Departamento de Ciência da COMPUTAÇÃO

UDESC Joinville

Redes Neurais Convolucionais

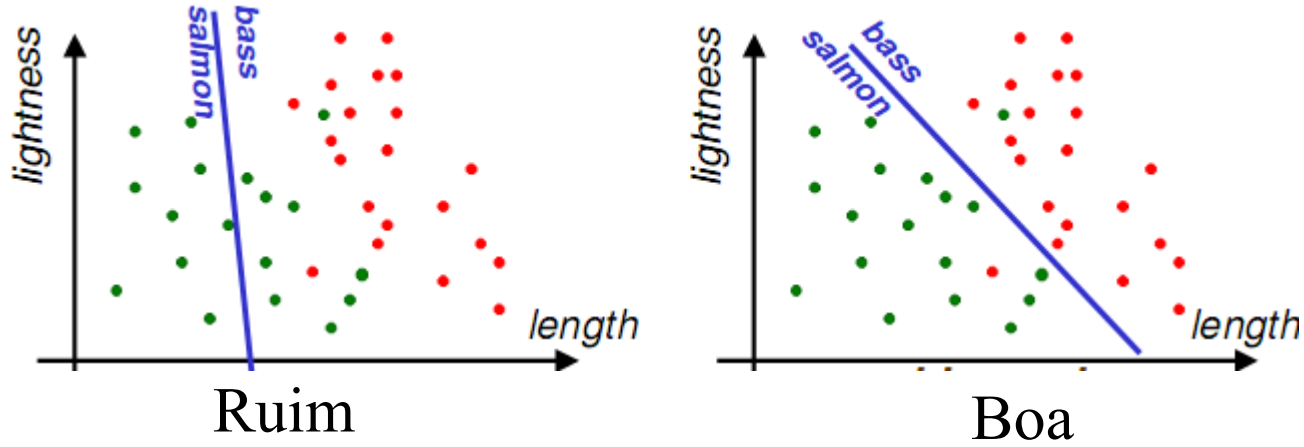
André Tavares da Silva

andre.silva@udesc.br

Roteiro

- Introduzir os o conceito de classificação linear.
- LDA (*Linear Discriminant Analysis*)
- Funções Discriminantes Lineares
- Perceptron
- Introdução à RNA (Rede Neural Artificial)
- Redes Neurais Convolucionais

Introdução



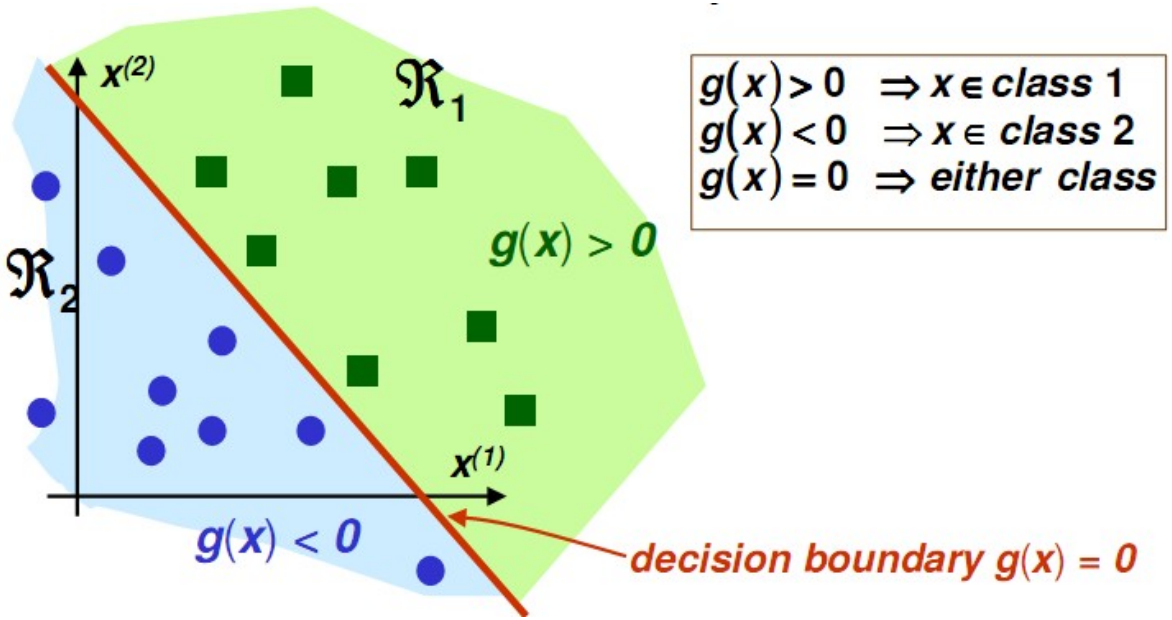
- Suponha duas classes
- Assuma que elas são linearmente separáveis por uma fronteira l(θ)
- Otimizar o parâmetro θ para encontrar a melhor fronteira.
- Como encontrar o parâmetro
  - Minimizar o erro no treinamento
  - O ideal é utilizar uma base de validação.

