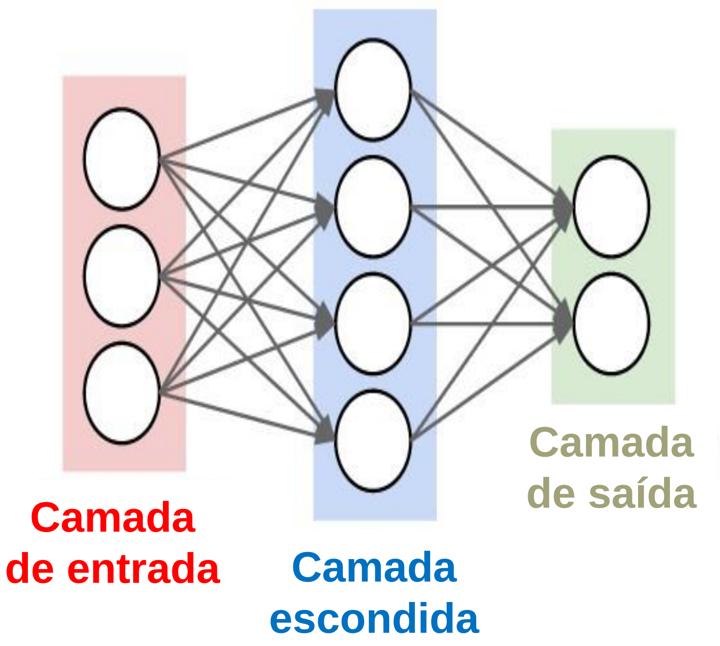
REDES NEURAIS ARTIFICIAIS

DETALHES DE ARQUITETURA / HISTÓRICO

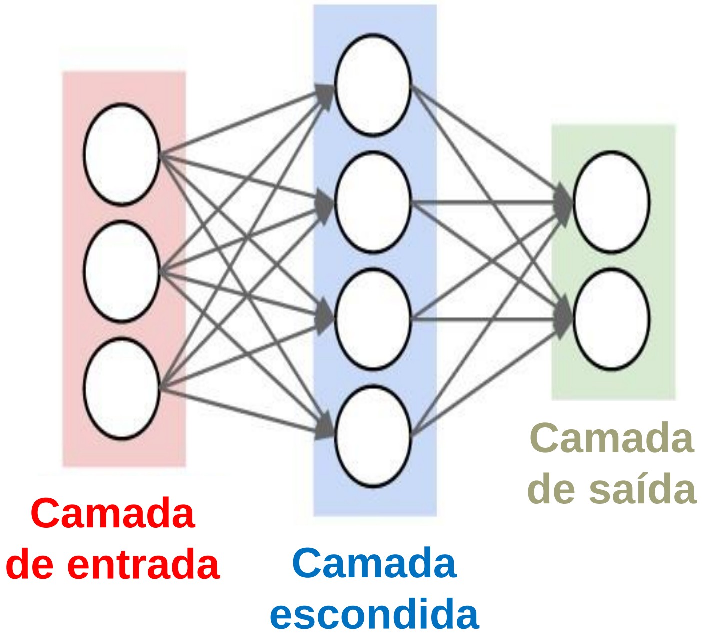
Zenilton K. G. Patrocínio Jr

zenilton@pucminas.br

Arquitetura de Rede Neural *Feed-Forward*

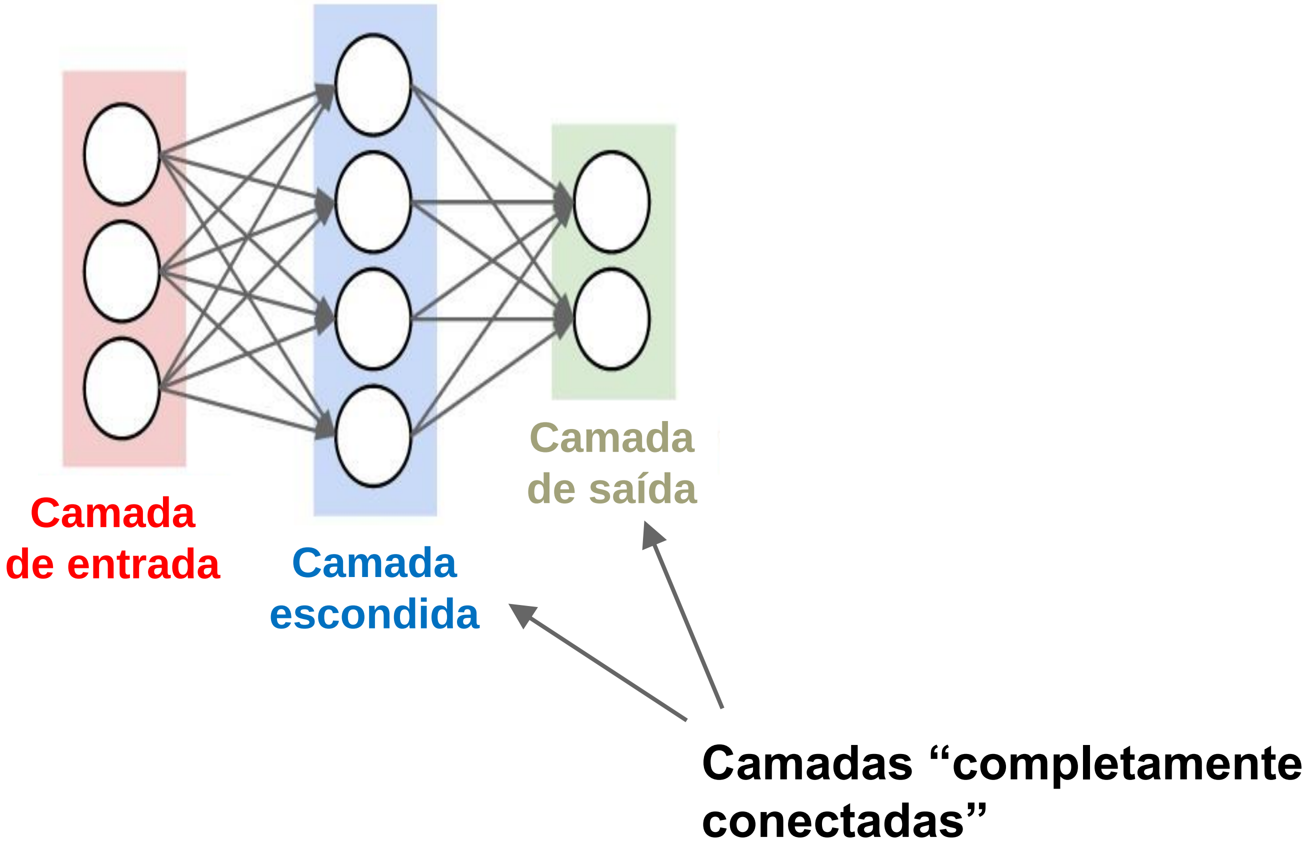


Arquitetura de Rede Neural *Feed-Forward*



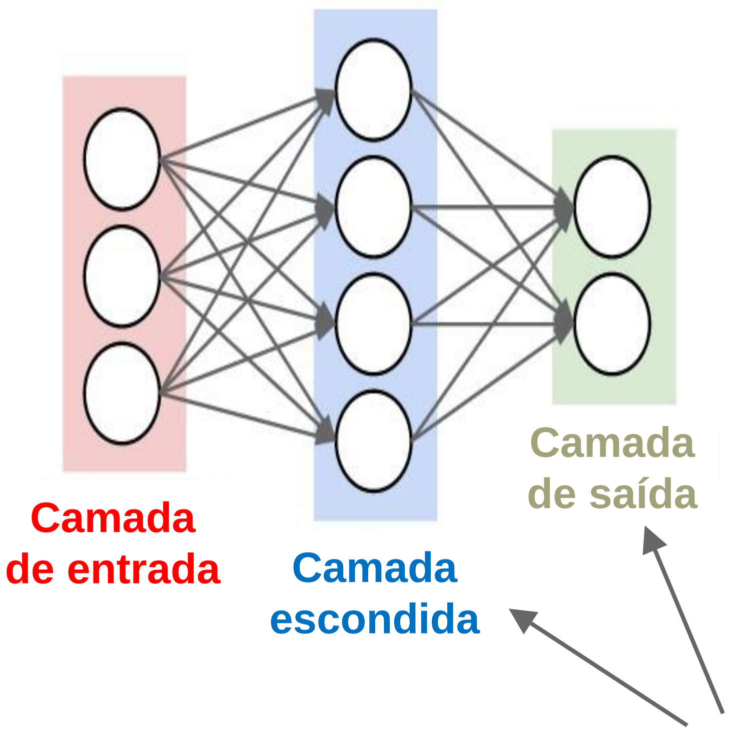
