RNP-VC

Aula 2: Introdução a Redes Neurais

Lucas Pereira, Rafael Teixeira, Lucas Assis, Anderson Soares
Instituto de Informática
Universidade Federal de Goiás (UFG)

DEEP LEARNING BRASIL

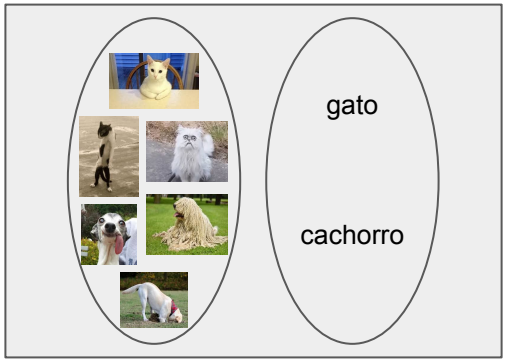
Sumário

- No último episódio...
- Portas lógicas: “aprendendo” as portas NOT e AND
- Perceptron
- O problema da porta XOR
- Redes **M**ulti**L**ayer **P**erceptron (**MLP**)
- Backprop (retropropagação)
- Redes neurais convolucionais (CNN)
- No próximo episódio...

No último episódio...

- Áreas para tomadas de decisões

Dados



Modelo

$$f(\mathbf{x}; \mathbf{W}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Custo

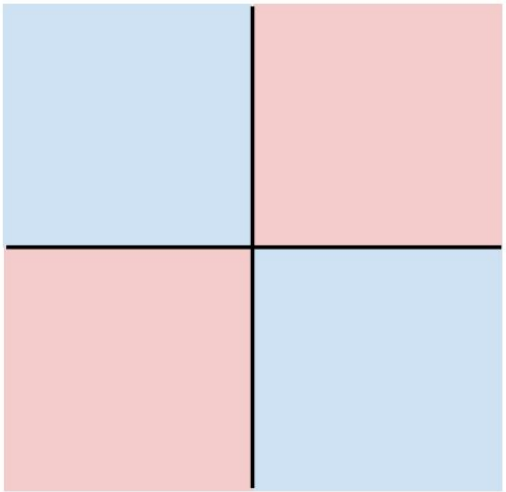
$$\|f(x; \mathbf{W}) - \widehat{y}\|_2^2$$

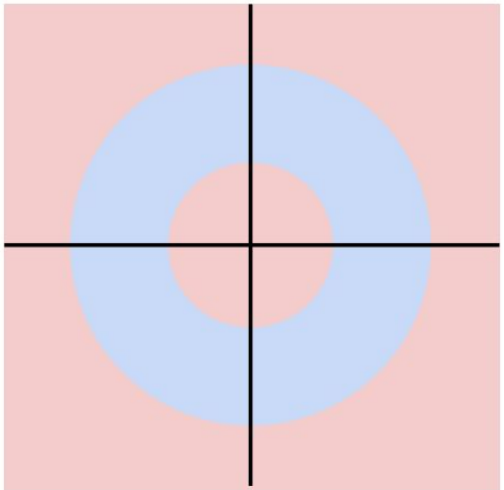
GD

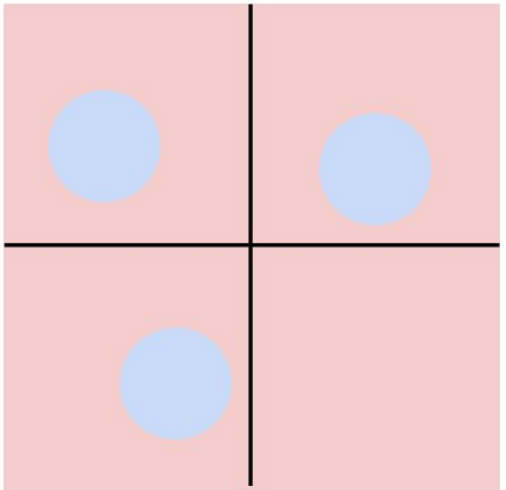
Otimização

No último episódio...

- Modelos lineares não resolvem problemas não-lineares…. derr...

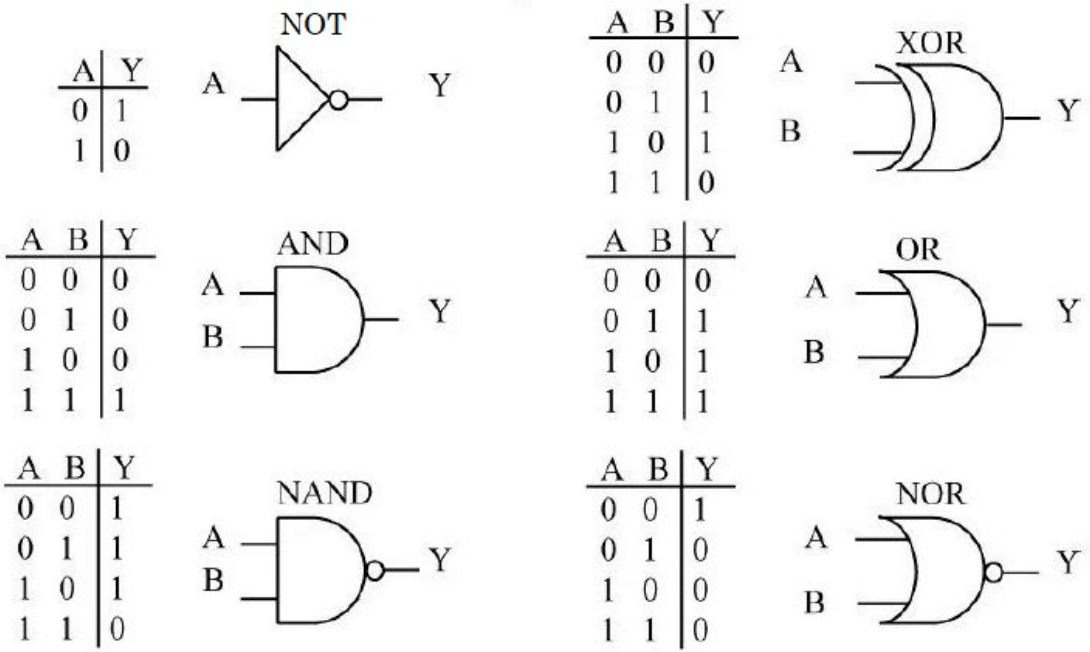






Portas lógicas: “aprendendo” as portas NOT e AND

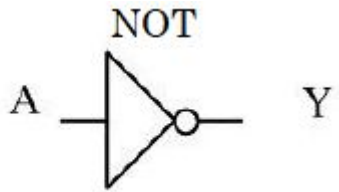
- Portas lógicas



“Aprendendo” a porta NOT

- Porta NOT

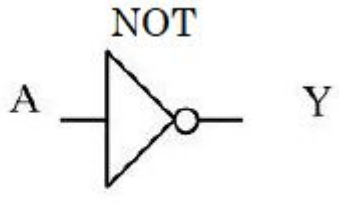
A	Y
0	1
1	0

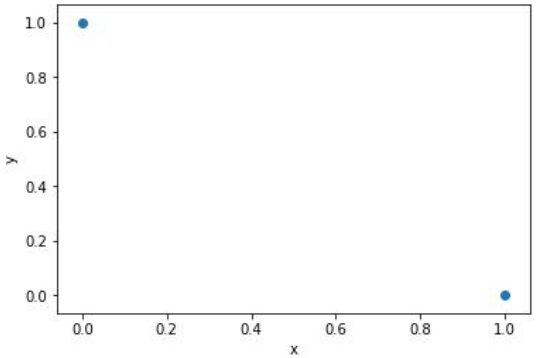


“Aprendendo” a porta NOT

- Porta NOT

A	Y
0	1
1	0





Dados

A	Y
0	1
1	0

Modelo

$$f(\mathbf{x};\mathbf{W}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Custo

$$\|f(x;\mathbf{W}) - \widehat{y}\|_2^2$$

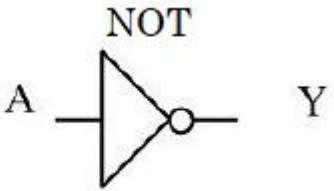
GD

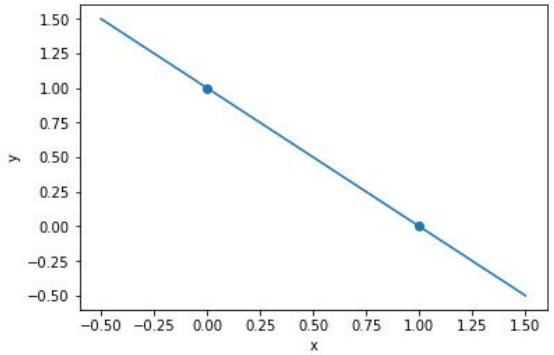
Otimização

“Aprendendo” a porta NOT

- Porta NOT

A	Y
0	1
1	0



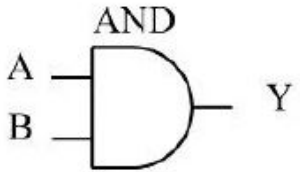


$$f(x) = -x + 1$$

“Aprendendo” a porta AND

- Porta AND

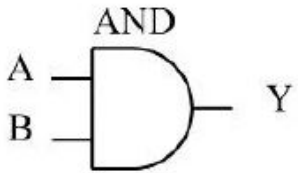
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

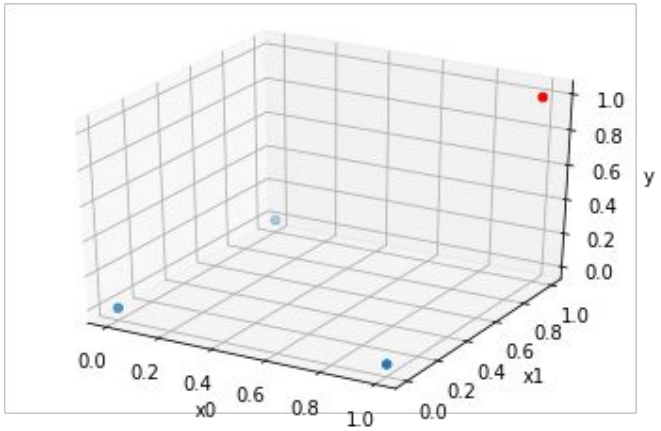


“Aprendendo” a porta AND

- Porta AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1





Dados

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Modelo

$$f(\mathbf{x};\mathbf{W}) = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Custo

$$\|f(x;\mathbf{W}) - \widehat{y}\|_2^2$$

?