Funções Discriminante Lineares

- Em geral, uma função discriminante linear pode ser escrita na forma

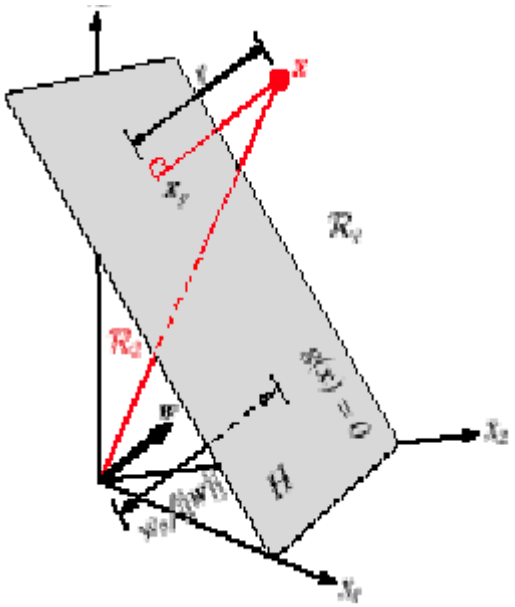
$$g(x) = w^T x + w_0$$

- $\mathbf{w}^T$ é a componente angular e $\mathbf{w_0}$ o linear
- $\mathbf{w}^T$ também pode ser um vetor (características, por exemplo)



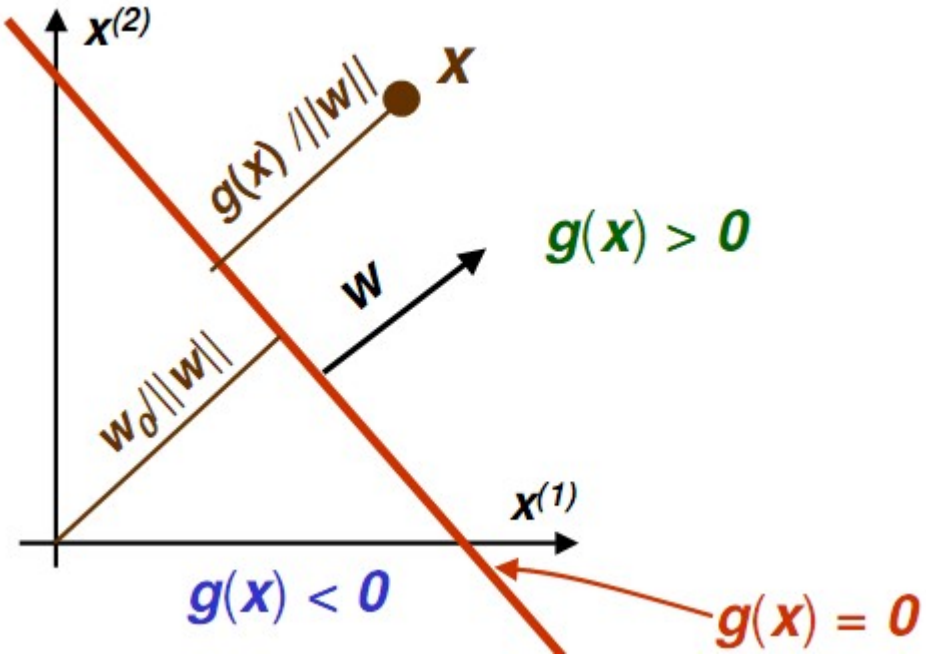
Funções Discriminante Lineares

- $g(x) = w^t x + w_0 = 0$ é um hiperplano
  - Um hiperplano também é
    - Um ponto em 1D
    - Uma reta em 2D
    - Um plano em 3D