“Rede Neural de 2 camadas” ou
“Rede Neuras com 1 camada escondida”

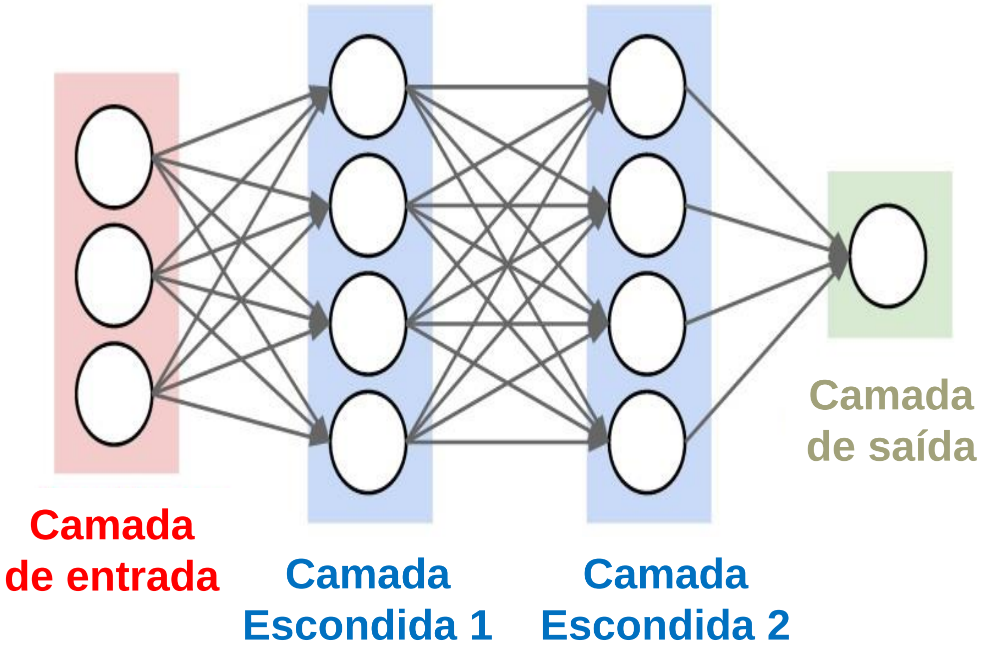
Arquitetura de Rede Neural *Feed-Forward*



“Rede Neural de 2 camadas” ou
“Rede Neuras com 1 camada escondida”

Arquitetura de Rede Neural *Feed-Forward*



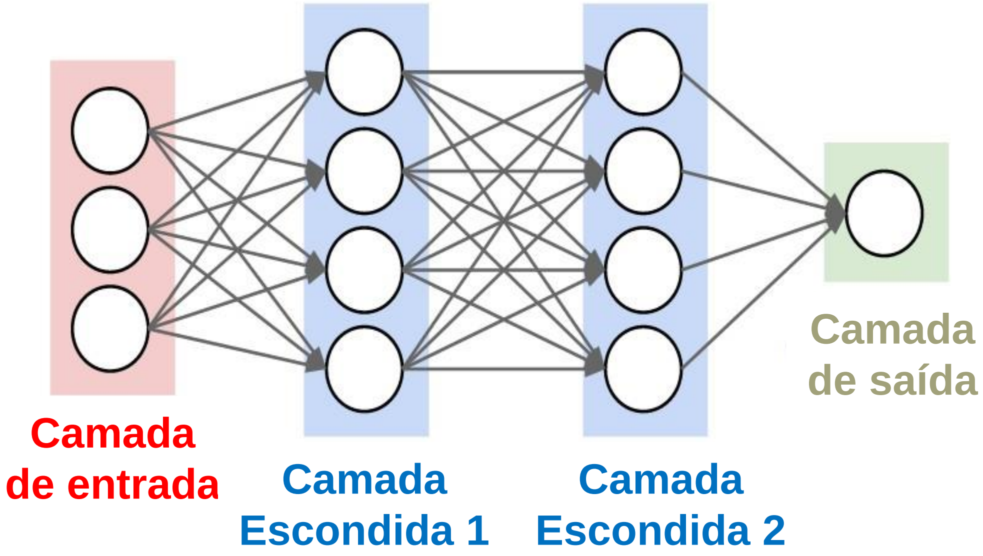


Camadas “completamente conectadas”

