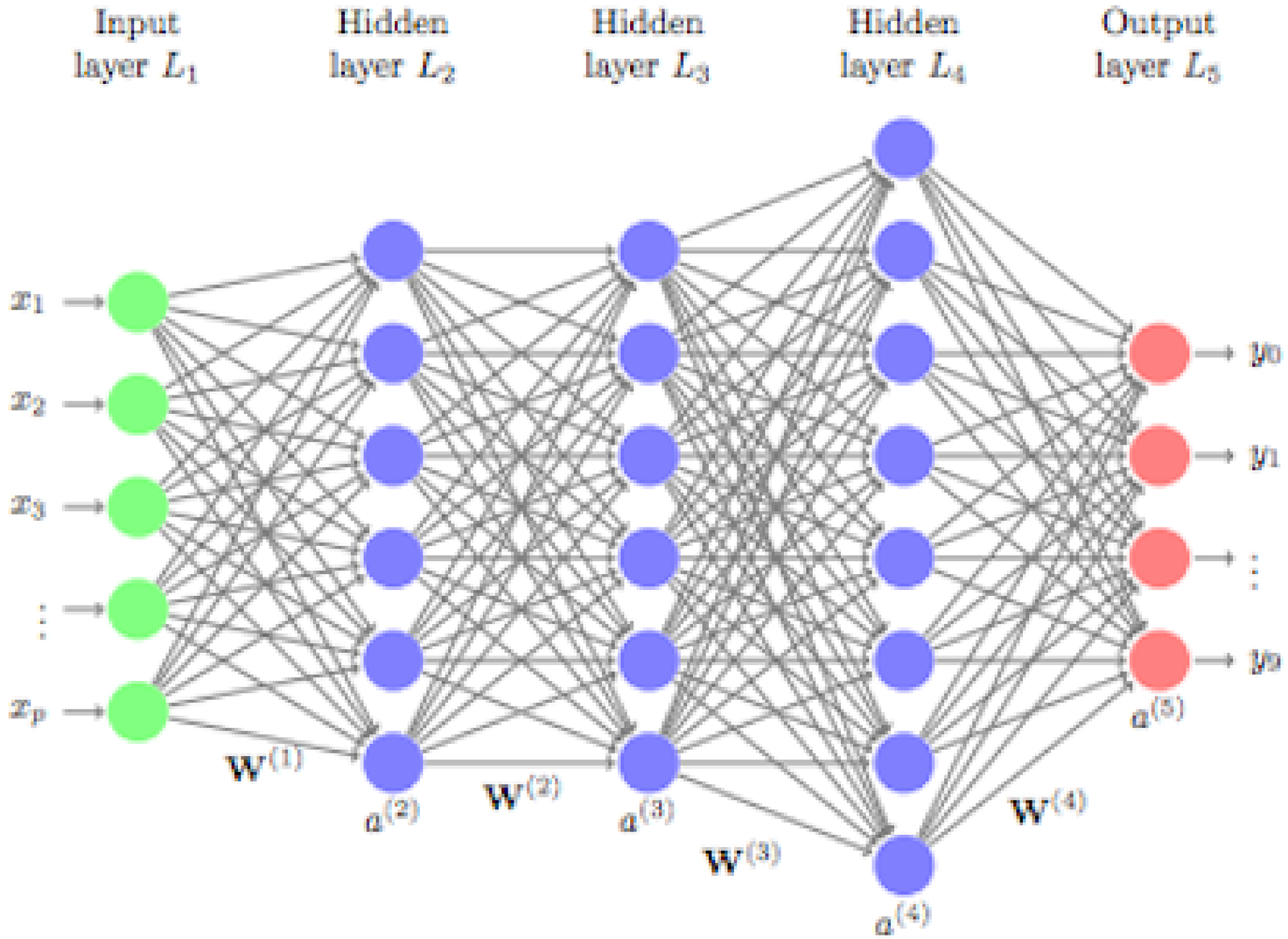


Feedforward Neural Net (FFNN)

Útil para problemas de classificação e regressão

Pode substituir a maioria dos algoritmos tradicionais de ML

Treinamento rápido, e portanto é o modelo mais comum de ser encontrado



Treinamento Feedforward

1) Inicialização dos pesos entre as camadas

2) Colocar cada variável de entrada nos neurônios de entrada

3) Forward-propagation

4) Comparar a previsão com o valor real e medir o erro

5) Backpropagation

6) Atualização do pesos

7) Finaliza uma época

Inicialiazação de pesos

Inicialiazação normal

$$w \approx N(0, \sigma)$$

Inicialiazação de pesos

Inicialiazação Xavier Glorot
(uniforme)

$$w \approx \mathcal{U}\left[-\sqrt{\frac{6}{\text{in}+\text{out}}}, \sqrt{\frac{6}{\text{in}+\text{out}}} \right]$$

Inicialiazação de pesos

Inicialiazação He
(uniforme)

$$w \approx \mathcal{U}\left[-\sqrt{\frac{6}{\text{in}}}, \sqrt{\frac{6}{\text{in}}} \right]$$

Treinamento Feedforward

1) Inicialização dos pesos entre as camadas

2) Colocar cada variável de entrada nos neurônios de entrada

3) Forward-propagation

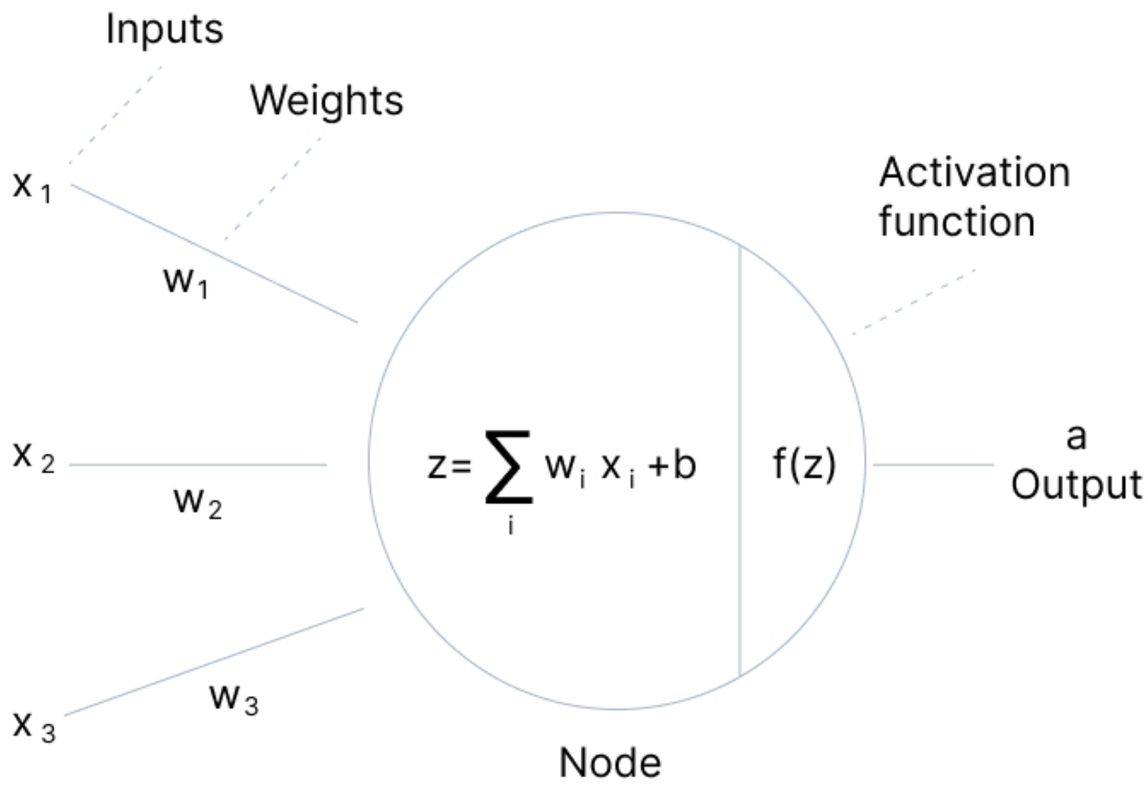
4) Comparar a previsão com o valor real e medir o erro