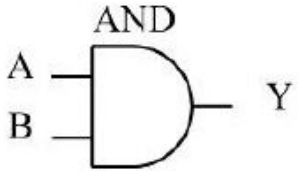
GD

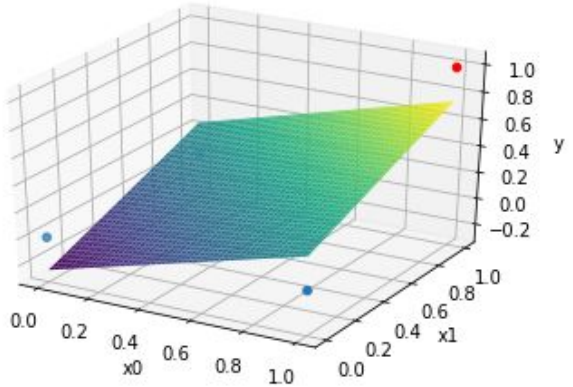
Otimização

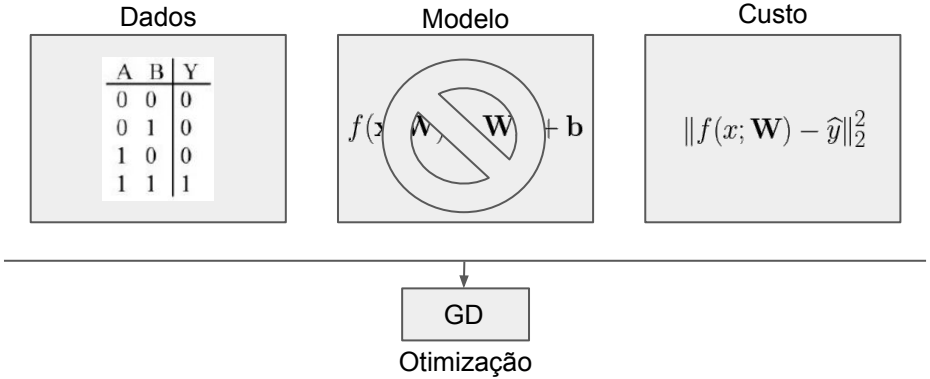
“Aprendendo” a porta AND

- Porta AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



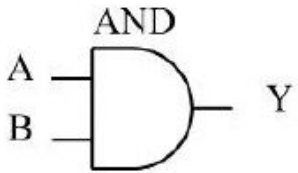


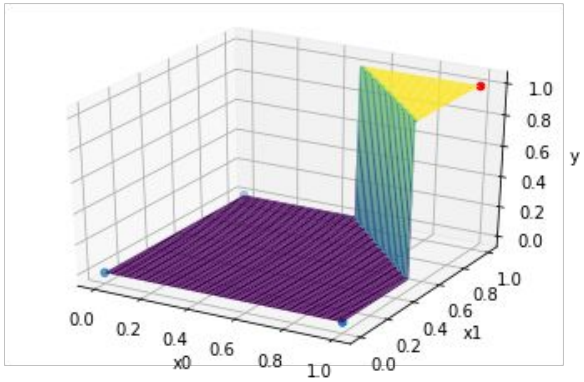


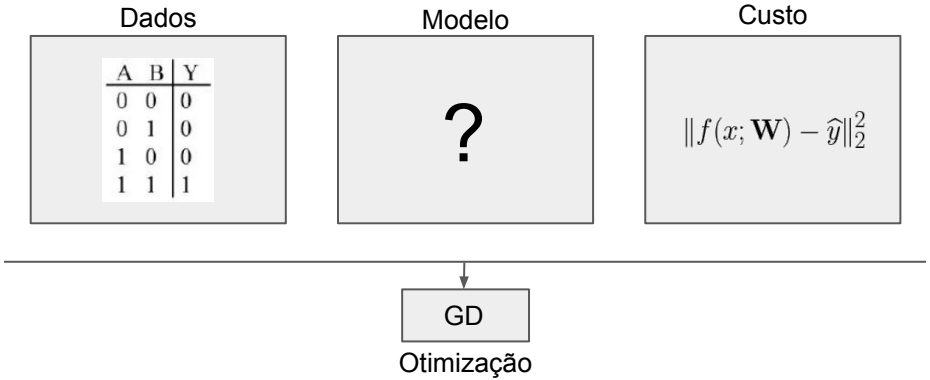
“Aprendendo” a porta AND

- Porta AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



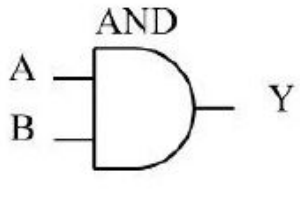


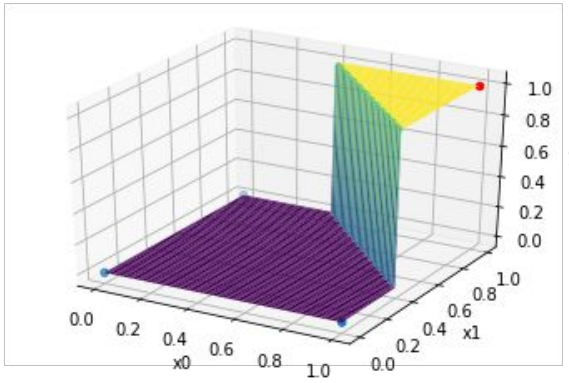


“Aprendendo” a porta AND

- Porta AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



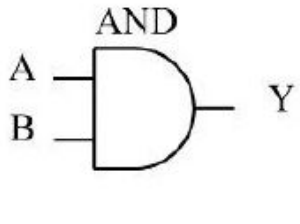


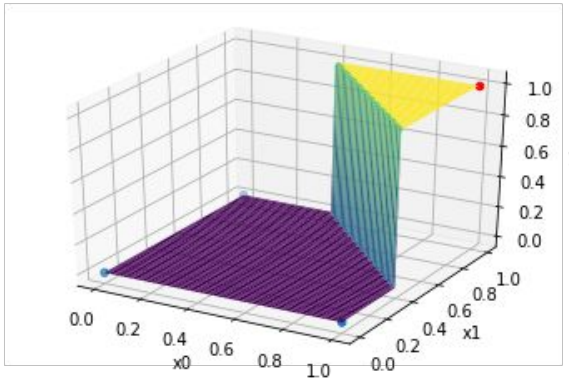
$$f(x) = (x_0 + x_1 - 1.5 > 0)$$

“Aprendendo” a porta AND

- Porta AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



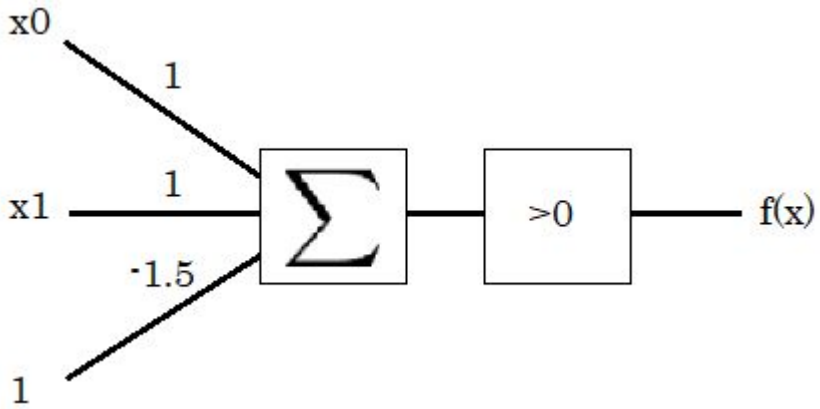


$$f(x) = (x_0 + x_1 - 1.5 > 0)$$

Não é derivavel! =(

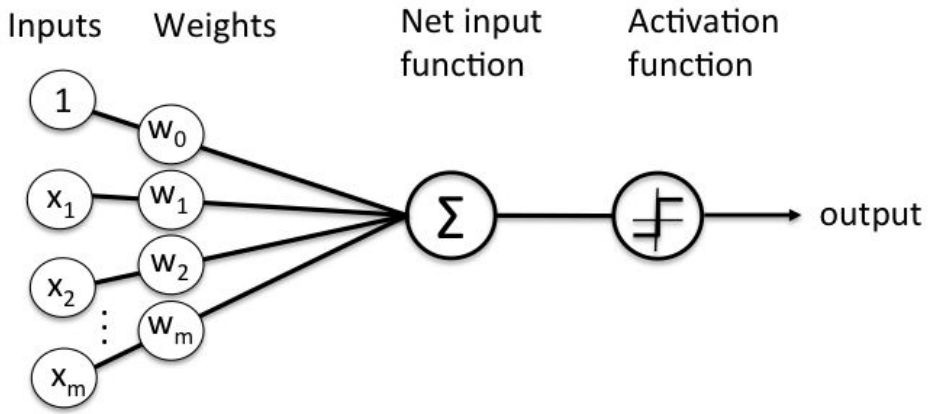
Perceptron

- Escrevendo f de outra forma...