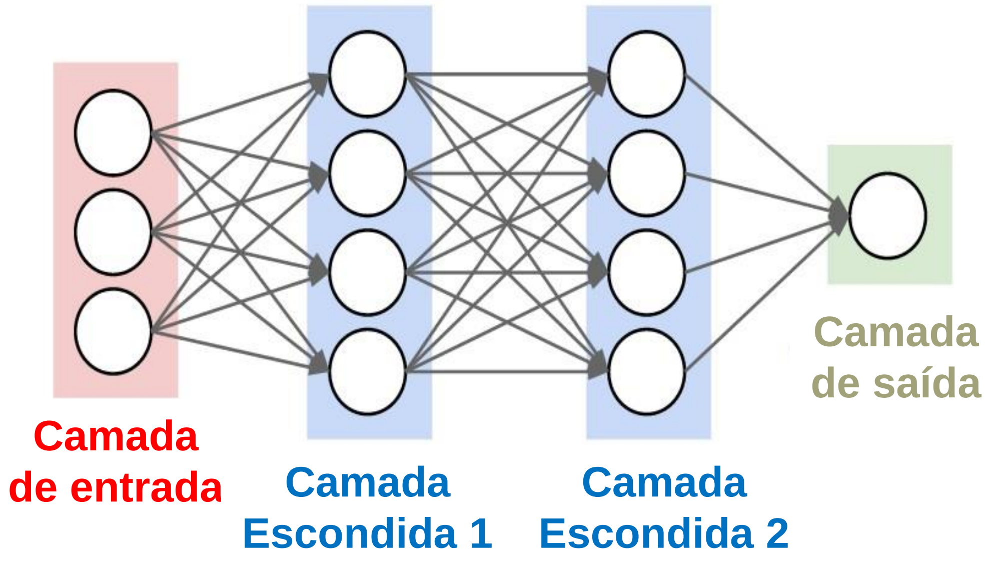
“Rede Neural de 2 camadas” ou
“Rede Neuras com 1 camada escondida”

“Rede Neural de 3 camadas” ou
“Rede Neuras com 2 camadas escondidas”

Exemplo de Avaliação de Rede *Feed-Forward*



Exemplo de Avaliação de Rede *Feed-Forward*



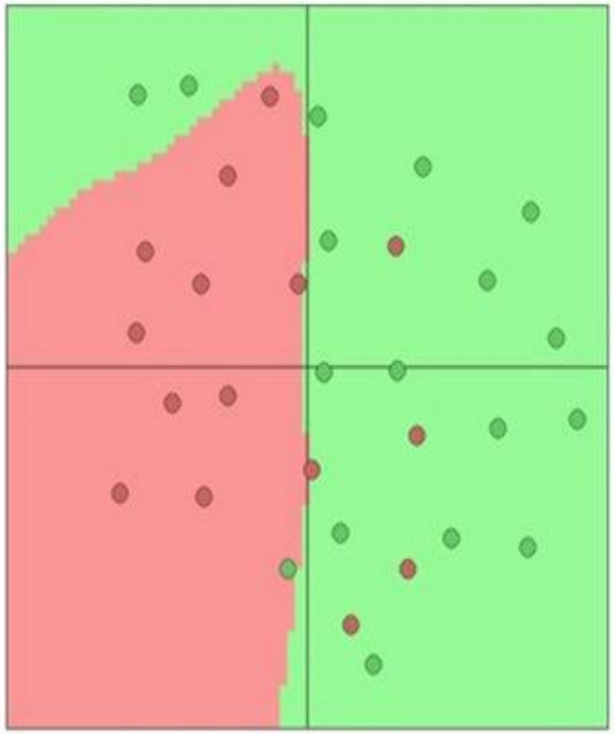
Pode-se avaliar eficientemente uma camada inteira de neurônios

```
f = lambda x: 1.0/(1.0 + np.exp(-x)) # activation function (use sigmoid)
x = np.random.randn(3, 1) # random input vector of three numbers (3x1)
h1 = f(np.dot(W1, x) + b1) # calculate first hidden layer activations (4x1)
h2 = f(np.dot(W2, h1) + b2) # calculate second hidden layer activations (4x1)
out = np.dot(W3, h2) + b3 # output neuron (1x1)
```