5) Backpropagation

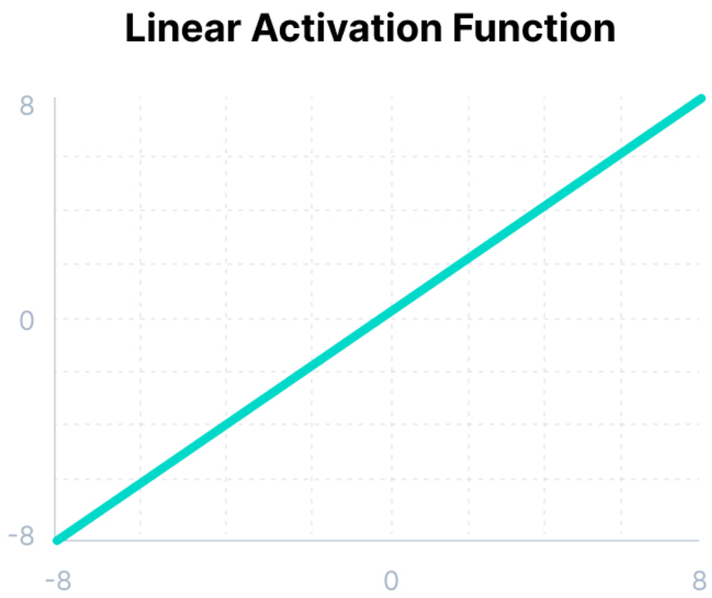
6) Atualização do pesos

7) Finaliza uma época

Função de ativação

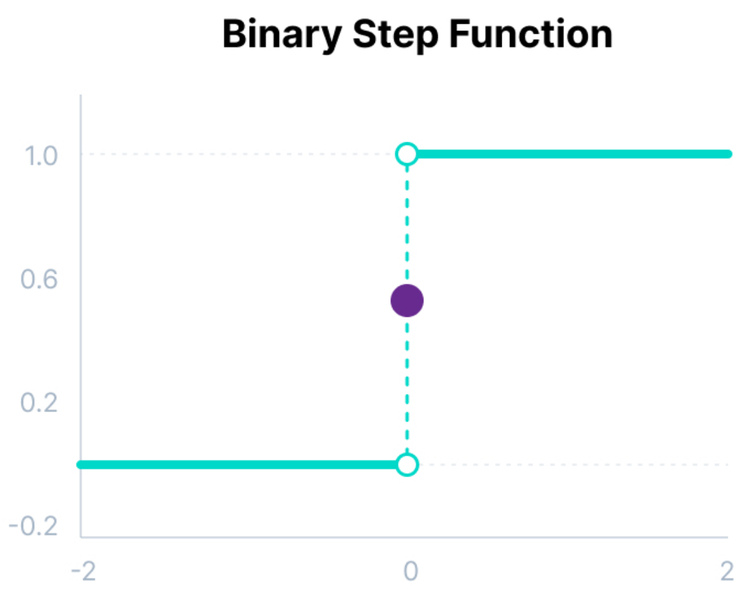


Funções de ativação lineares



Linear

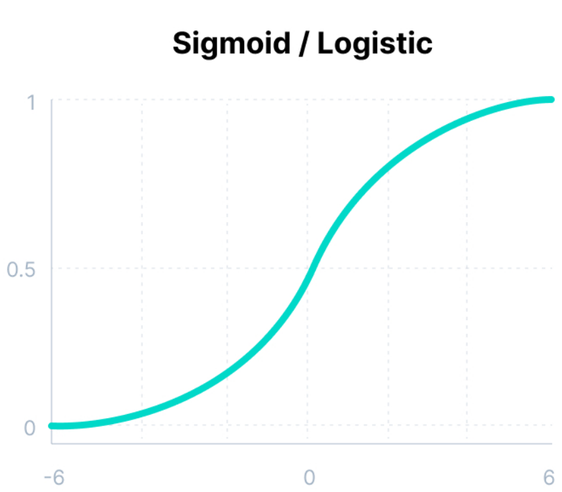
$$f(x) = x$$

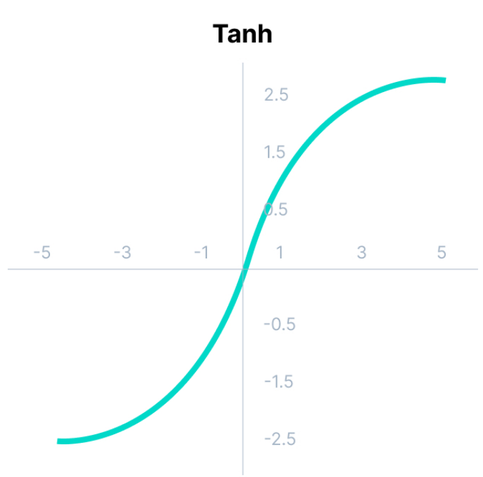


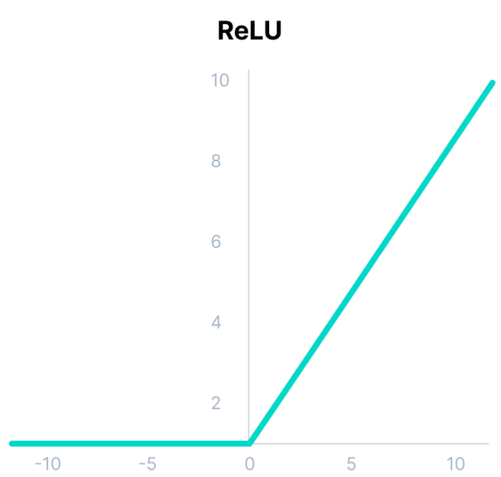
Binary step

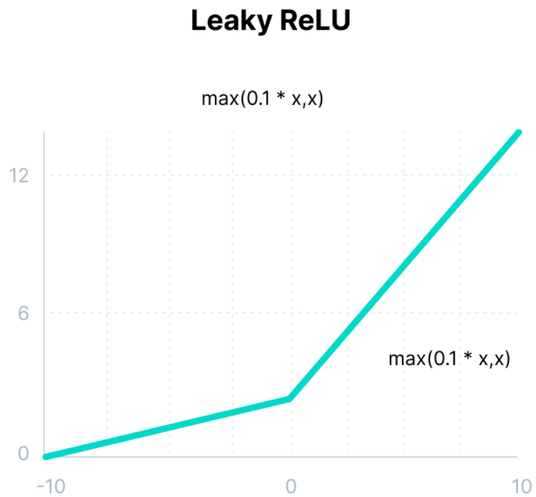
$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

Funções de ativação não-lineares









Sigmoid / Logistic

$$f(x) = \frac{1}{1 + e^{-x}}$$

Tanh

$$f(x) = \frac{\left(e^{x} - e^{-x}\right)}{\left(e^{x} + e^{-x}\right)}$$

ReLU

$$f(x) = max\ (0, x)$$

Leaky ReLU

$$f(x) = max\ (0.1x, x)$$

Outras: Parametric ReLU, ELU, Softmax, Swish, GELU, SELU...

Treinamento Feedforward

1) Inicialização dos pesos entre as camadas

2) Colocar cada variável de entrada nos neurônios de entrada

3) Forward-propagation

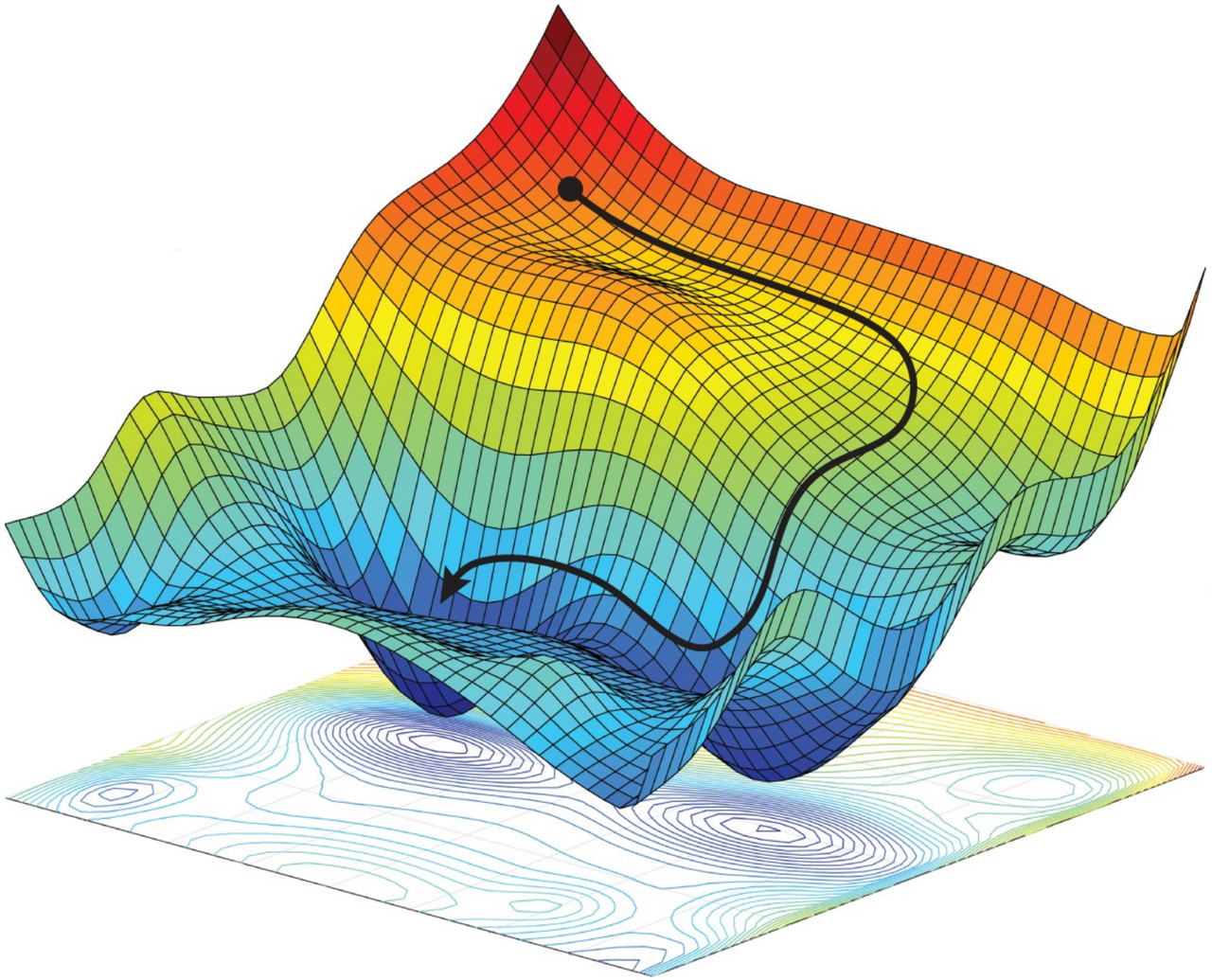
4) Comparar a previsão com o valor real e medir o erro