Perceptron

- O perceptron de Rosenblatt, 1958

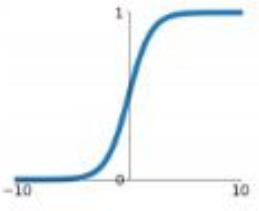


Perceptron

- A função sigmoid no lugar do “>0”

Sigmoid

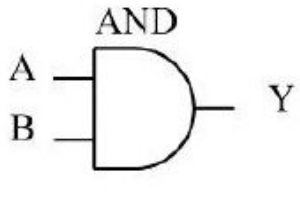
$$\sigma(x) = \frac{1}{1+e^{-x}}$$

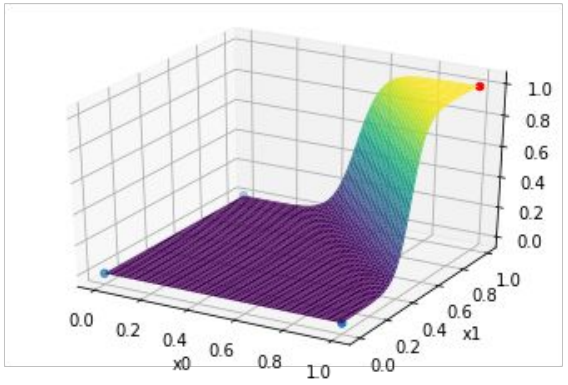


Perceptron

- Porta AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1





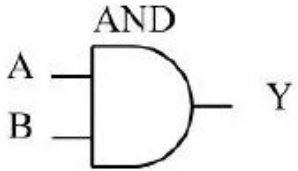
$$f(x) = \frac{1}{1 + e^{-20(x_0 + x_1 - 1.5)}}$$

É derivavel! =)

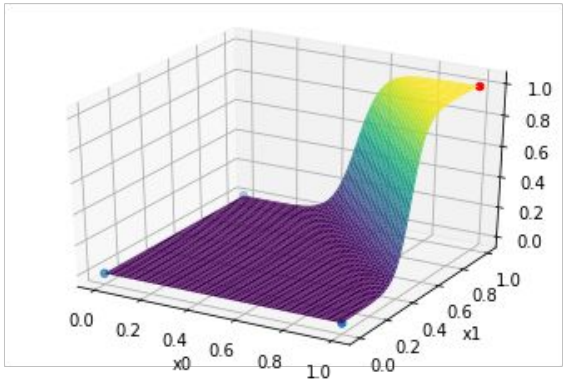
Perceptron

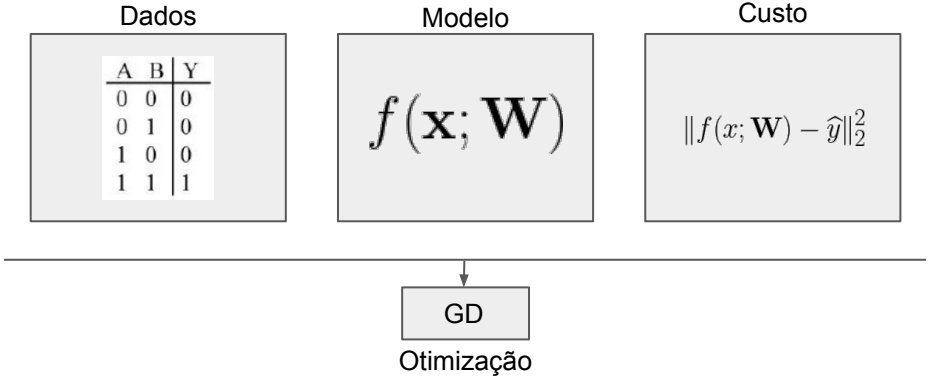
- Porta AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



$$f(\mathbf{x};\mathbf{W}) = \frac{1}{1+e^{-(\mathbf{W}\mathbf{x}+\mathbf{b})}}$$

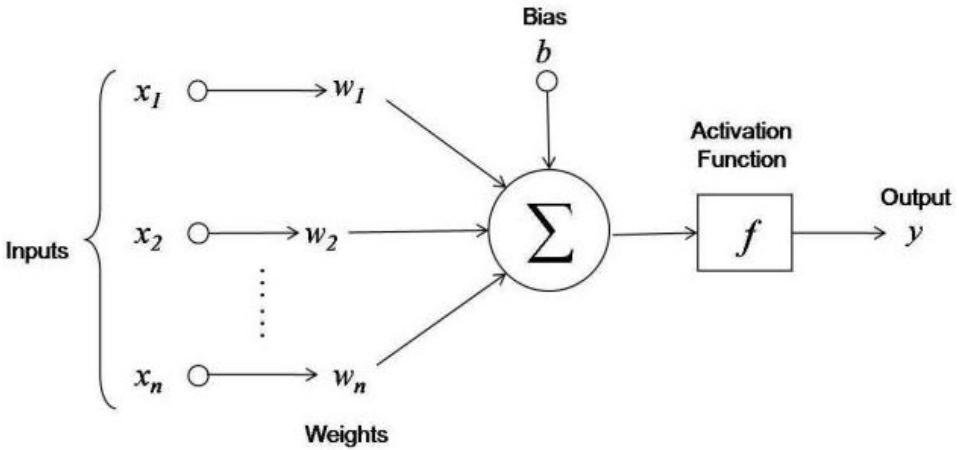




!

Perceptron

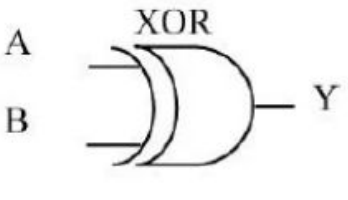
- O perceptron atual



O problema da porta XOR

- Porta XOR

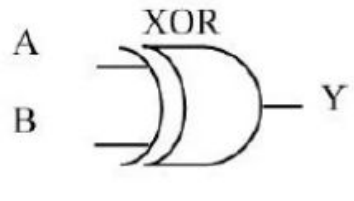
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

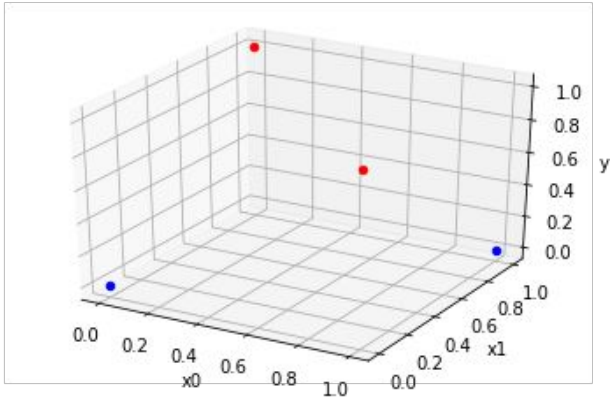


O problema da porta XOR

- Porta XOR

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

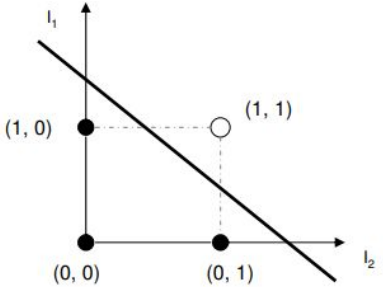




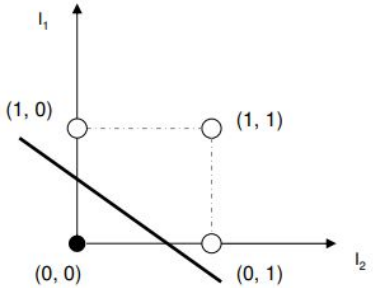
O problema da porta XOR

- Porta XOR

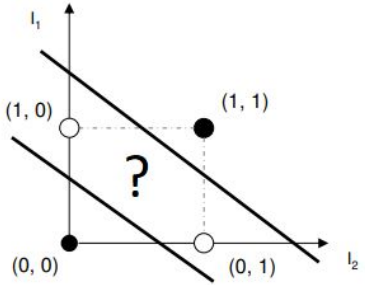
AND		
I_1	I_2	out
0	0	0
0	1	0
1	0	0
1	1	1



OR		
I_1	I_2	out
0	0	0
0	1	1
1	0	1
1	1	1

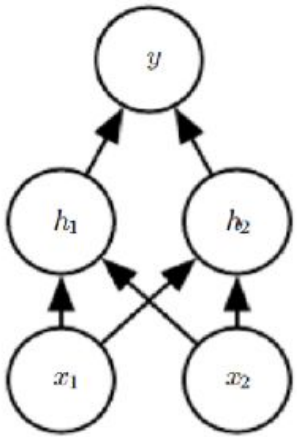


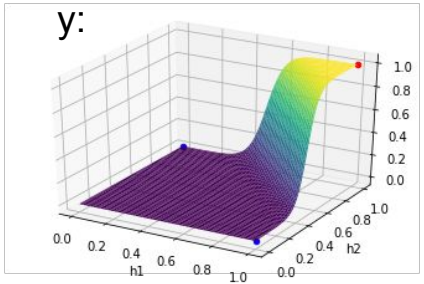
XOR		
I_1	I_2	out
0	0	0
0	1	1
1	0	1
1	1	0