Definindo Tamanho das Camadas

Número de Neurônios na Camada Escondida

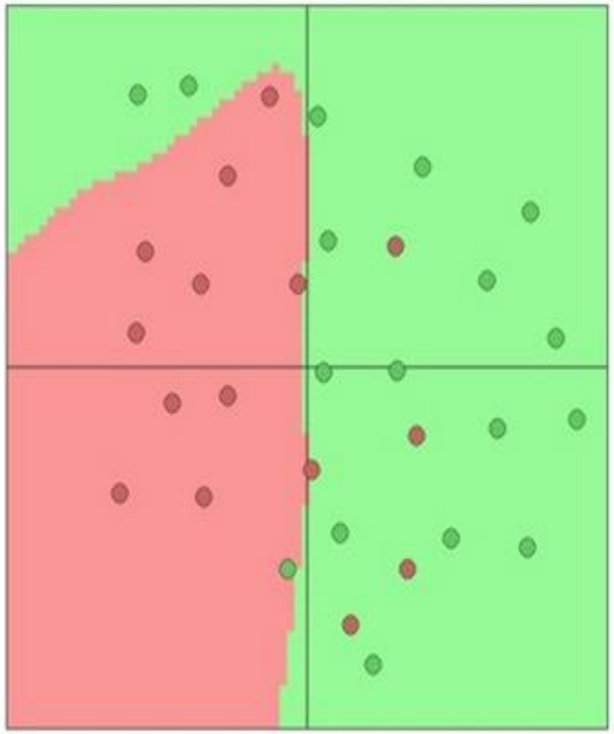
03 neurônios



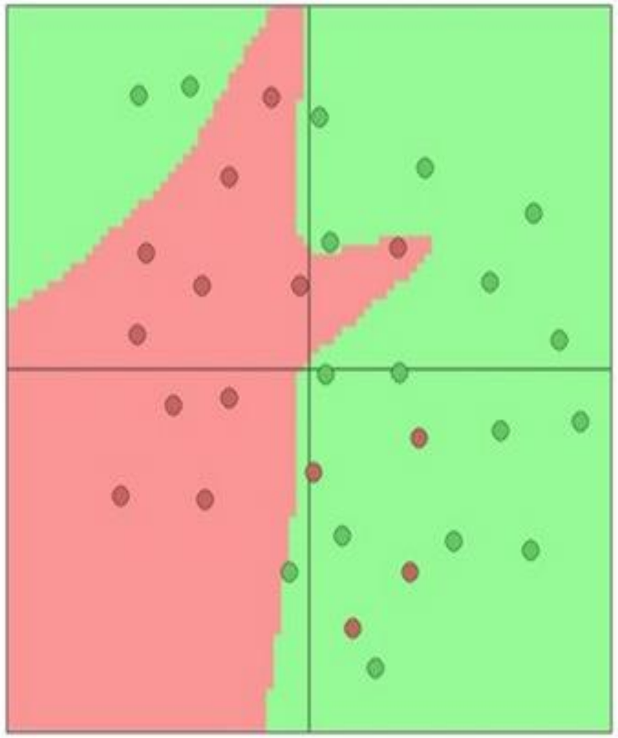
Definindo Tamanho das Camadas

Número de Neurônios na Camada Escondida

03 neurônios



06 neurônios



Definindo Tamanho das Camadas

Número de Neurônios na Camada Escondida

03 neurônios **06 neurônios** **20 neurônios**

Definindo Tamanho das Camadas

Número de Neurônios na Camada Escondida

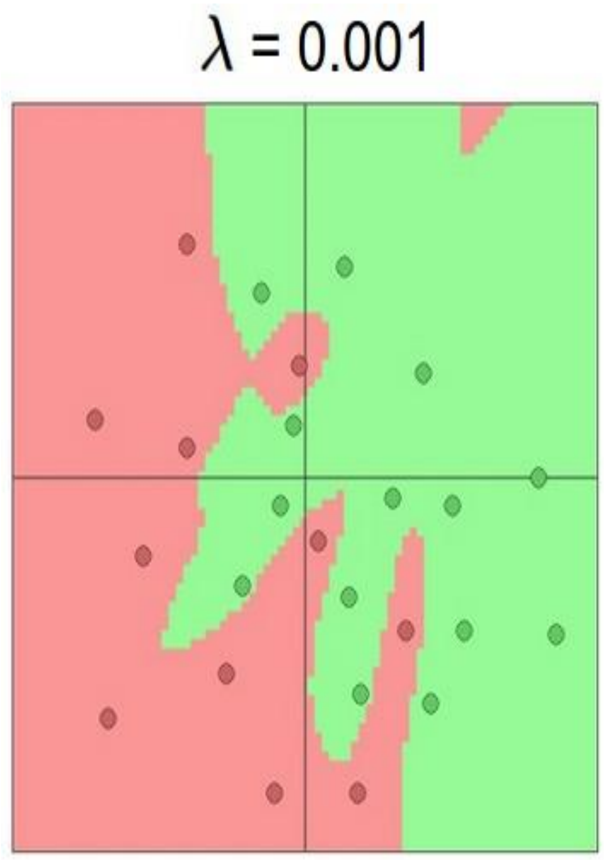
03 neurônios **06 neurônios** **20 neurônios**

mais neurônios ≡ maior capacidade

Regularização

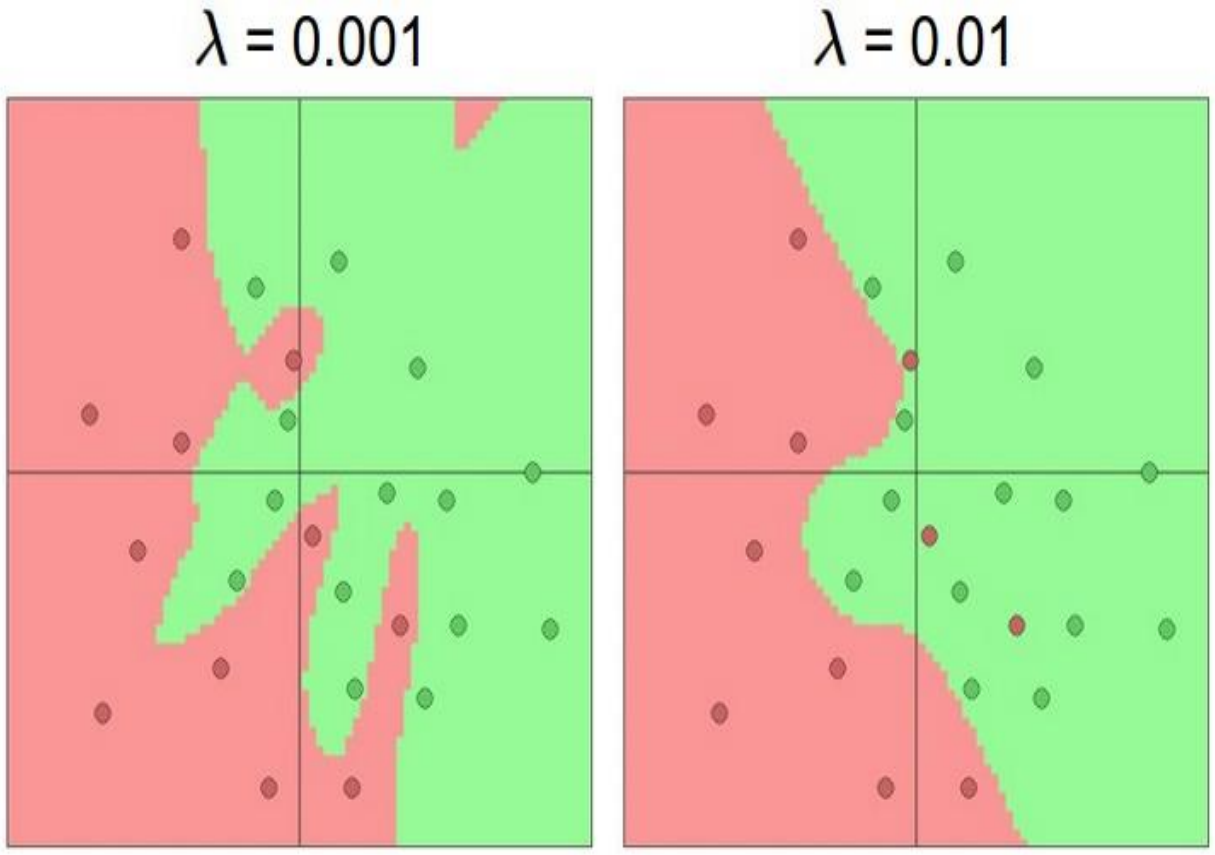
Não se deve usar o tamanho de uma rede para regularização