Funções Discriminante Lineares

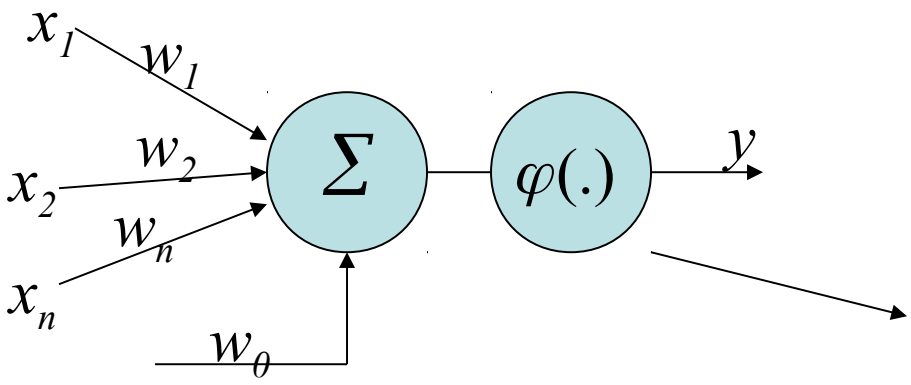
- Para duas dimensões, **w** determina a orientação do hiperplano enquanto $\mathbf{w}_0$ representa o deslocamento com relação a origem



Perceptron

- Um classificador linear bastante simples, mas bastante importante no desenvolvimento das redes neurais é o Perceptron.
  - O perceptron é considerado como sendo a primeira e mais primitiva estrutura de rede neural artificial.
  - Concebido por McCulloch and Pits na década de 50.
- Ele tenta encontrar a melhor fronteira que separa os dados, ou seja, é uma Função Discriminante Linear.

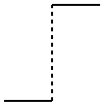
Perceptron



$$y = \varphi\left(\sum w_i \times x_i + w_0\right)$$

Uma função de ativação utilizada no perceptron é a hardlim (threshold)

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



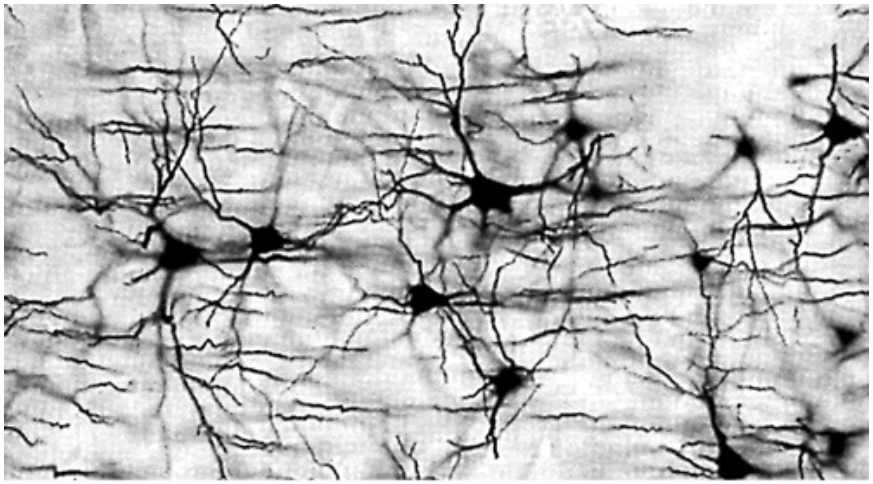
A função de ativação é responsável por determinar a forma e a intensidade de alteração dos valores transmitido de um neurônio a outro.