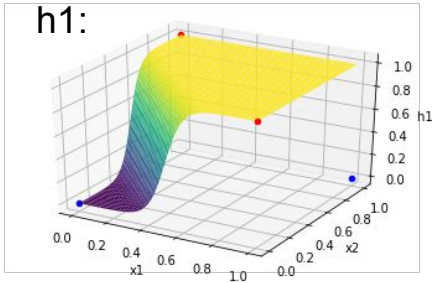


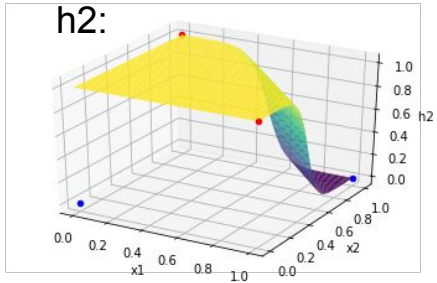
Redes **M**ulti**L**ayer **P**erceptron (**MLP**)

- Composição de representações



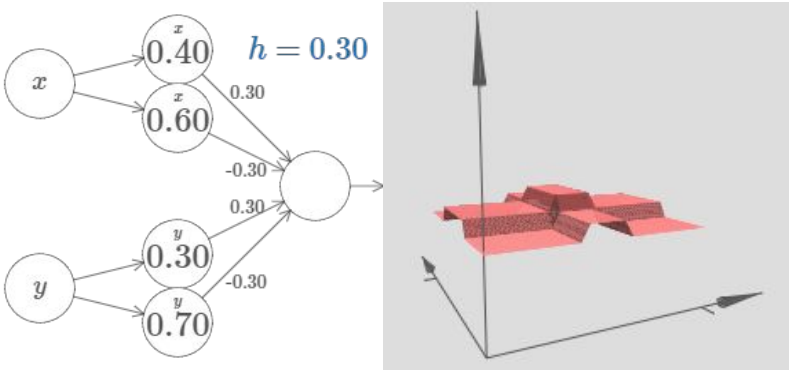


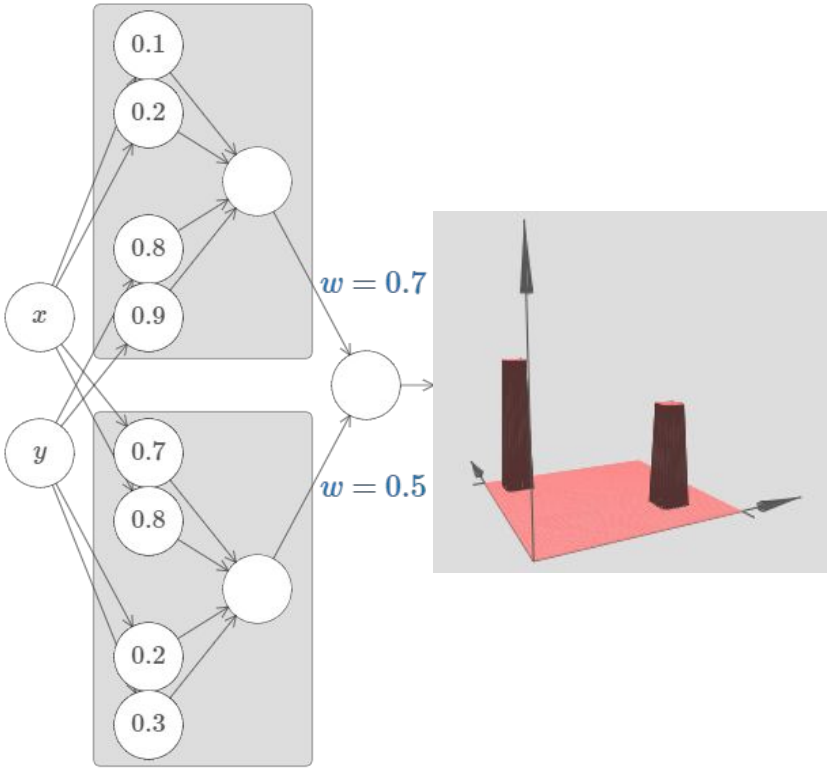




Redes **M**ulti**L**ayer **P**erceptron (**MLP**)

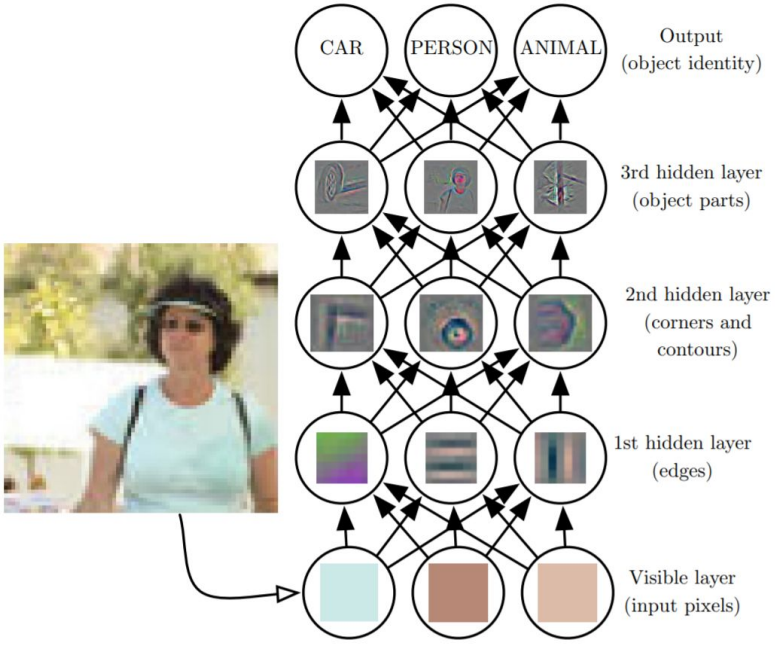
- Aproximador universal de funções





Redes **M**ulti**L**ayer **P**erceptron (**MLP**)

- Largura vs Profundidade



The Expressive Power of Neural Networks: A View from the Width

Zhou Lu[1,3] 1400010739@pku.edu.cn
Hongming Pu[1] 1400010621@pku.edu.cn
Feicheng Wang[1,3] 1400010604@pku.edu.cn
Zhiqiang Hu[2] huzq@pku.edu.cn
Liwei Wang[2,3] wanglw@cis.pku.edu.cn

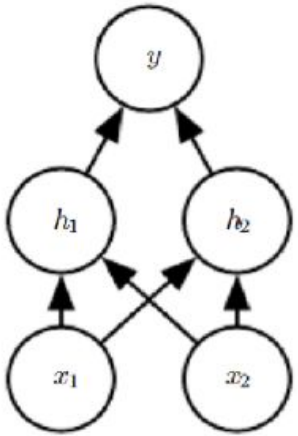
1, Department of Mathematics, Peking University
2, Key Laboratory of Machine Perception, MOE, School of EECS, Peking University
3, Center for Data Science, Peking University, Beijing Institute of Big Data Research

Abstract

The expressive power of neural networks is important for understanding deep learning. Most existing works consider this problem from the view of the depth of a network. In this paper, we study how width affects the expressiveness of neural networks. Classical results state that *depth-bounded* (e.g. depth-2) networks with suitable activation functions are universal approximators. We show a universal approximation theorem for *width-bounded* ReLU networks: width-$(n+4)$ ReLU networks, where n is the input dimension, are universal approximators. Moreover, except for a measure zero set, all functions cannot be approximated by width-n ReLU networks, which exhibits a phase transition. Several recent works demonstrate the benefits of depth by proving the depth-efficiency of neural networks. That is, there are classes of deep networks which cannot be realized by any shallow network whose size is no more than an *exponential* bound. Here we pose the dual question on the width-efficiency of ReLU networks: Are there wide networks that cannot be realized by narrow networks whose size is not substantially larger? We show that there exist classes of wide networks which cannot be realized by any narrow network whose depth is no more than a *polynomial* bound. On the other hand, we demonstrate by extensive experiments that narrow networks whose size exceed the polynomial bound by a constant factor can approximate wide and shallow network with high accuracy. Our results provide more comprehensive evidence that depth may be more effective than width for the expressiveness of ReLU networks.