5) Backpropagation

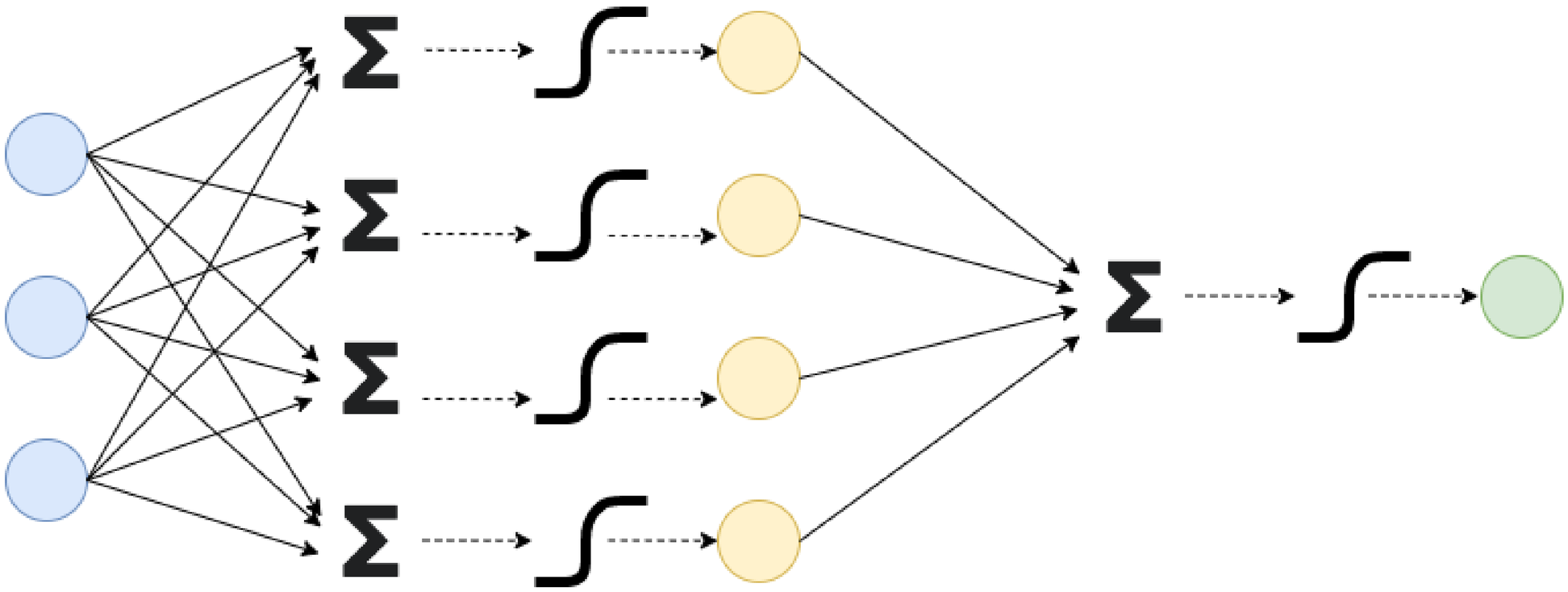
6) Atualização do pesos

7) Finaliza uma época

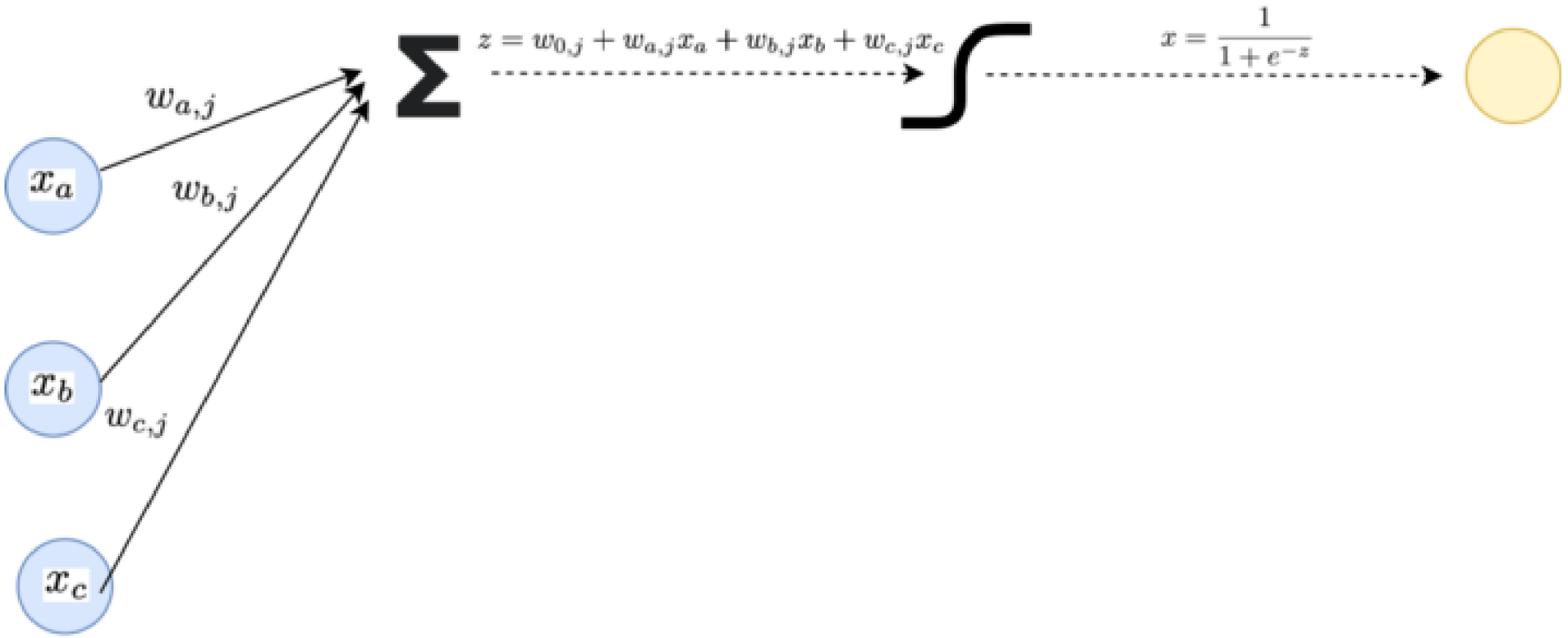
Backpropagation



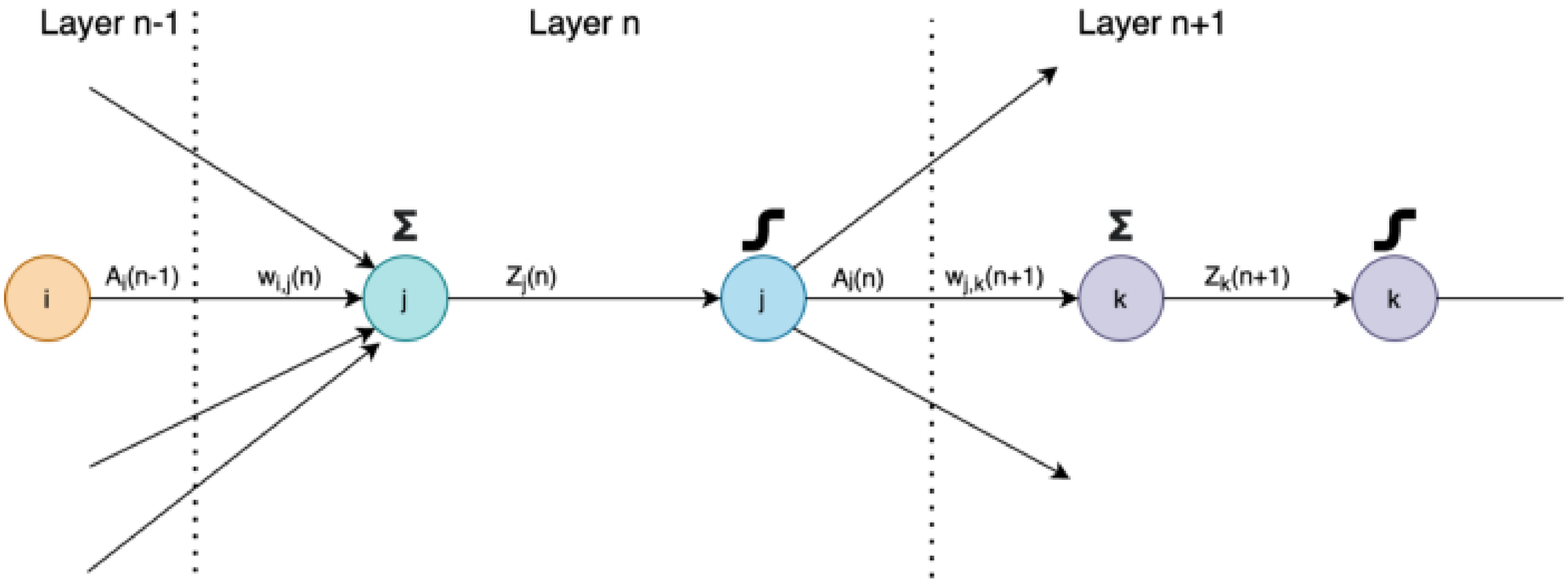
Backpropagation



Backpropagation



Backpropagation



$$w_{i,j} \leftarrow w_{i,j} - a\frac{\partial E(D, \mathbf{w})}{\partial w_{i,j}}$$