Deve-se aumentar a “força” da regularização



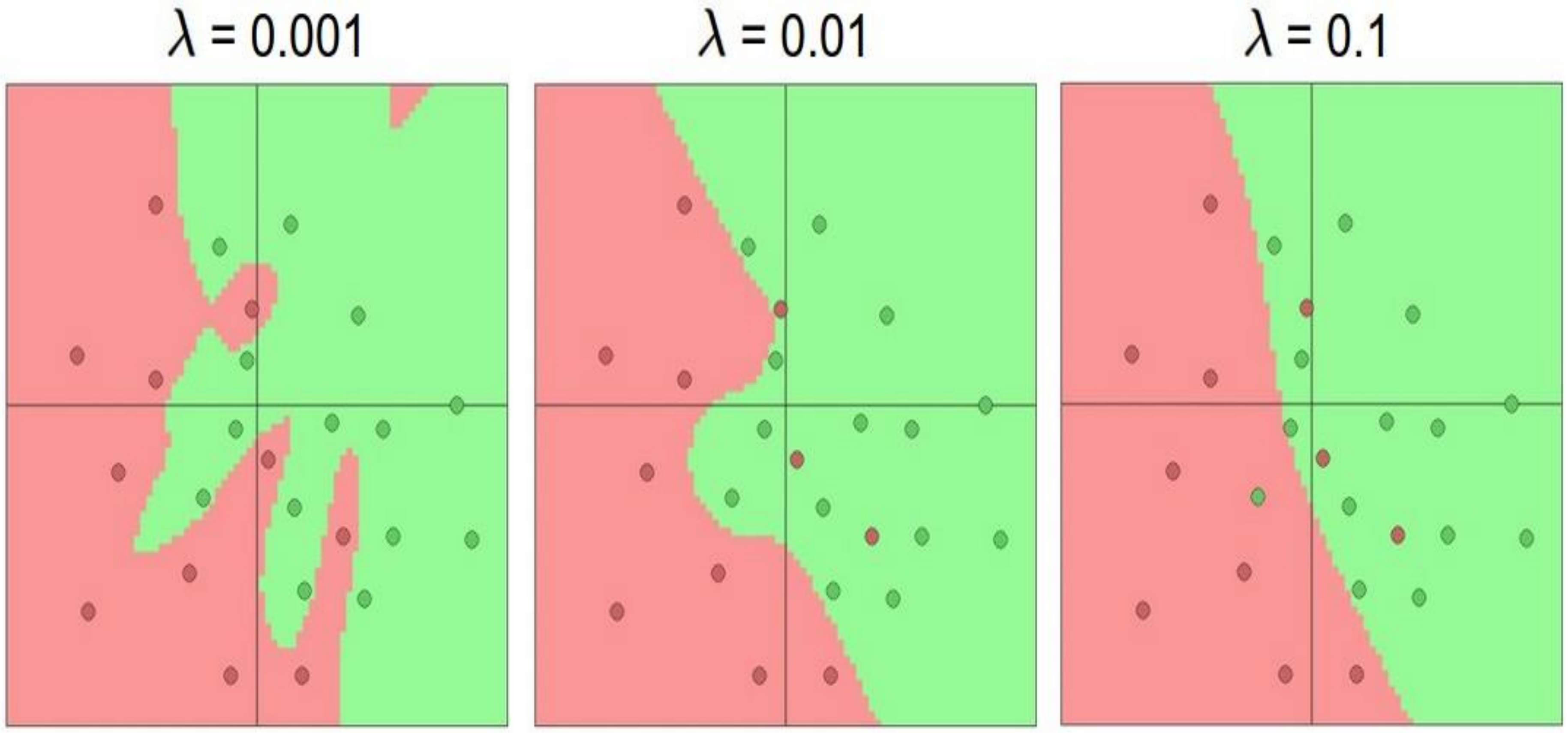
Regularização

Não se deve usar o tamanho de uma rede para regularização
Deve-se aumentar a “força” da regularização



Regularização

Não se deve usar o tamanho de uma rede para regularização
Deve-se aumentar a “força” da regularização



Um Pouco de História

A máquina **Mark I Perceptron** foi a primeira implementação do algoritmo perceptron

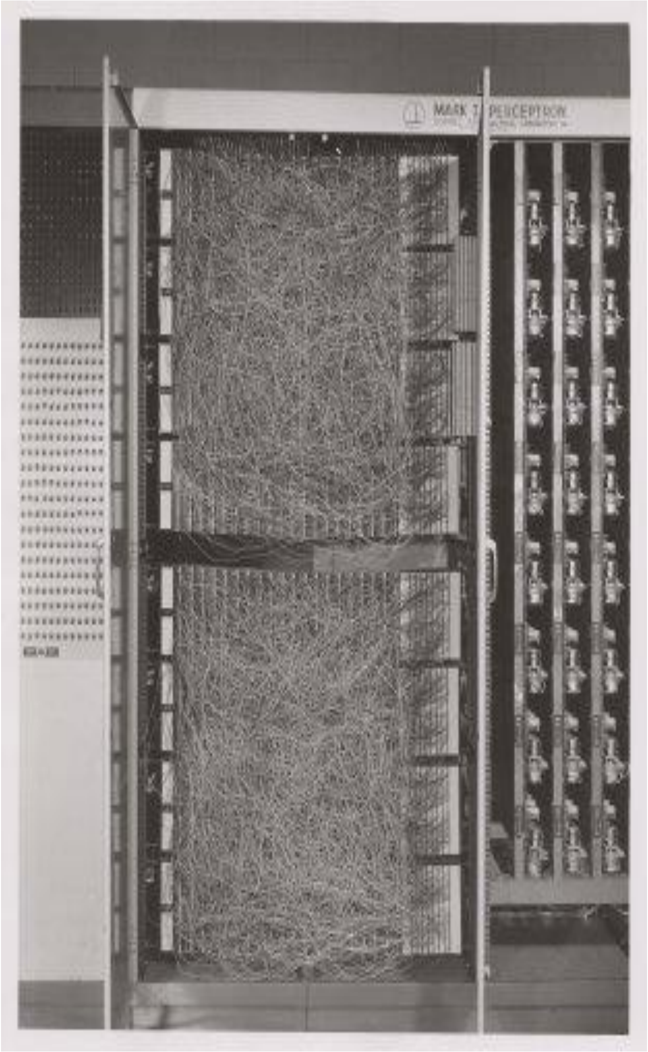
Essa máquina foi conectada a uma câmera capaz de produzir uma imagem de 400 pixels

Seu objetivo básico era o reconhecimento de imagens

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

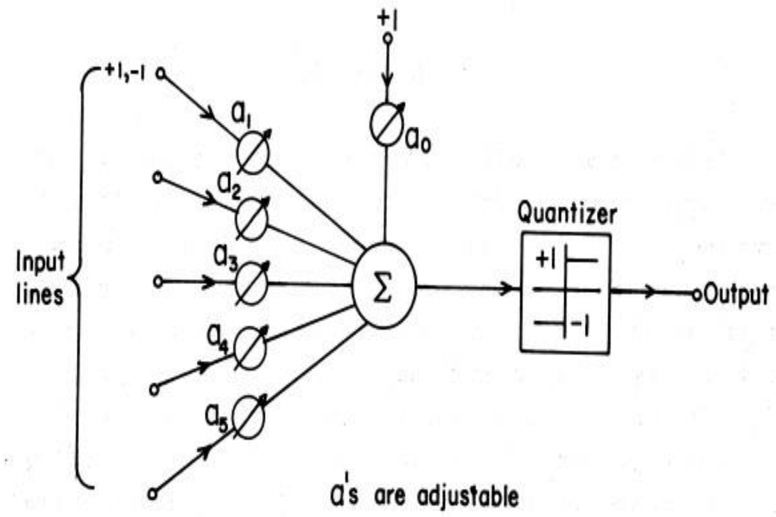
Regra de atualização :