Backprop (retropropagação)

- Composição de funções



$$y = f(h_1, h_2; W_y)$$
$$h_1 = g_1(x_1, x_2; W_{h_1})$$
$$h_2 = g_2(x_1, x_2; W_{h_2})$$
$$y = f(W_y, g_1(W_{h_1}), g_2(W_{h_2}))$$

Backprop (retropropagação)

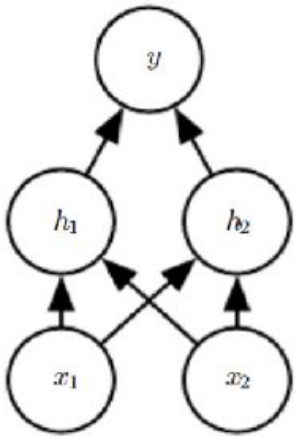
- Regra da cadeia

$$F(x) = f(g(x))$$

$$\frac{\mathrm{d}F}{\mathrm{d}x} = \frac{\mathrm{d}f(g(x))}{\mathrm{d}x} = \frac{\mathrm{d}f}{\mathrm{d}g(x)}\frac{\mathrm{d}g(x)}{\mathrm{d}x}$$

Backprop (retropropagação)

- Composição de funções + regra da cadeia



$$\frac{\partial y}{\partial W_y} = \frac{\partial f}{\partial W_y}$$

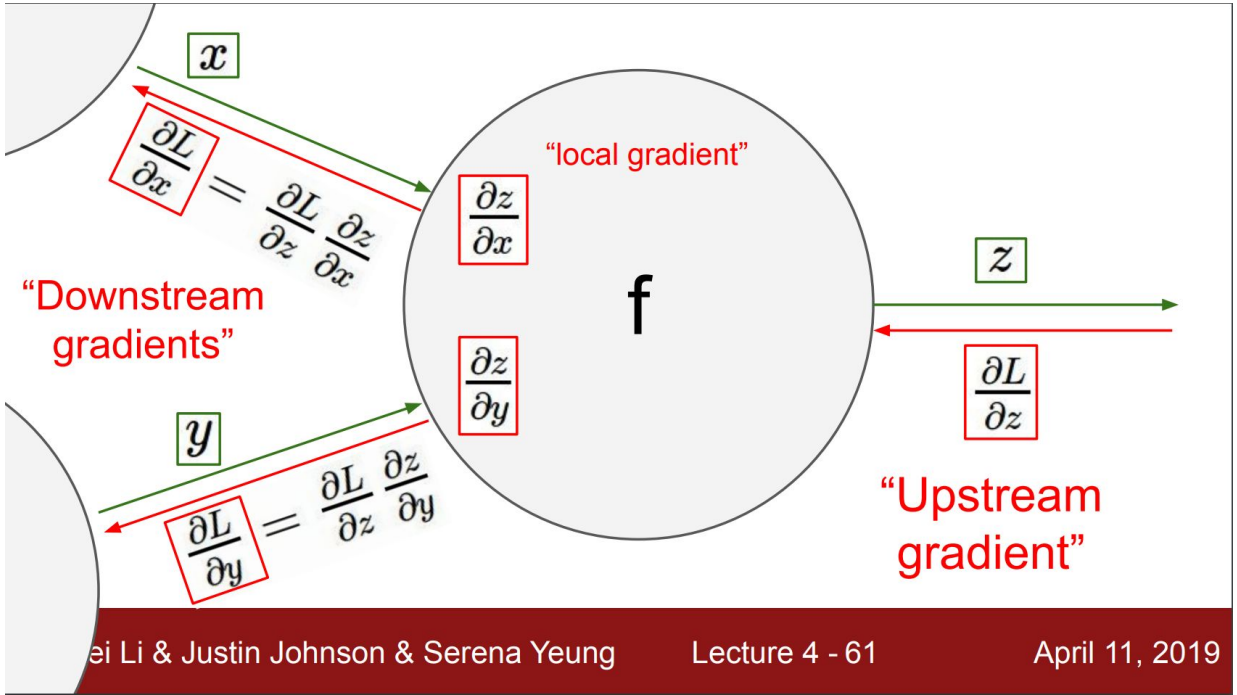
$$\frac{\partial y}{\partial W_{h_1}} = \frac{\partial f(g_1(W_{h_1}))}{\partial W_{h_1}} = \frac{\partial f}{\partial h_1}\frac{\partial g(W_{h_1})}{\partial W_{h_1}}$$

$$\frac{\partial y}{\partial W_{h_2}} = \frac{\partial f(g_2(W_{h_2}))}{\partial W_{h_2}} = \frac{\partial f}{\partial h_2}\frac{\partial g(W_{h_2})}{\partial W_{h_2}}$$

$$\nabla y$$

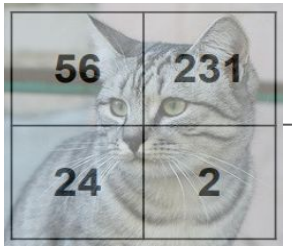
Backprop (retropropagação)

- Backprop = cálculo computacionalmente eficiente do gradiente pela regra da cadeia



Redes neurais convolucionais (CNN)

- Problema de classificação de imagens (aula 01)

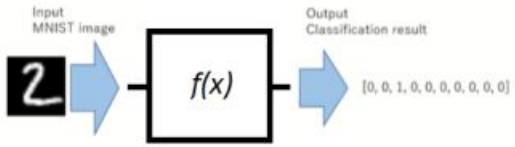


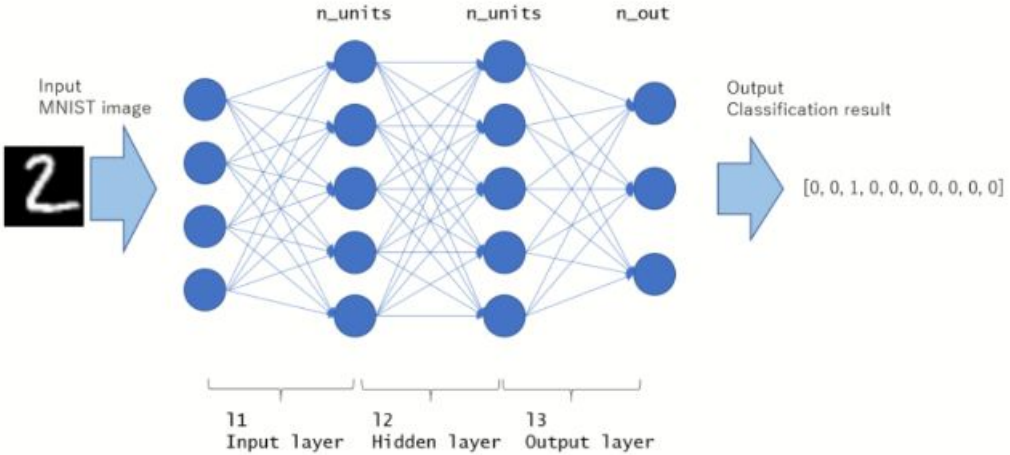
56
231
24
2

$\theta_{0,0}$	$\theta_{0,1}$	$\theta_{0,2}$	$\theta_{0,3}$	*	56	+	$\theta_{0,4}$	=	$f(x;\theta)$
$\theta_{1,0}$	$\theta_{1,1}$	$\theta_{1,2}$	$\theta_{1,3}$		231		$\theta_{1,4}$		
					24				
					2				

Redes neurais convolucionais (CNN)

- Problema de classificação de imagens (aula 02)



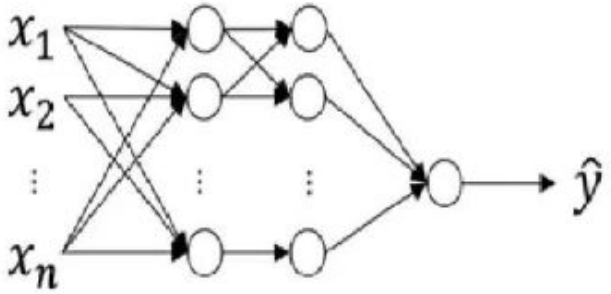


Redes neurais convolucionais (CNN)

- Complexidade

1920 px

1080 px



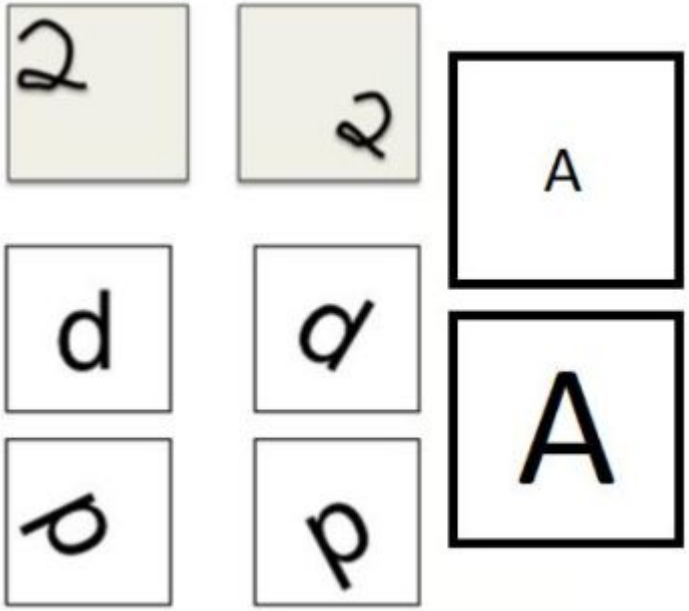
1080 x 1920 x 3 ≈ 6M

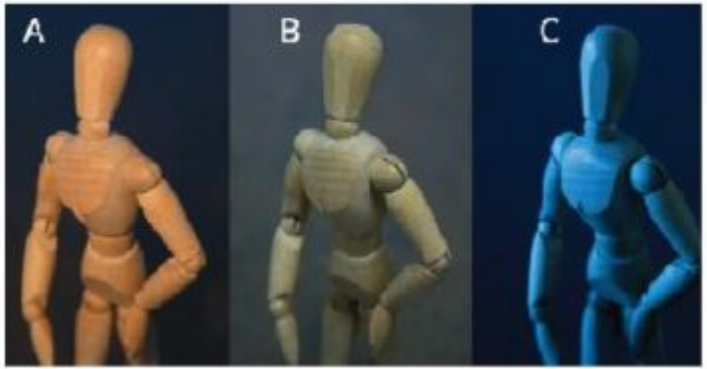
1 hidden layer (camada oculta) = 1000 neuronios

≈ 6 bilhões de parâmetros treináveis

Redes neurais convolucionais (CNN)

- Variabilidade dos dados





Redes neurais convolucionais (CNN)

- Algo em comum?

Redes neurais convolucionais (CNN)

- Algo em comum?