Treinamento Feedforward

1) Inicialização dos pesos entre as camadas

2) Colocar cada variável de entrada nos neurônios de entrada

3) Forward-propagation

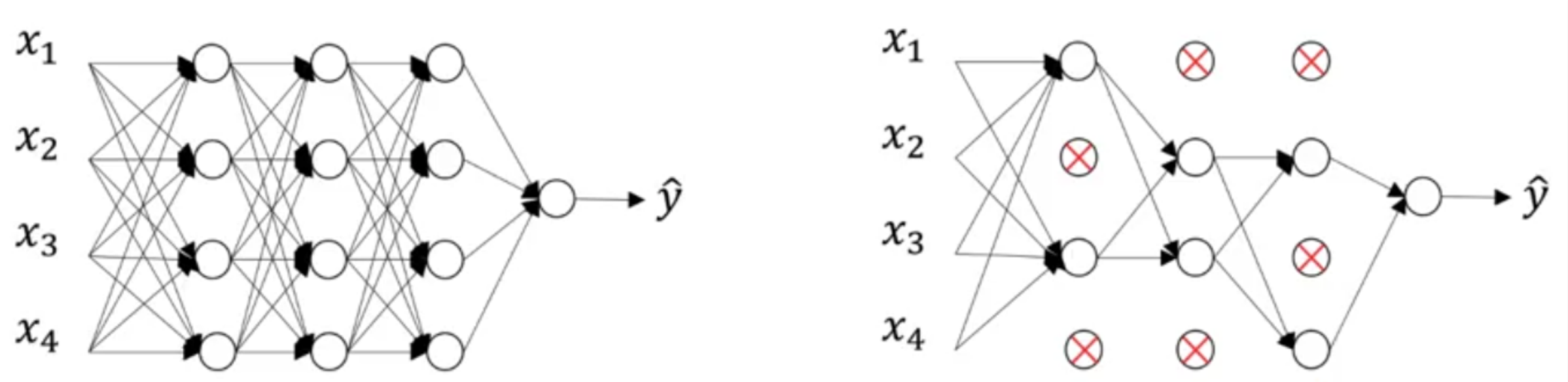
4) Comparar a previsão com o valor real e medir o erro

5) Backpropagation

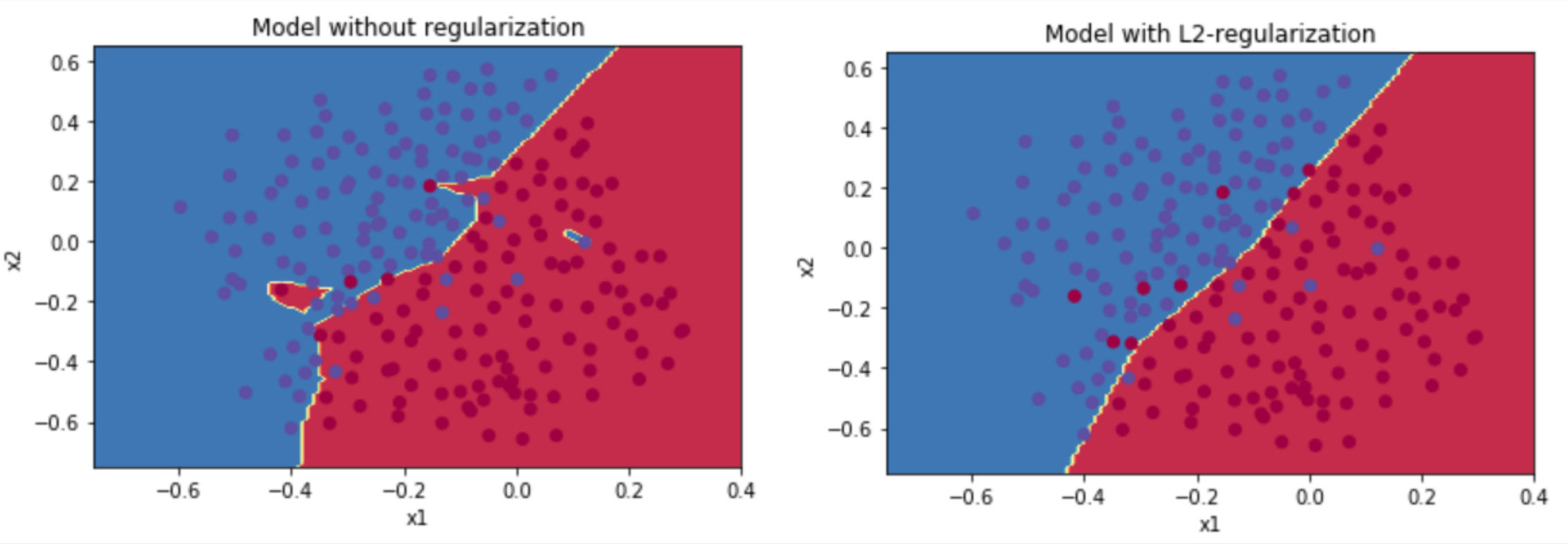
6) Atualização do pesos

7) Finaliza uma época

Regularizadores



Regularizadores



Diversos algoritmos

Além do feedforward

Recurrent Neural Net

Autoencoder

Convolutional Neural Net

Generative Adversarial Network

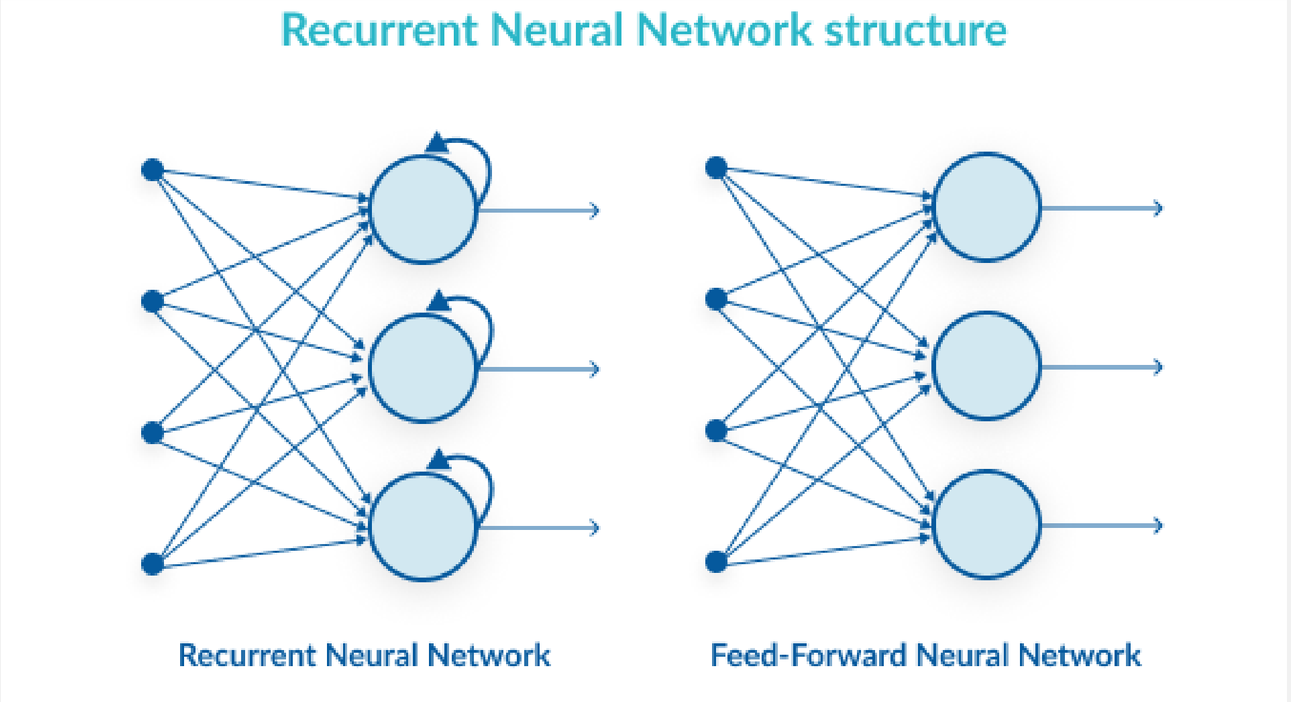
Entre outros....