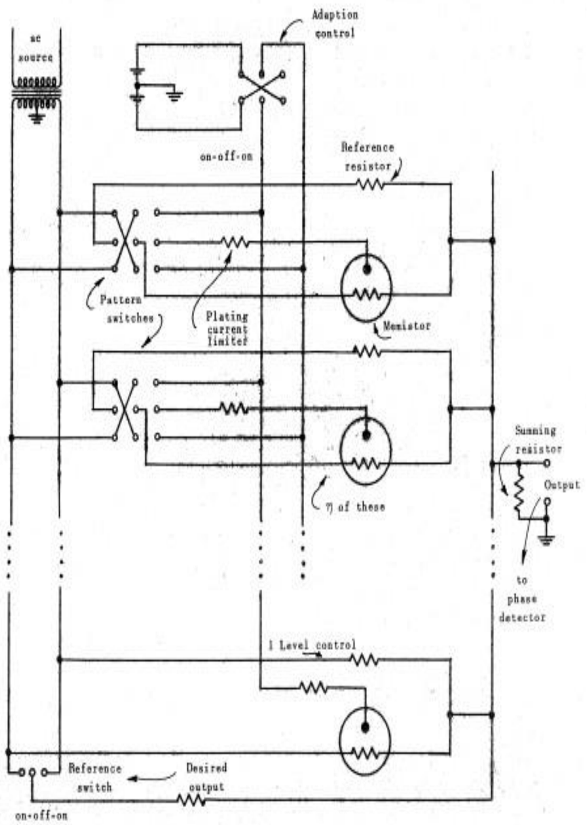
$$w_i(t+1) = w_i(t) + \alpha(d_j - y_j(t))x_{j,i},$$



Frank Rosenblatt, ~1957: Perceptron

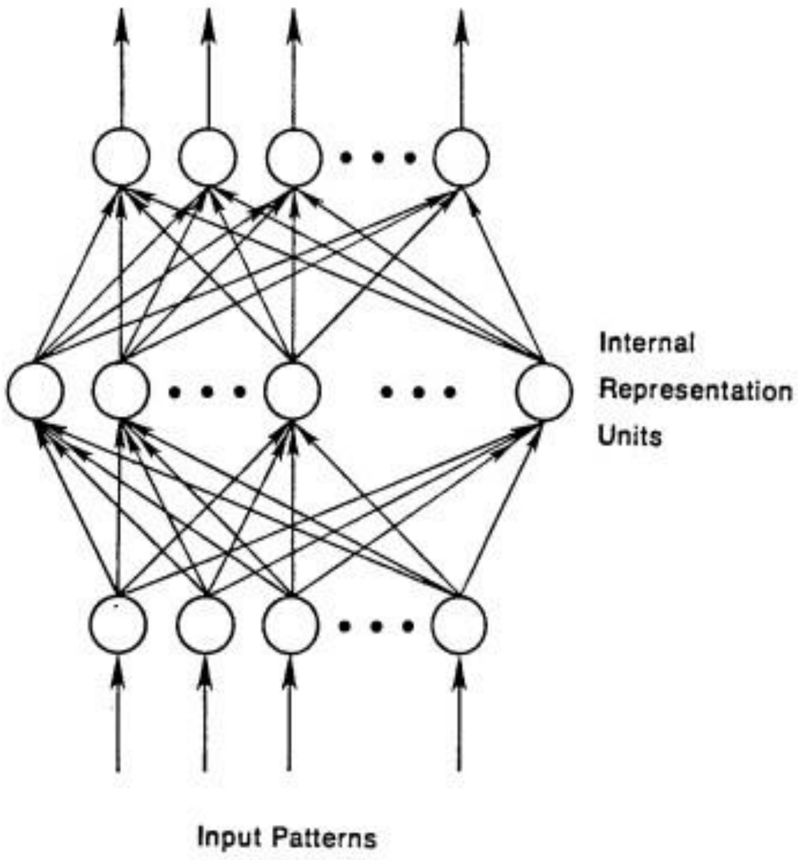
Um Pouco de História





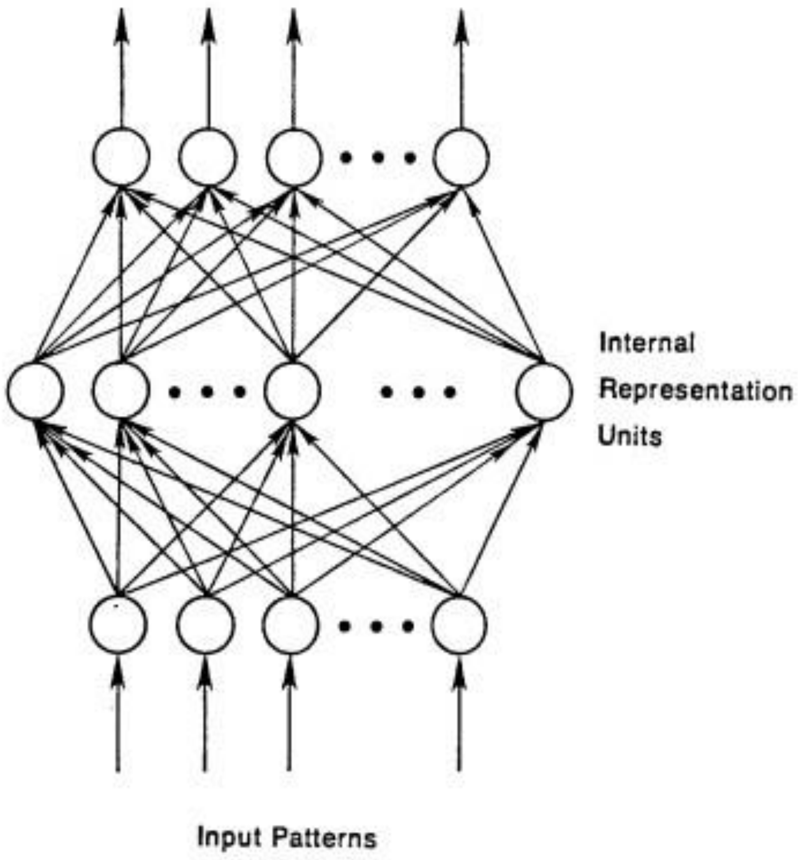
Widrow and Hoff, ~1960: Adaline/Madaline

Um Pouco de História



Rumelhart et al. 1986: Primeira vez em que a propagação retrógada se torna popular

Um Pouco de História



To be more specific, then, let

$$E_p = \frac{1}{2}\sum_j (t_{pj} - o_{pj})^2$$

be our measure of the error on input/output pattern p
overall measure of the error. We wish to show that the
dient descent in E when the units are linear. We will
that