    - Relação espacial entre os pixels:
        - Pixels próximos: alta correlação
        - Pixels distantes: baixa correlação

Redes neurais convolucionais (CNN)

- E se… (LeCun, 1989)

Generalization and Network Design Strategies

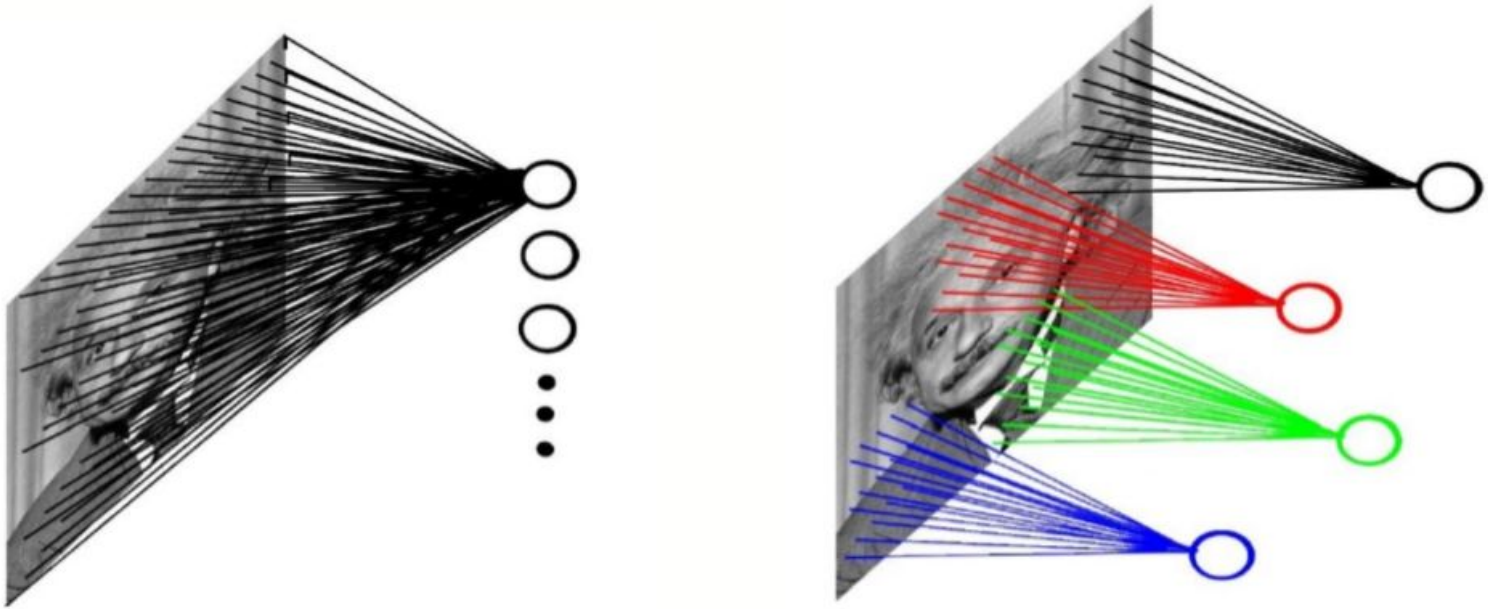
Yann le Cun *
Department of Computer Science, University of Toronto
Toronto, Ontario, M5S 1A4. CANADA.

Abstract

An interesting property of connectionist systems is their ability to learn from examples. Although most recent work in the field concentrates on reducing learning times, the most important feature of a learning machine is its generalization performance. It is usually accepted that good generalization performance on real-world problems cannot be achieved unless some *a priori* knowledge about the task is built into the system. Back-propagation networks provide a way of specifying such knowledge by imposing constraints both on the architecture of the network and on its weights. In general, such constraints can be considered as particular transformations of the parameter space

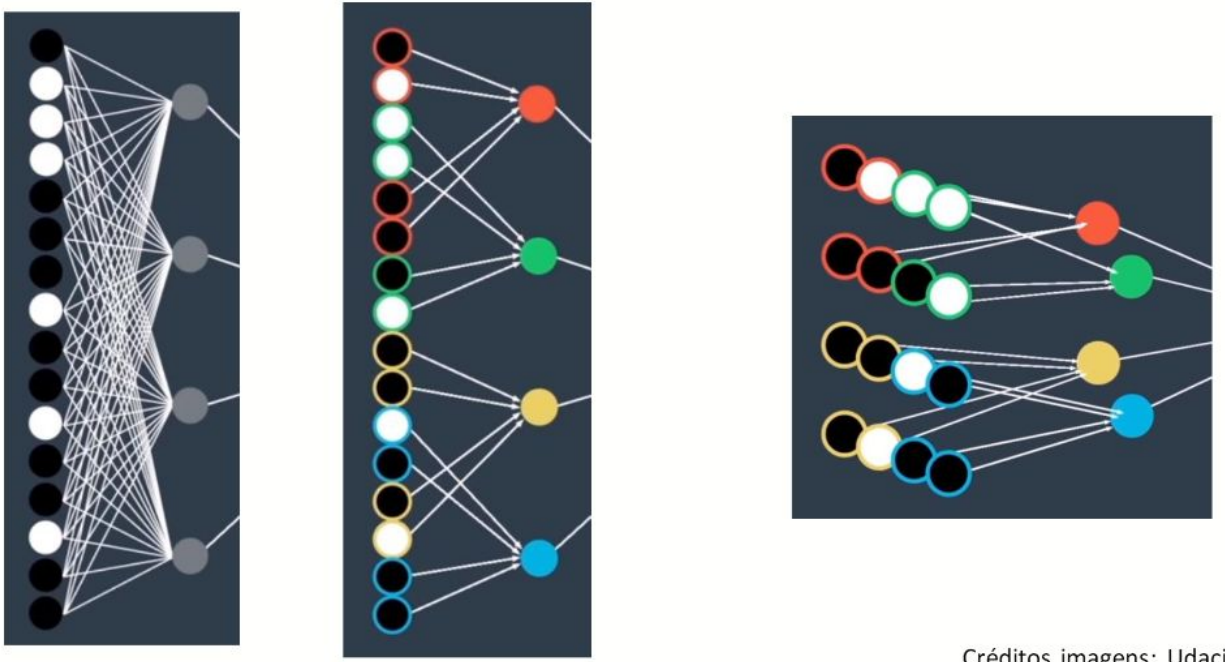
Redes neurais convolucionais (CNN)

- Viés indutivo: ligações locais



Redes neurais convolucionais (CNN)

- Viés indutivo: ligações locais



Créditos imagens: Udacity

Redes neurais convolucionais (CNN)

- Viés indutivo: ligações locais

    - Justificativas:

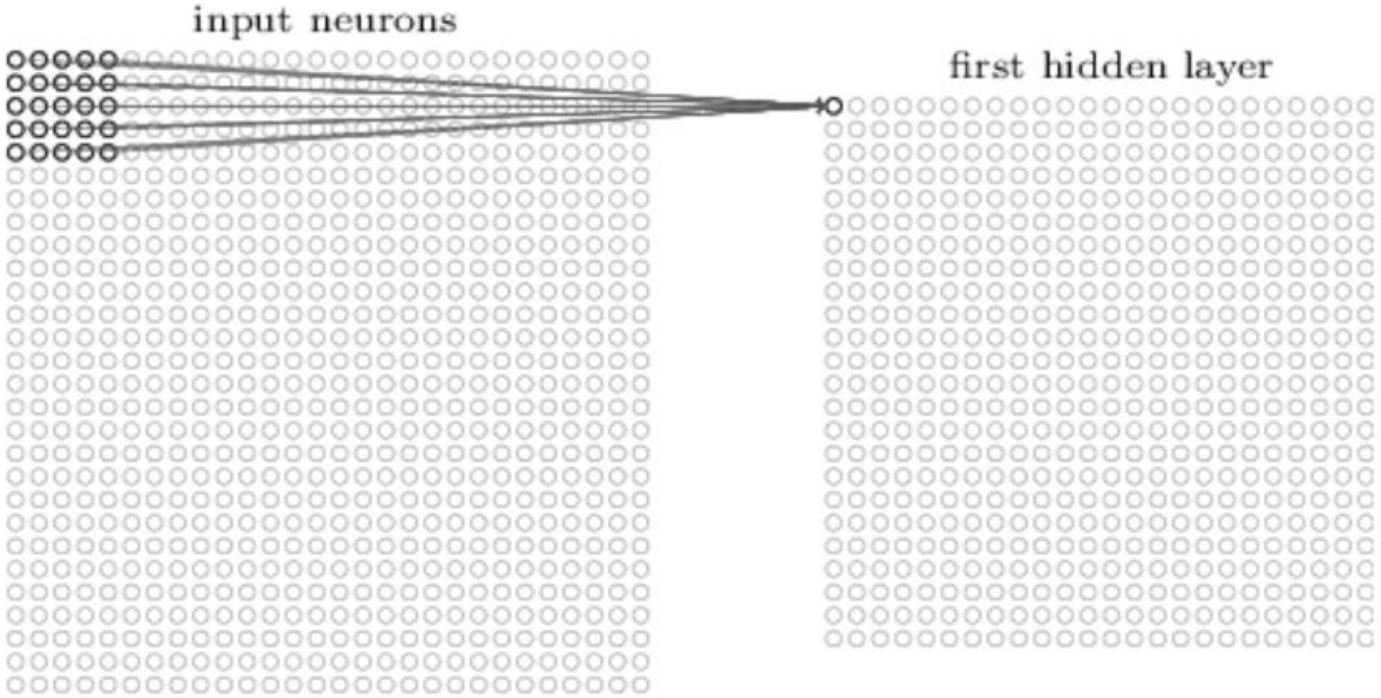
Redes neurais convolucionais (CNN)

- Viés indutivo: ligações locais

    - Justificativas:
        - Relação espacial
        - Diminuição do custo computacional (!!!)