Recurrent Neural Net (RNN)

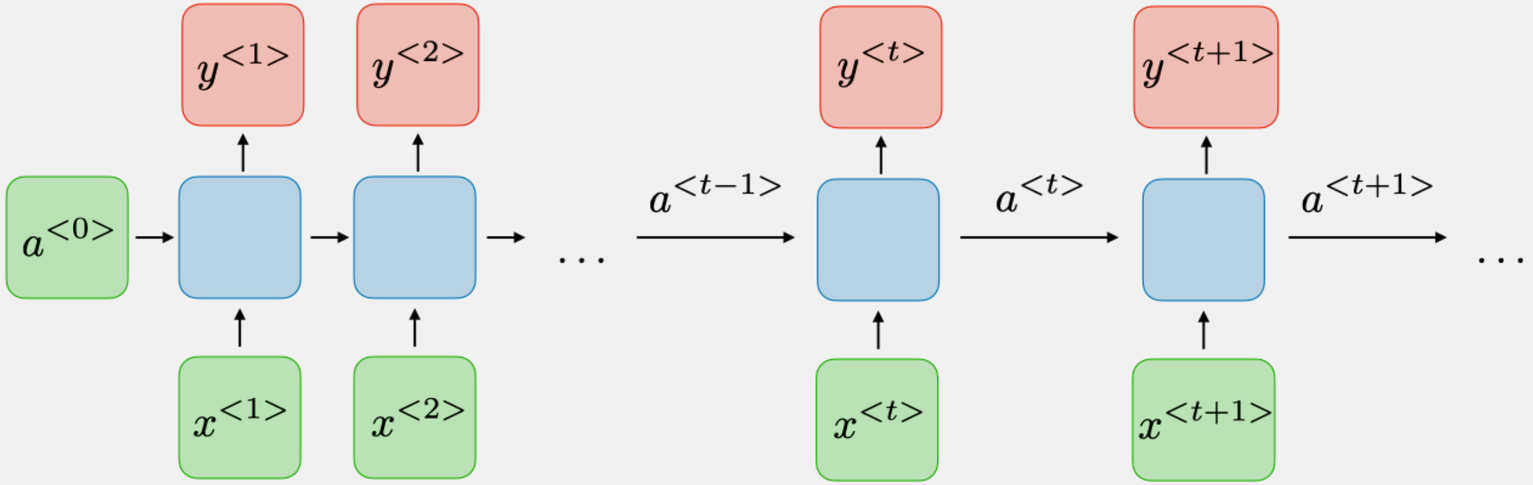
Sequências

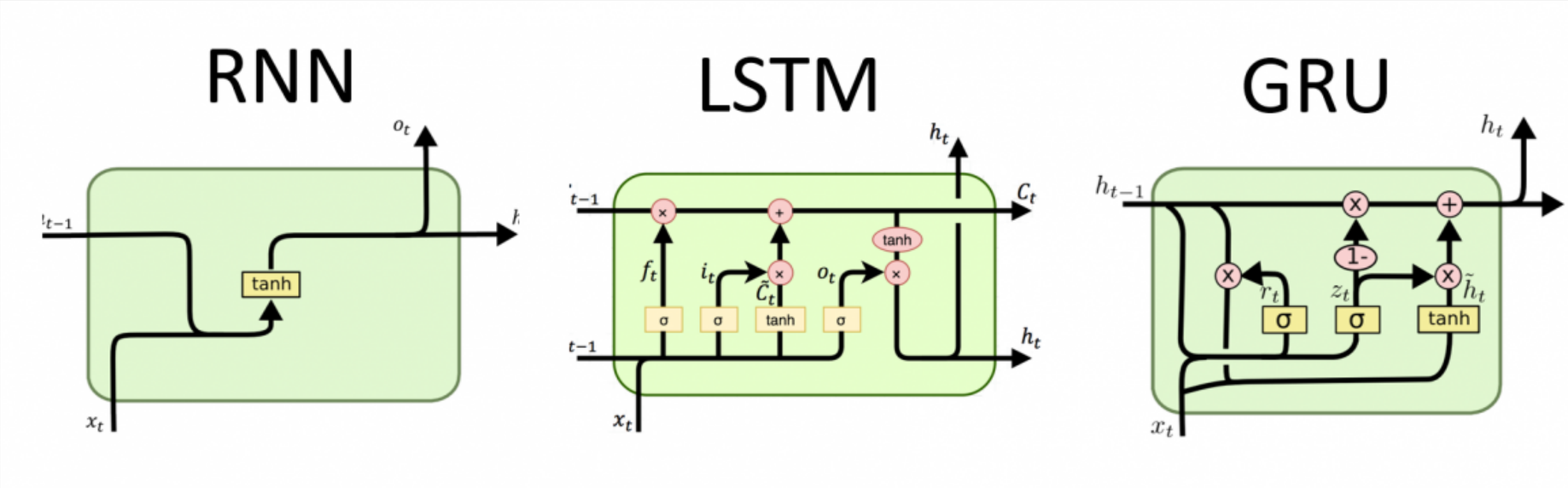
Série temporal

Treinamento mais lento e pesado

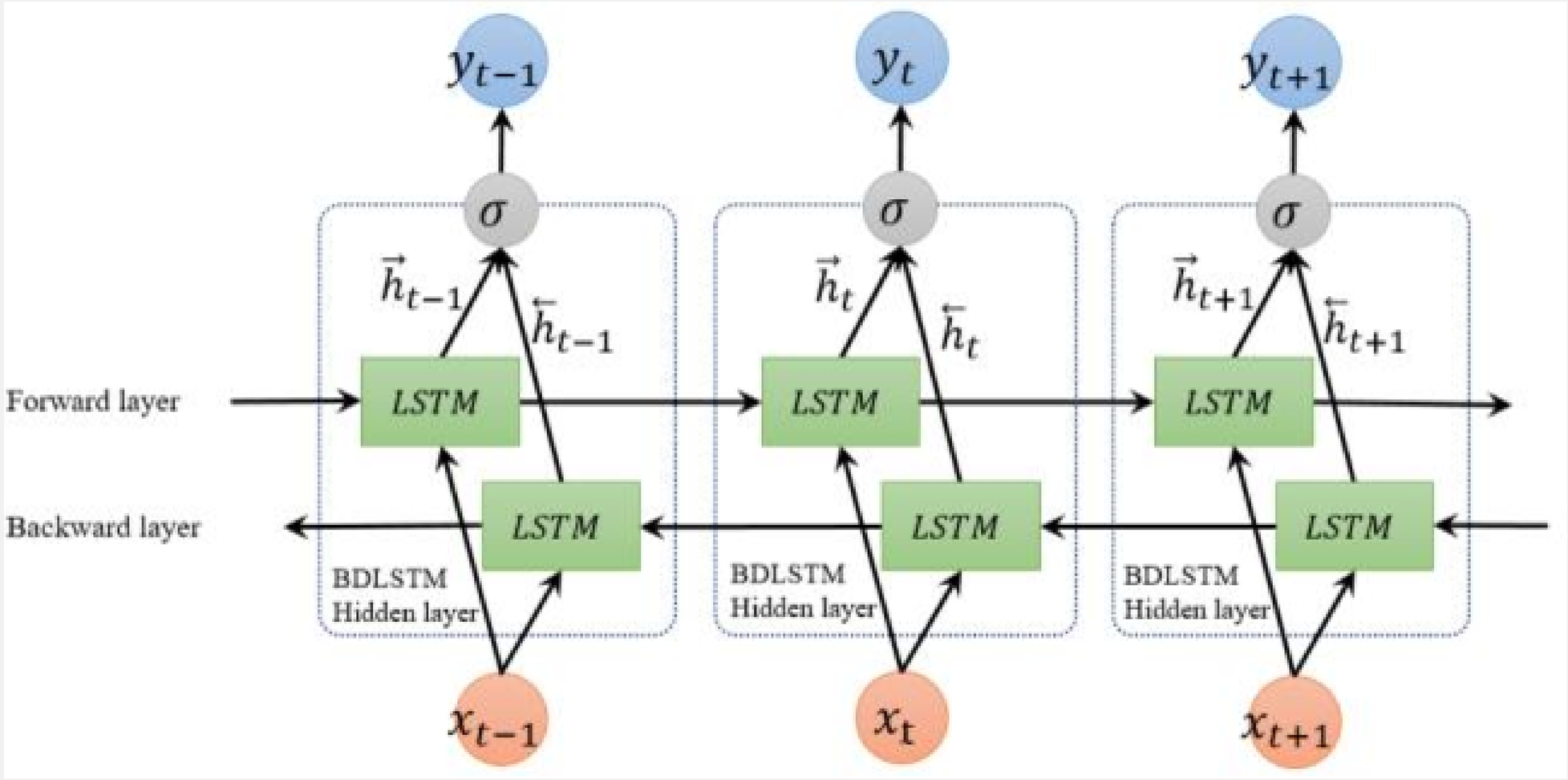


Recurrent Neural Net (RNN)





Recurrent Neural Net (RNN)