Perceptron: Algoritmo de Aprendizagem

1. Iniciar os pesos e bias com valores pequenos, geralmente no intervalo [0.3-0.8]
2. Aplicar um padrão de entrada com seu respectivo valor desejado de saída (t_i) e verificar a saída y da rede.
3. Calcular o erro da saída $e = t_j - a$
4. Se e=0, volta ao passo 2
5. Se e<>0,
   1. Atualizar pesos $w_i <- w_i + e \times x_i$
   2. Atualizar o bias $b <- b + e$
6. Voltar ao passo 2

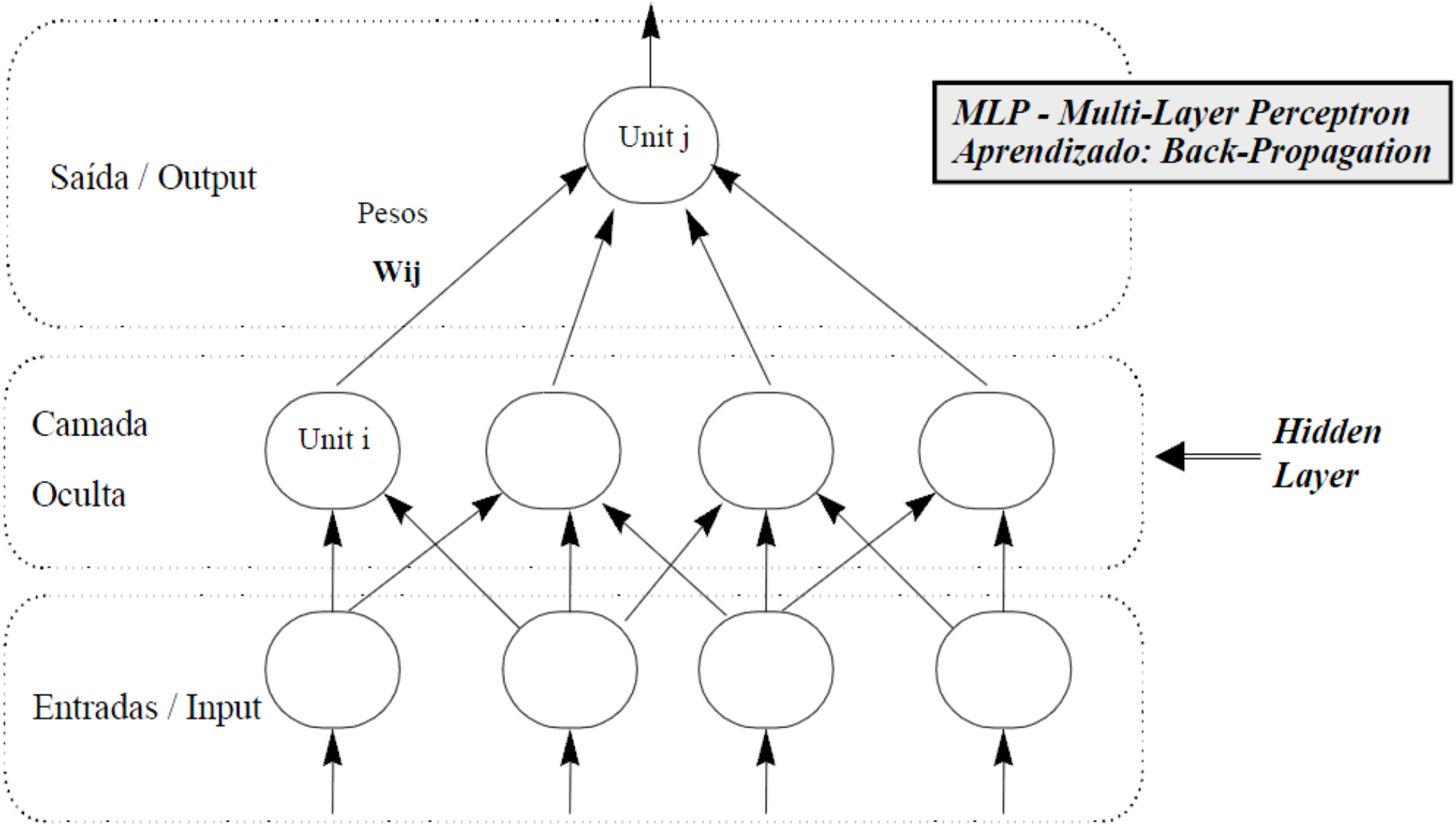
- Critério de parada: Todos os padrões classificados corretamente.

Redes Neurais



- Cérebro humano.
  - Mais fascinante processador existente.
  - Aprox. 10 bilhões de neurônios conectados através de sinapses.
  - Sinapses transmitem estímulos e o resultado pode ser estendido por todo o corpo humano.