Redes neurais convolucionais (CNN)

- Viés indutivo: compartilhamento de parâmetros



Redes neurais convolucionais (CNN)

- Viés indutivo: compartilhamento de parâmetros

    - Justificativas:

Redes neurais convolucionais (CNN)

- Viés indutivo: compartilhamento de parâmetros

    - Justificativas:
        - Variabilidade dos dados - equivariância translacional
        - Diminuição do custo computacional (!!!)

Redes neurais convolucionais (CNN)

- Experimentos do artigo (LeCun, 1989)

  - Dataset: MNIST

    - Imagens de digitos (0-9) escritos a mão
    - 60 mil exemplos de treino
    - 10 mil exemplos de teste

Redes neurais convolucionais (CNN)

- MNIST

Redes neurais convolucionais (CNN)

- Redes propostas

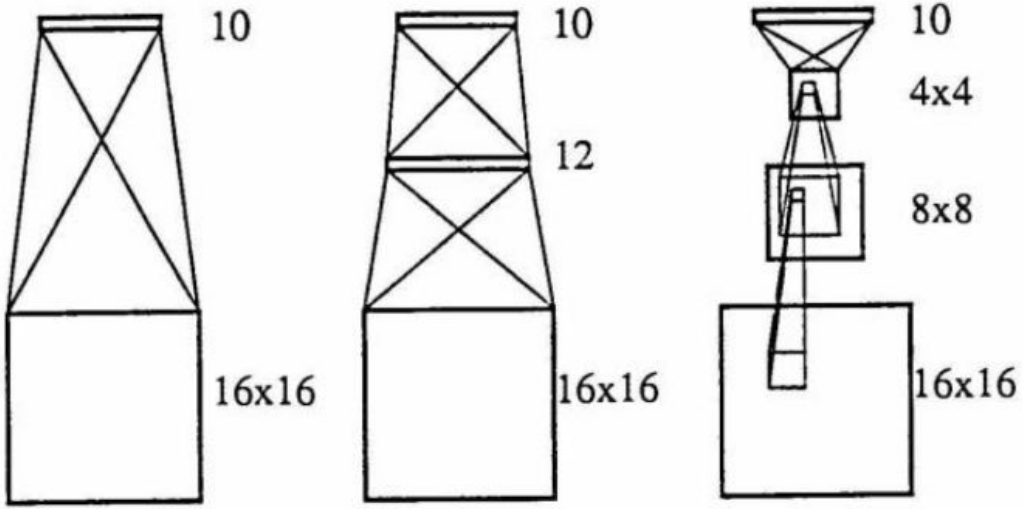


Figure 4: three network architectures Net-1, Net-2 and Net-3

Redes neurais convolucionais (CNN)

- Redes propostas

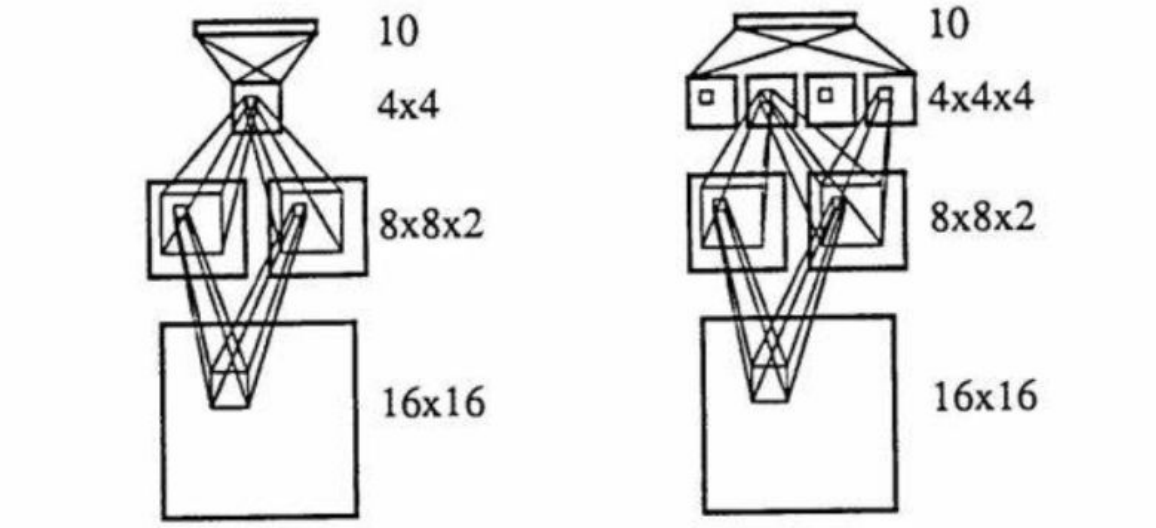
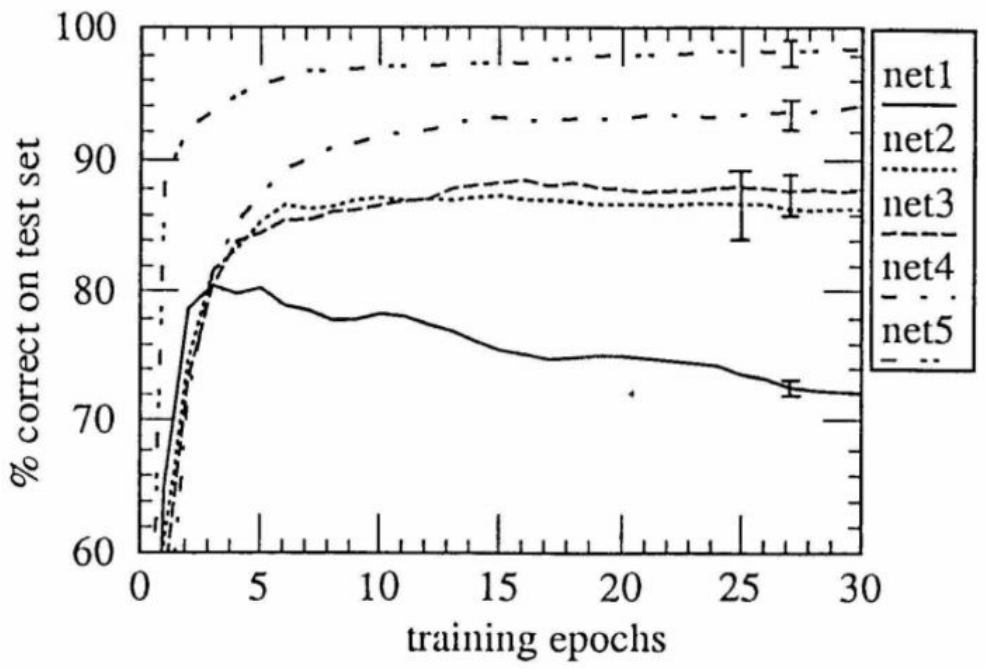


Figure 5 two network architectures with shared weights: Net-4 and Net-5

Redes neurais convolucionais (CNN)

- Resultados



Redes neurais convolucionais (CNN)

- Resultados

network architecture	links	weights	performance
single layer network	2570	2570	80 %
two layer network	3240	3240	87 %
locally connected	1226	1226	88.5 %
constrained network	2266	1132	94 %
constrained network 2	5194	1060	98.4 %

Redes neurais convolucionais (CNN)

- Convoluções?

Redes neurais convolucionais (CNN)

- Convoluções!
  (LeCun, 1990)

Handwritten Digit Recognition with a Back-Propagation Network

Y. Le Cun, B. Boser, J. S. Denker, D. Henderson,
R. E. Howard, W. Hubbard, and L. D. Jackel
AT&T Bell Laboratories, Holmdel, N. J. 07733

ABSTRACT

We present an application of back-propagation networks to handwritten digit recognition. Minimal preprocessing of the data was required, but architecture of the network was highly constrained and specifically designed for the task. The input of the network consists of normalized images of isolated digits. The method has 1% error rate and about a 9% reject rate on zipcode digits provided by the U.S. Postal Service.

Redes neurais convolucionais (CNN)

- Convoluções!
  (LeCun, 1990)

that has a local receptive field, and store the states of this neuron in corresponding locations in a layer called a *feature map* (see figure 3). This operation is equivalent to a convolution with a small size kernel, followed by a squashing function. The process can be performed in parallel by implementing the feature map as a plane

Redes neurais convolucionais (CNN)

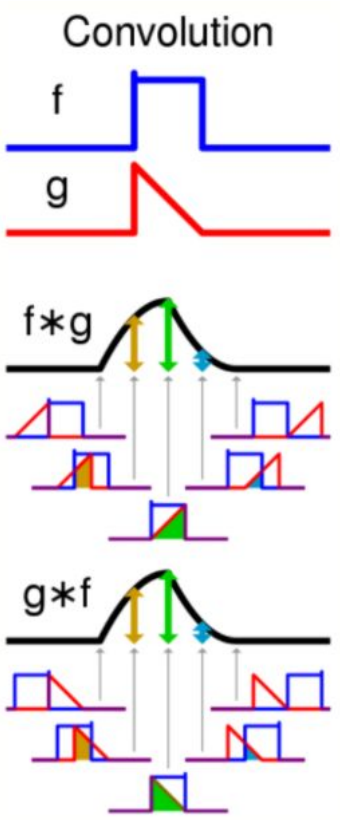
- Convoluções!