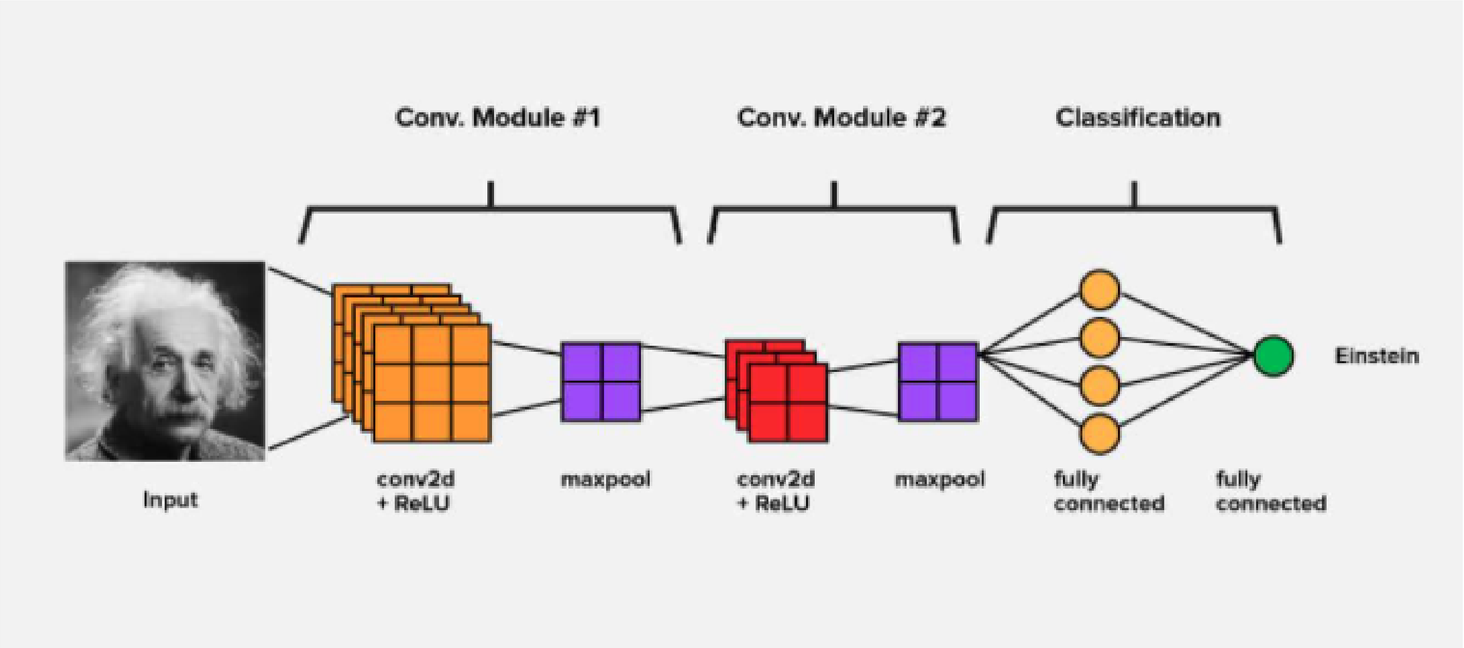


Convolutional Neural Net (CNN)

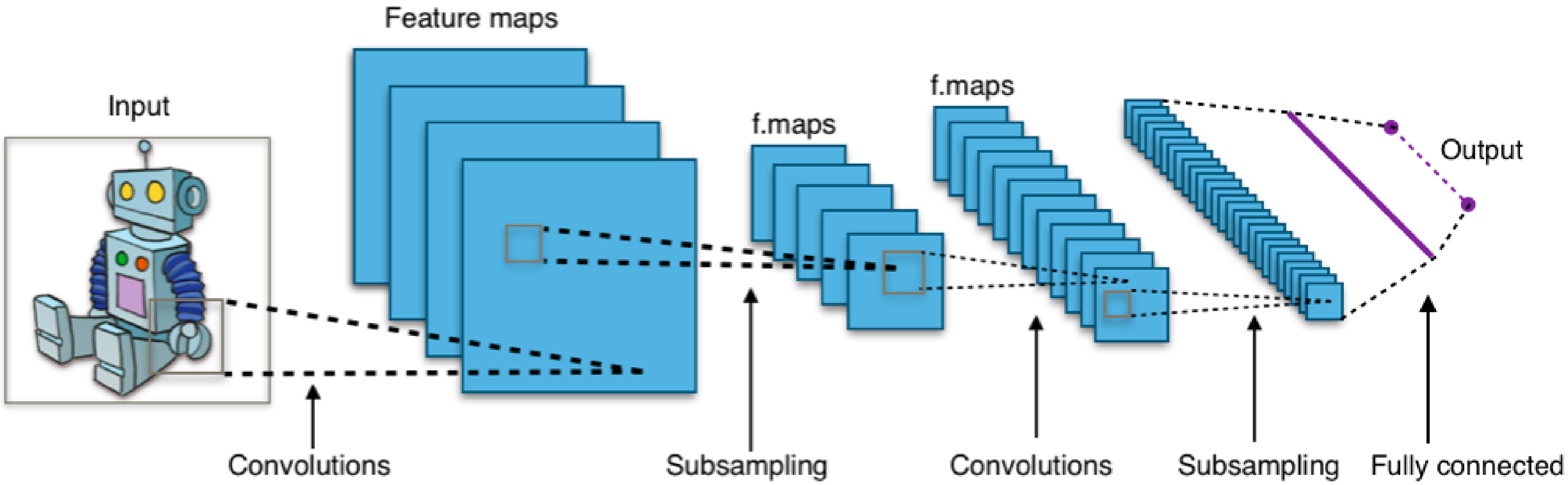
Usada para resolver a maioria dos problemas com Deep Learning

Feedforward ou recurrent

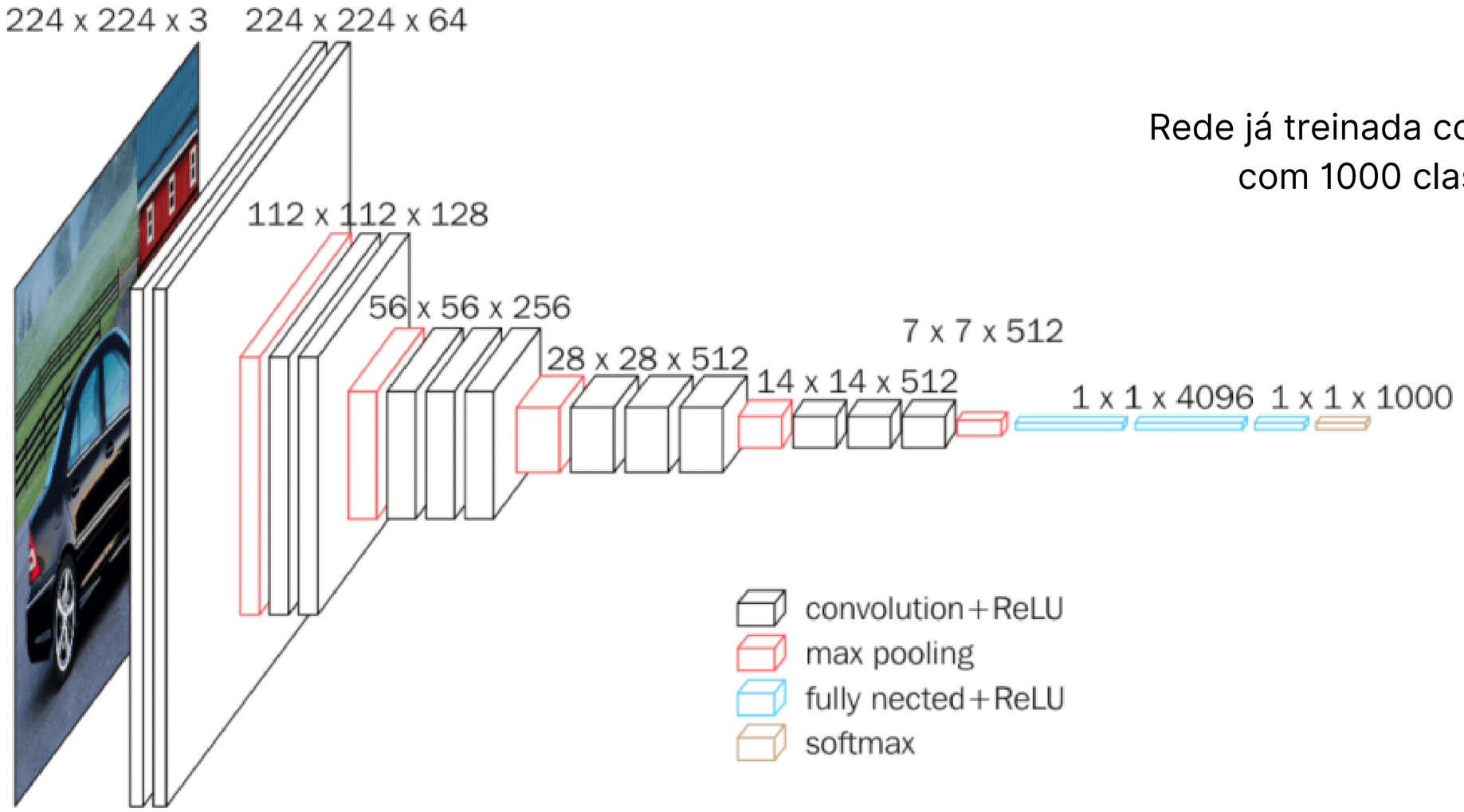
Principal arquitetura para trabalhar com imagens