$$-\frac{\partial E_p}{\partial w_{ji}} = \delta_{pj} i_{pi},$$

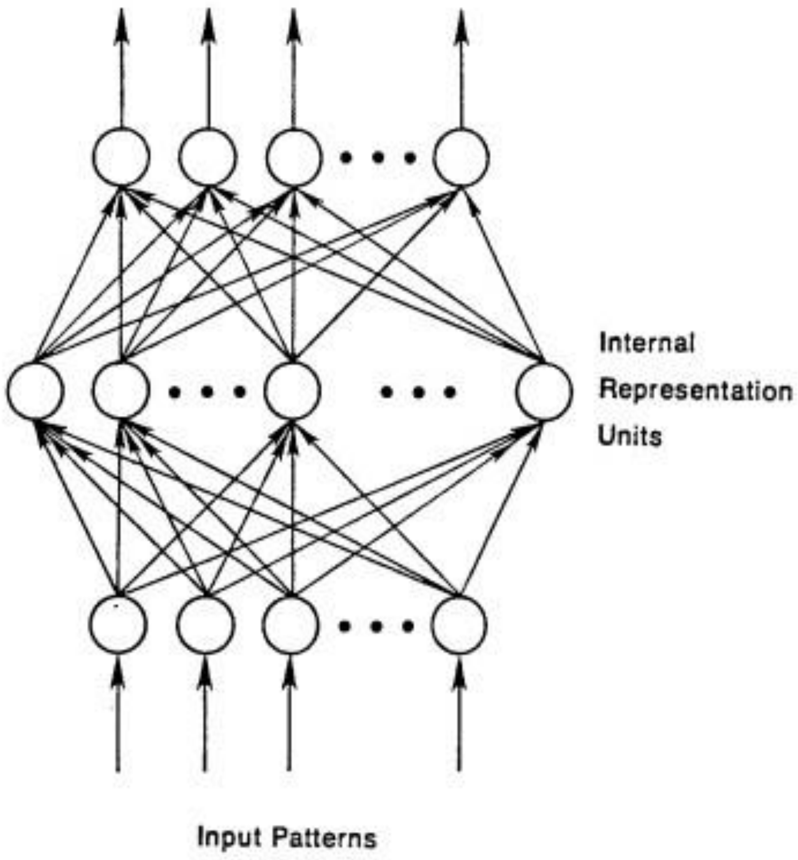
which is proportional to $\Delta_p w_{ji}$ as prescribed by the del
hidden units it is straightforward to compute the relevant
we use the chain rule to write the derivative as the prod
tive of the error with respect to the output of the unit tim
put with respect to the weight.

$$\frac{\partial E_p}{\partial w_{ji}} = \frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial w_{ji}}.$$

The first part tells how the error changes with the out
second part tells how much changing w_{ji} changes that

Rumelhart et al. 1986: Primeira vez em que a propagação retrógada se torna popular

Um Pouco de História



To be more specific, then, let

$$E_p = \frac{1}{2}\sum_j (t_{pj} - o_{pj})^2$$

be our measure of the error on input/output pattern p
overall measure of the error. We wish to show that the
dient descent in E when the units are linear. We will
that

$$-\frac{\partial E_p}{\partial w_{ji}} = \delta_{pj} i_{pi},$$

which is proportional to $\Delta_p w_{ji}$ as prescribed by the del
hidden units it is straightforward to compute the relevant
we use the chain rule to write the derivative as the prod
tive of the error with respect to the output of the unit tim
put with respect to the weight.

$$\frac{\partial E_p}{\partial w_{ji}} = \frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial w_{ji}}.$$

The first part tells how the error changes with the out
second part tells how much changing w_{ji} changes that o

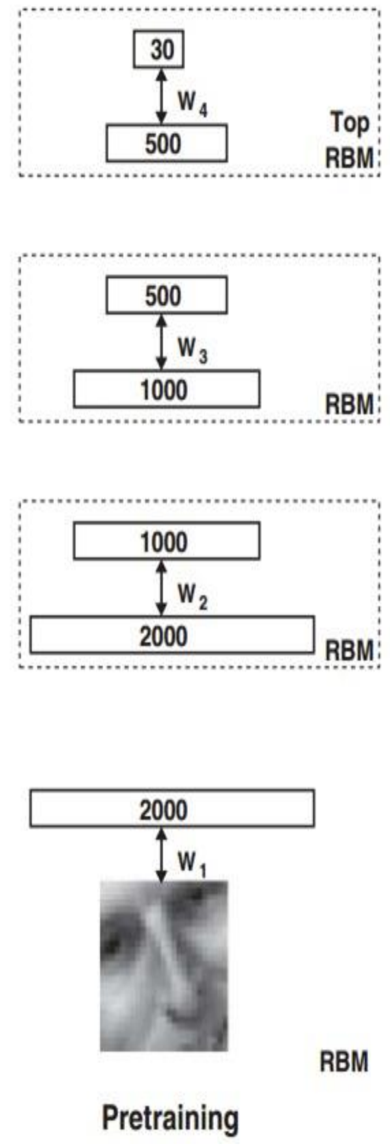
Matemática reconhecível

Rumelhart et al. 1986: Primeira vez em que a propagação retrógada se torna popular