  • Definição formal (1D)

$$s(t) = \int x(a)w(t-a)da.$$

$$s(t) = (x * w)(t).$$

$$s(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a).$$



Redes neurais convolucionais (CNN)

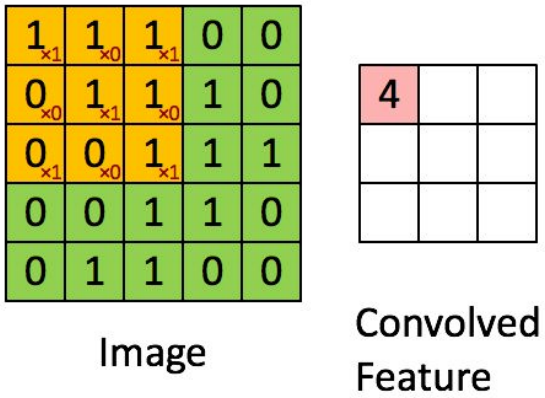
- Convoluções!

- Definição formal (2D)

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n).$$

Redes neurais convolucionais (CNN)

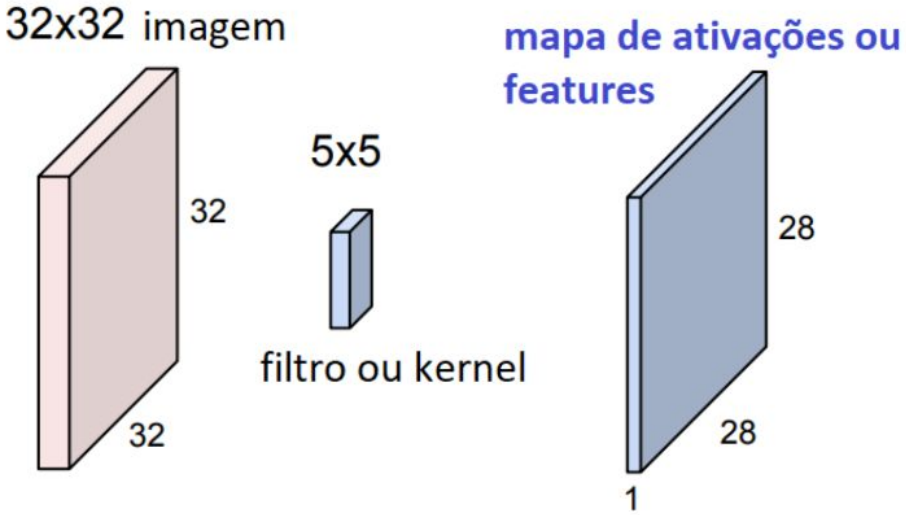
- Convoluções!



Redes neurais convolucionais (CNN)

- Convoluções!

  - Vocabulário



Redes neurais convolucionais (CNN)

- Convoluções!

  - Artigo de cabeceira:

A guide to convolution arithmetic for deep learning

Vincent Dumoulin[1]★ and Francesco Visin[2]★†

★MILA, Université de Montréal
†AIRLab, Politecnico di Milano

Redes neurais convolucionais (CNN)

- LeNet-5 (LeCun, 1998)

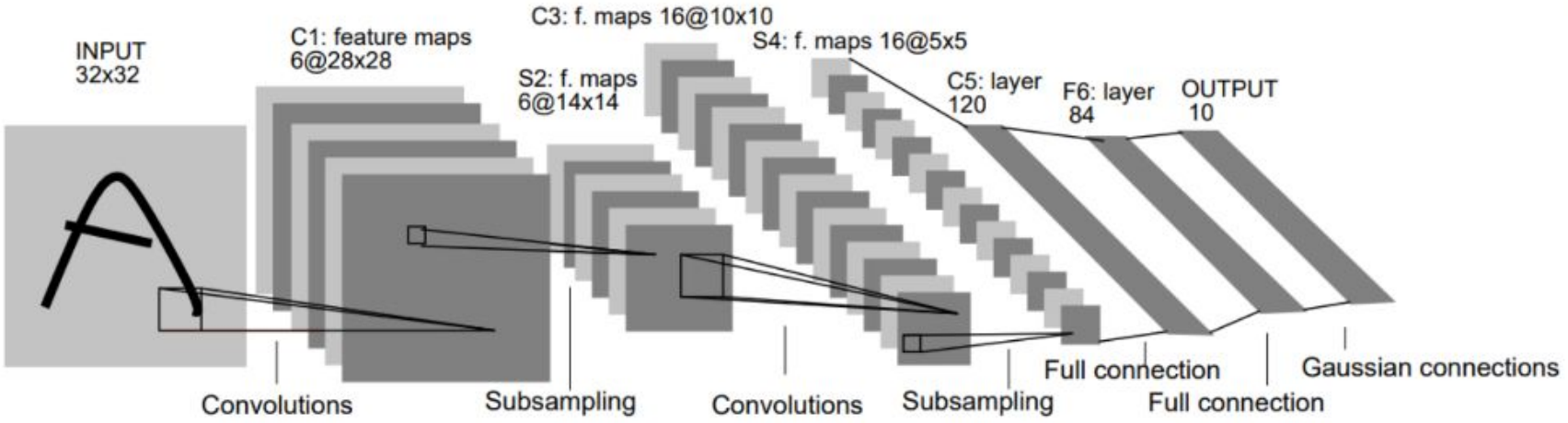
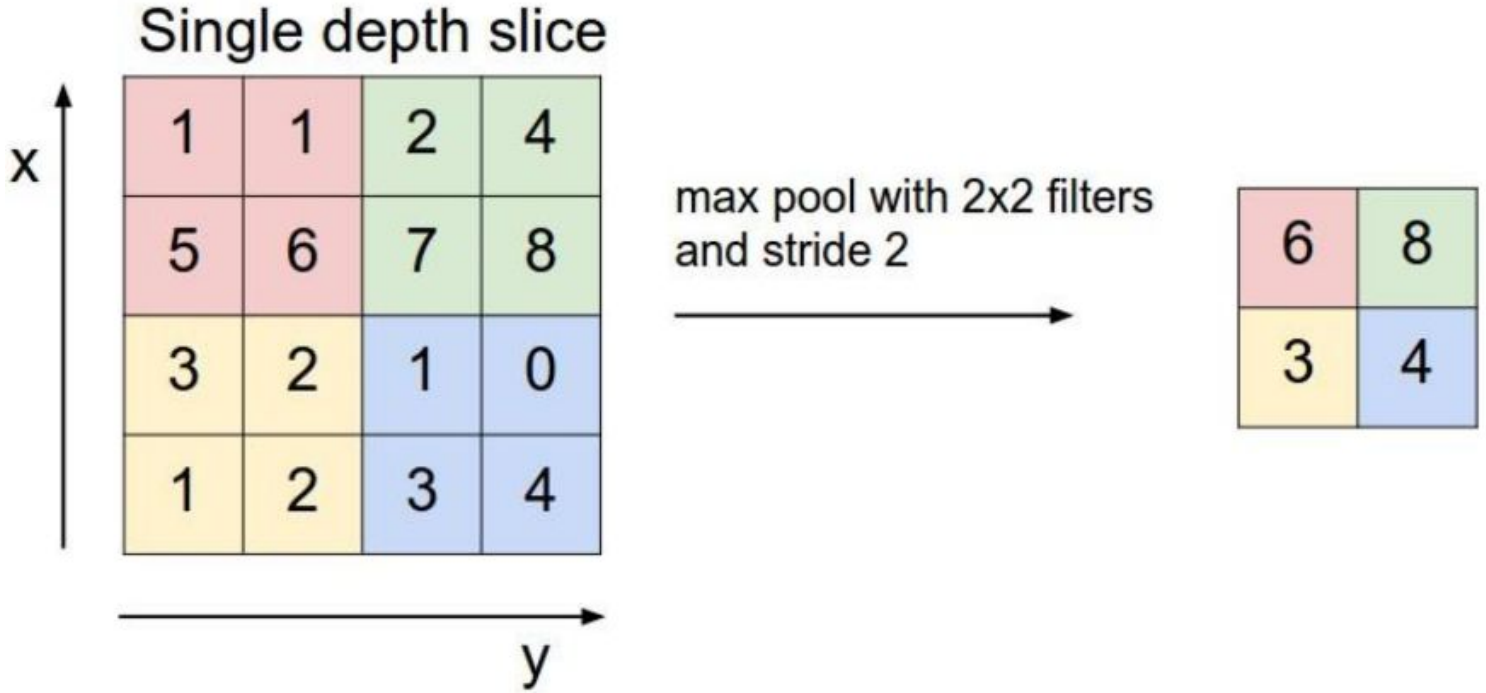


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Redes neurais convolucionais (CNN)

- Pooling ou Subsampling



Redes neurais convolucionais (CNN)

- Pooling ou Subsampling

    - Tipos:
        - MaxPool - seleciona a ativação máxima do kernel
        - AvgPool - calcula a média das ativações do kernel
        - … … … (alta resolução -> operação -> baixa resolução)

Redes neurais convolucionais (CNN)

- Pooling ou Subsampling

    - Justificativas:

Redes neurais convolucionais (CNN)

- Pooling ou Subsampling
  - Justificativas:
    - “Invariância” a pequenas translações

In all cases, pooling helps to make the representation approximately **invariant** to small translations of the input. Invariance to translation means that if we

do not change. See figure 9.8 for an example of how this works. *Invariance to local translation can be a useful property if we care more about whether some feature is present than exactly where it is.* For example, when determining whether an

[Deep Learning Book, Goodfellow]

No próximo episódio...

- Frameworks de desenvolvimento de redes neurais
- Ciclo vicioso de ML