Convolutional Neural Net (CNN)



Convolutional Neural Net (CNN)
Fine-tuning



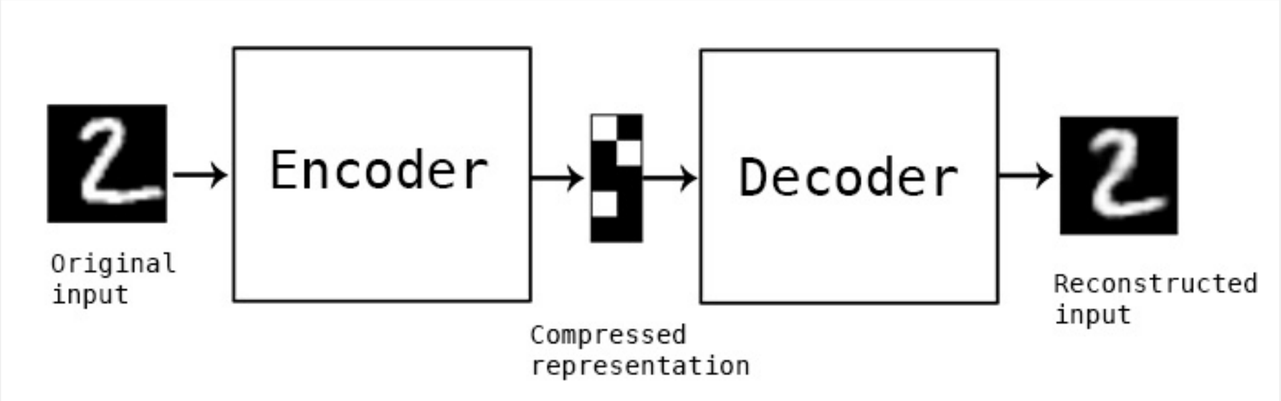
Rede já treinada com diversas imagens
com 1000 classes diferentes

Autoencoders

Compressão e Codificação

Decodificação mais próxima do *input original*

Redução de dimensionalidade ao aprender a ignorar o ruído dos dados



Autoencoders

$$\phi : \mathcal{X} \to \mathcal{F}$$
$$\psi : \mathcal{F} \to \mathcal{X}$$

$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$$

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

$$\mathcal{L}(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^2 = \|\mathbf{x} - \sigma'(\mathbf{W}'(\sigma(\mathbf{W}\mathbf{x} + \mathbf{b})) + \mathbf{b}')\|^2$$

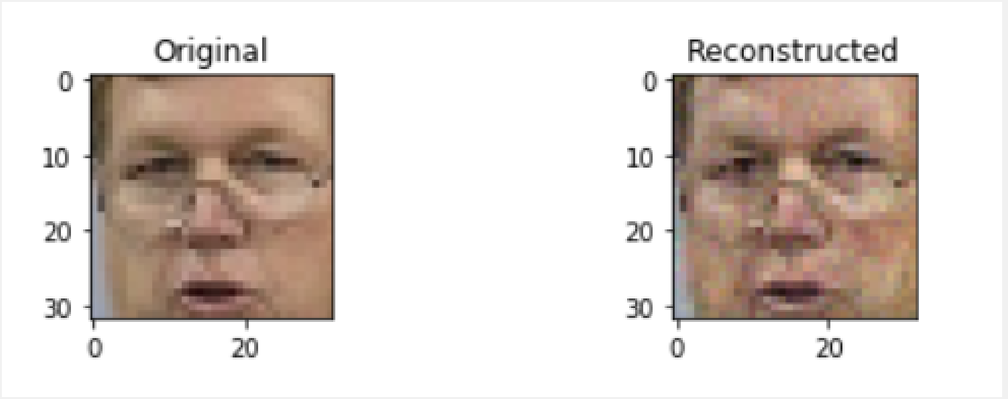
Autoencoders

```python
from keras.layers import Dense, Flatten, Reshape, Input, InputLayer
from keras.models import Sequential, Model

def build_autoencoder(img_shape, code_size):
    # The encoder
    encoder = Sequential()
    encoder.add(InputLayer(img_shape))
    encoder.add(Flatten())
    encoder.add(Dense(code_size))

    # The decoder
    decoder = Sequential()
    decoder.add(InputLayer((code_size,)))
    decoder.add(Dense(np.prod(img_shape)))
    decoder.add(Reshape(img_shape))

    return encoder, decoder
```

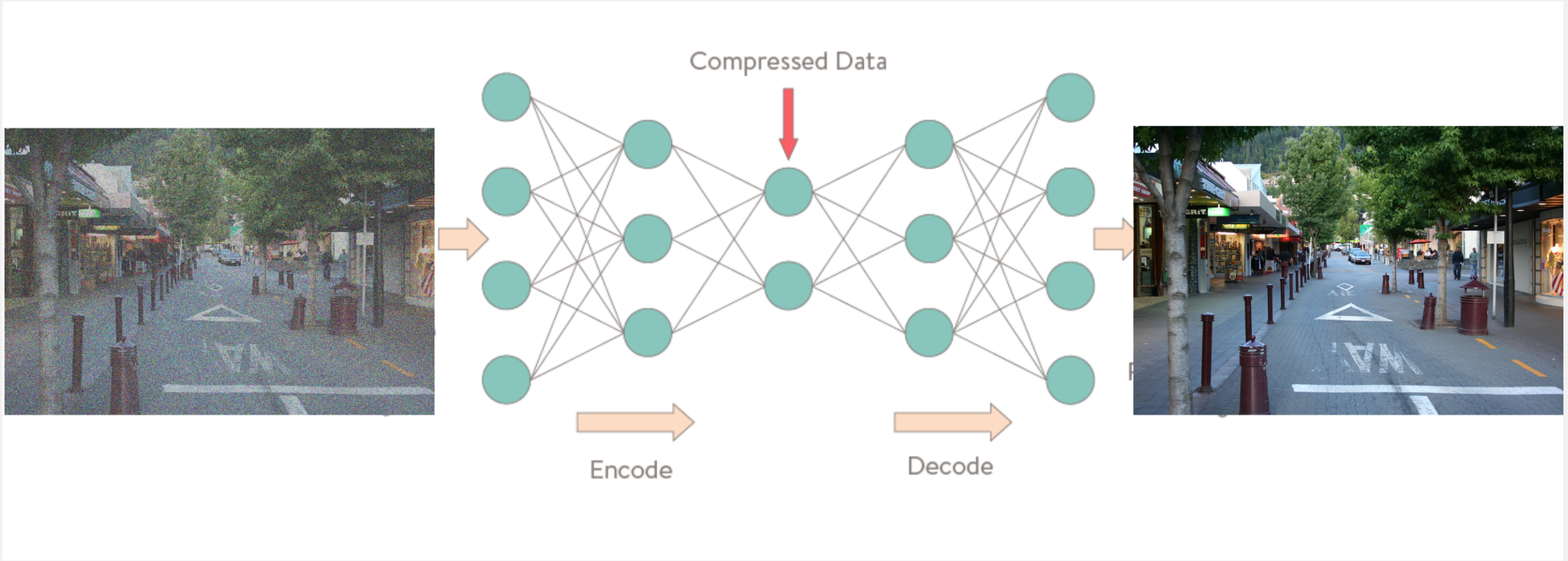



```python
from sklearn.model_selection import train_test_split
X_train, X_test = train_test_split(X, test_size=0.1, random_state=42)

history = autoencoder.fit(x=X_train, y=X_train, epochs=5,
                validation_data=[X_test, X_test])
```

Autoencoders

Exemplo - Reduzir ruídos de imagens (*Image Denoising)*



Generative adversarial network

Dado um set de treino, aprende a gerar dados novos similares ao set de treino

Duas redes neuras competem entre si em uma espécie de *jogo de soma zero*

Uso: GAN pode gerar novas fotos que se parecem autênticas a observadores humanos. *Deepfakes*

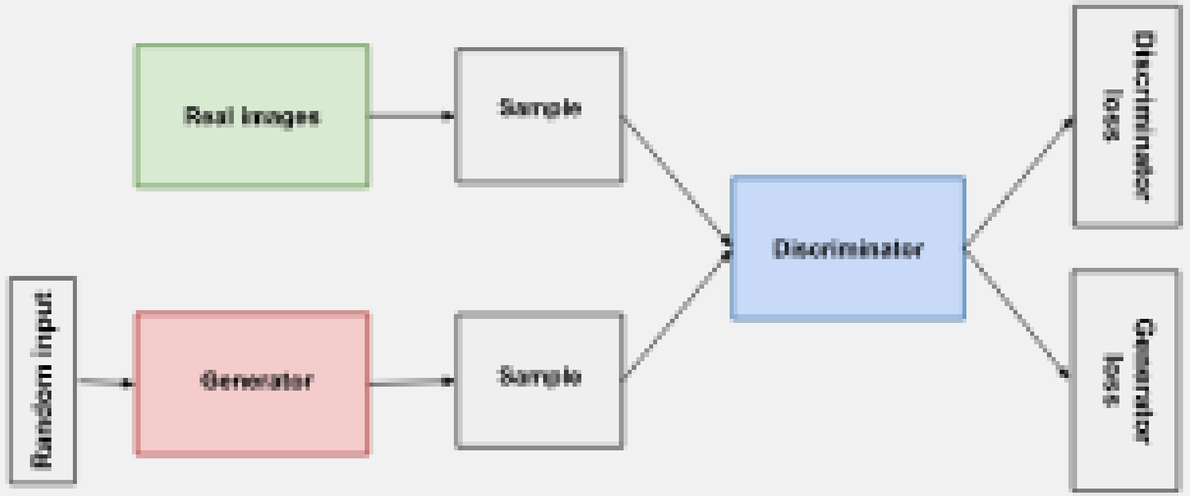
Essa imagem foi gerada pelo StyleGAN baseado na análise de um retrato