Um Pouco de História

Hinton and Salakhutdinov 2006

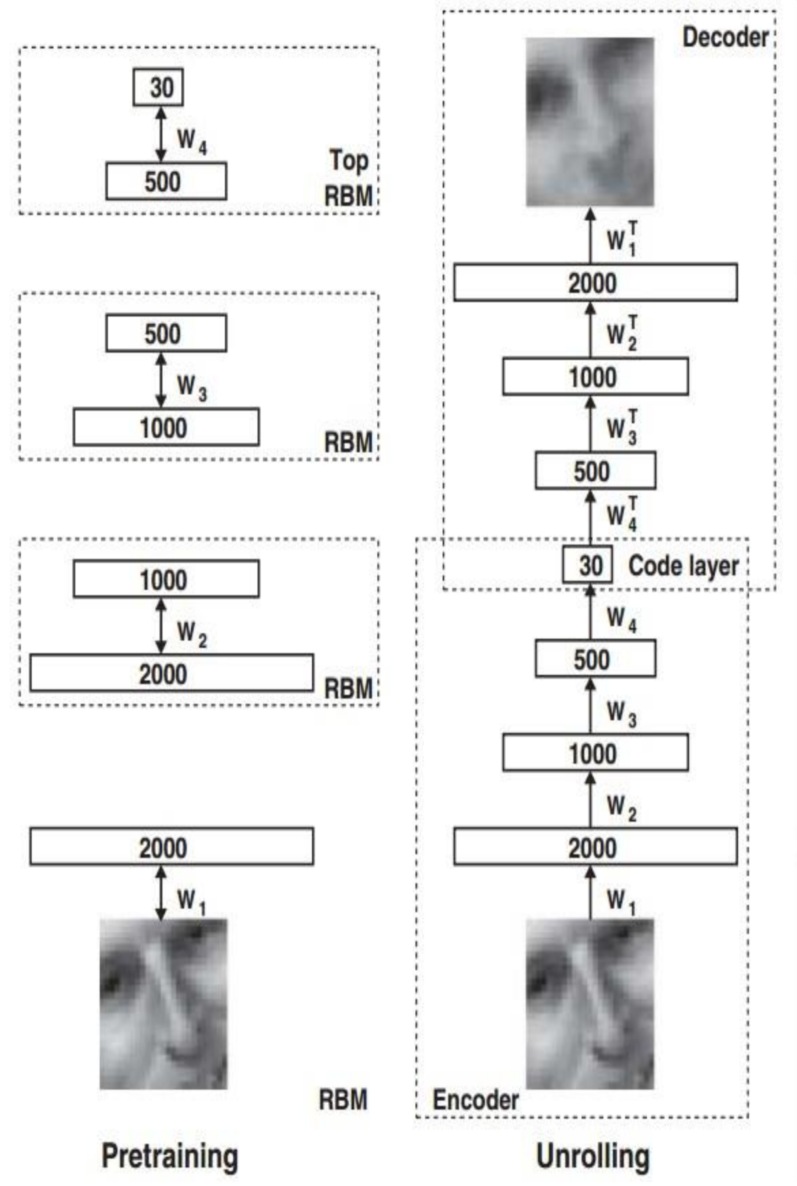
Pesquisa revigorada em *Deep Learning*



Um Pouco de História

Hinton and Salakhutdinov 2006

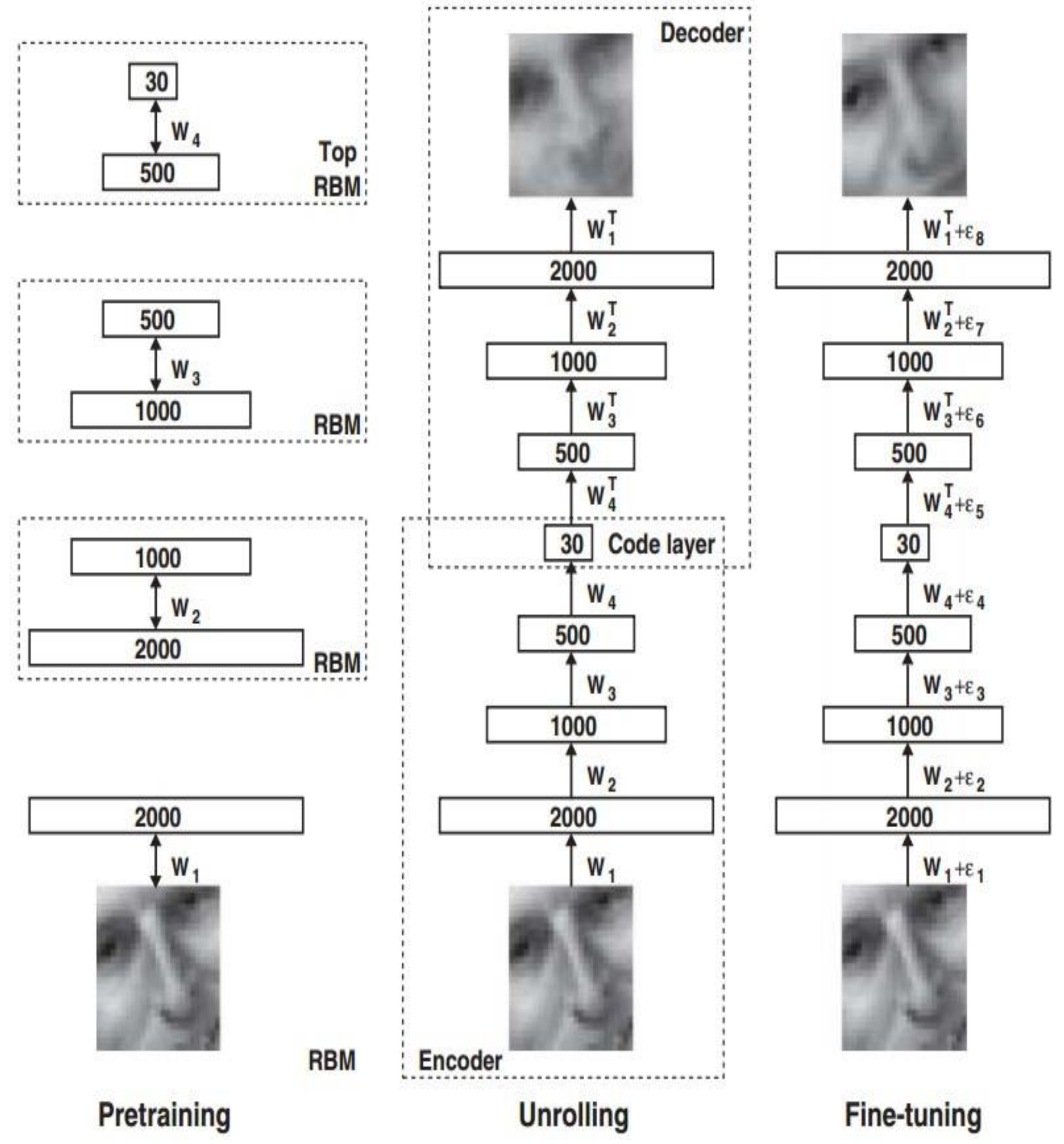
Pesquisa revigorada em
Deep Learning



Um Pouco de História

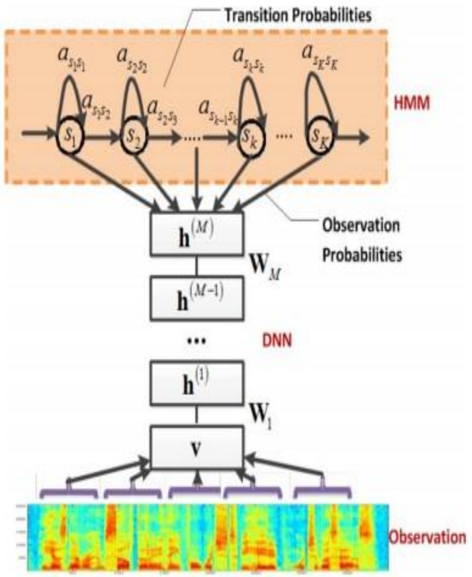
Hinton and Salakhutdinov 2006

Pesquisa revigorada em
Deep Learning



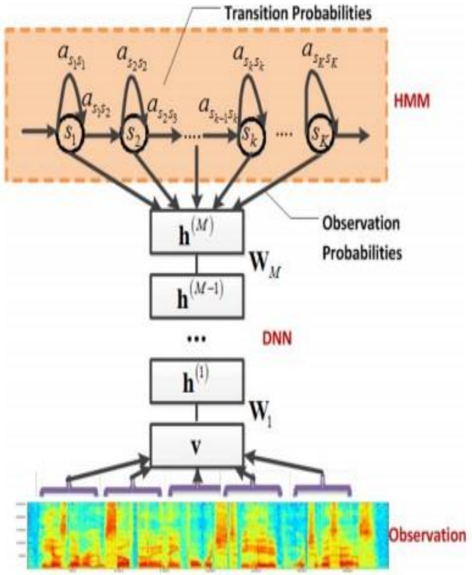
Um Pouco de História – 1os Resultados “Fortes”

Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition
George Dahl, Dong Yu, Li Deng, Alex Acero, 2010



Um Pouco de História – 1os Resultados "Fortes"

Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition
George Dahl, Dong Yu, Li Deng, Alex Acero, 2010



Imagenet classification with deep convolutional neural networks
Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton, 2012