Generative adversarial network

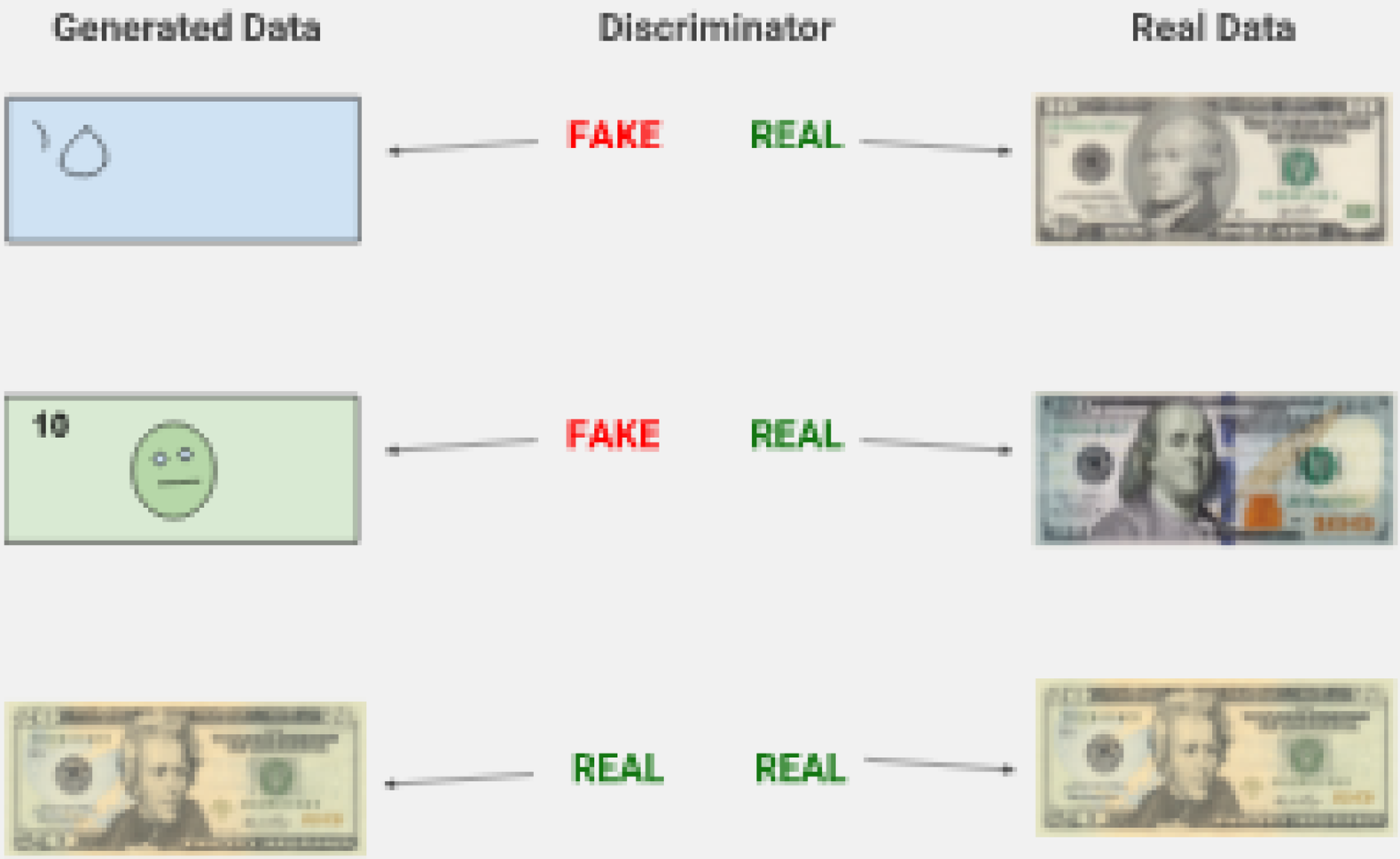
O GAN possui duas partes:

Gerador: aprende a gerar dados plausíveis. As instâncias passam a ser exemplos negativos de treino para o discriminador.

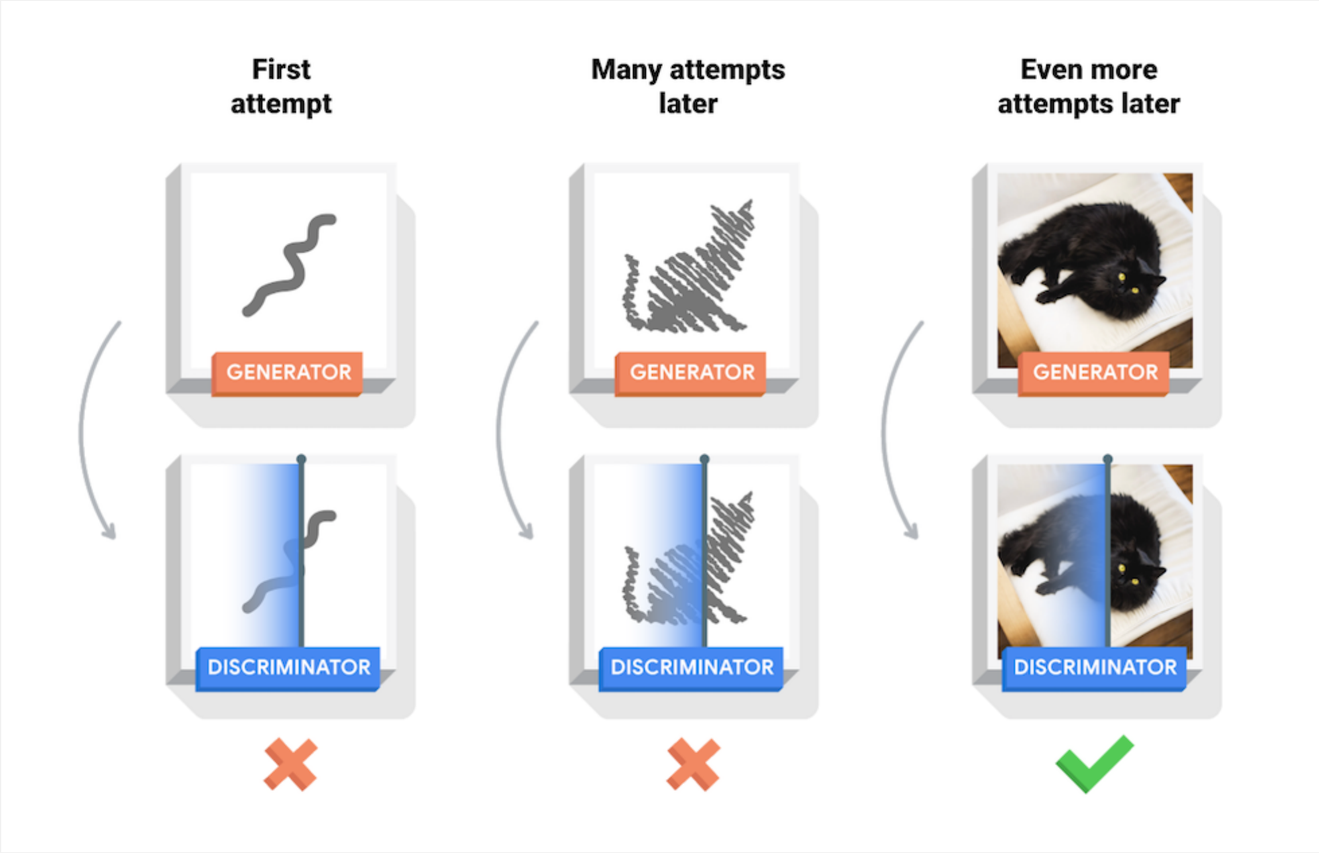
Discriminador: distingue o dado falso (advindo do gerador) do dado real. O discriminador penaliza o gerador de produzir dados não-plausíveis



Generative adversarial network



Generative adversarial network



Referências bibliográficas

Chollet, F. (2017). *Deep learning with python*. Manning Publications.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras and TensorFlow: concepts, tools, and techniques to build intelligent systems (2nd ed.)*. O'Reilly.

A Guide to Deep Learning and Neural Networks. (2021). Retrieved 14 February 2022, from https://serokell.io/blog/deep-learning-and-neural-network-guide

AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the Difference? (2020). Retrieved 14 February 2022, from https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks

12 Types of Neural Network Activation Functions: How to Choose? (2022). Retrieved 15 February 2022, from https://www.v7labs.com/blog/neural-networks-activation-functions