Rede MLP


Saída / Output
Pesos
Wij
Unit j
MLP - Multi-Layer Perceptron
Aprendizado: Back-Propagation
Camada
Oculta
Unit i
Hidden
Layer
Entradas / Input

Algoritmo Backpropagation

- O treinamento acontece da seguinte maneira:
  1. Inicializar os valores dos pesos e neurônios aleatoriamente.
  2. Apresentar um padrão a camada de entrada da rede
  3. Encontrar os valores para as camadas escondidas e a camada de saída.
  4. Encontrar o erro da camada de saída.
  5. Ajustar os pesos através da retropropagação dos erros (Backpropagation)
     1. Diminuir o erro a cada iteração
  6. Encontrar o erro na camada escondida
  7. Ajustar os pesos e retorna ao passo 2.

Critério de parada é o erro desejado

Redes Neurais Convolucionais

Operações com imagens

- Filtros
  - Suavização
  - Realce
  - Transformações
- Extração de características
  - Descritores globais e locais

Convolução Discreta 2D

Filtro representado por uma Matriz 3x3:

$$1/9 \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Imagem (5x5):

6	4	5	6	8
9	0	4	8	5
3	2	3	4	2
9	2	3	6	1
7	8	9	0	4

Algoritmo:

```
Para cada pixel da imagem
    •Posicionar centro do filtro sobre o pixel
    •Calcular média ponderada dos pixels vizinhos segundo o filtro
    •pixel correspondente na imagem final ganhará essa média
```

Exemplo no pixel (2,3):

6	4	5	6	8
9	0	4	8	5
3	2	3	4	2
9	2	**3**	6	1
7	8	9	0	4

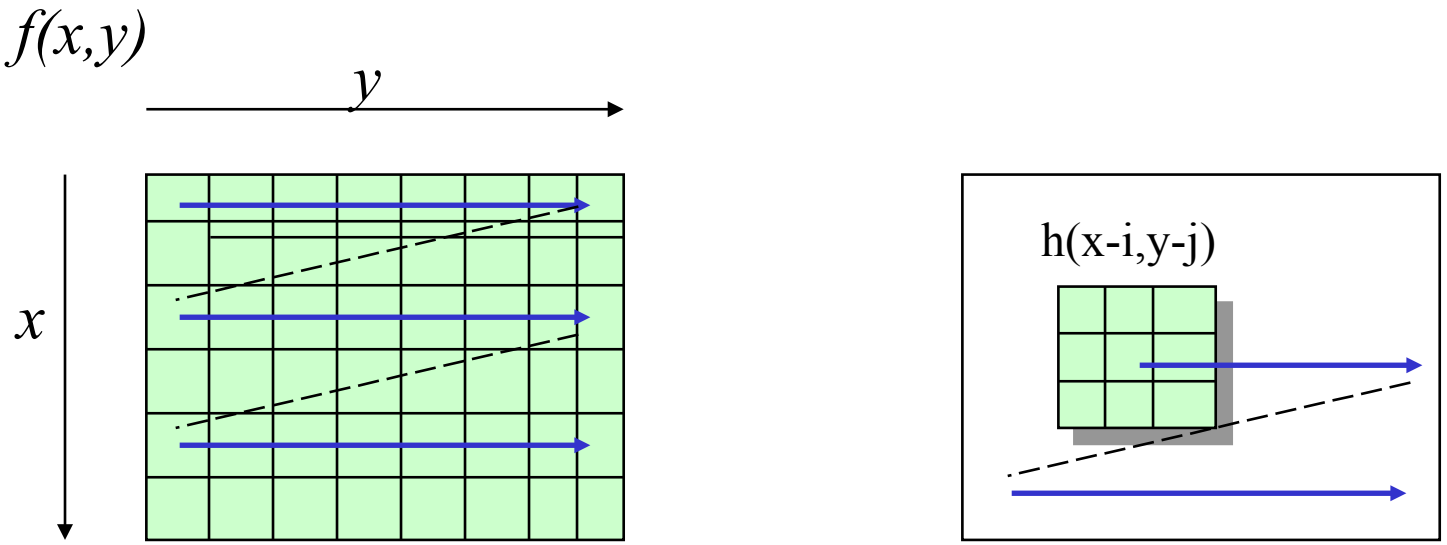
$(2+3+4+2+3+6+8+9+0) / 9 \approx 4$

Considerações:

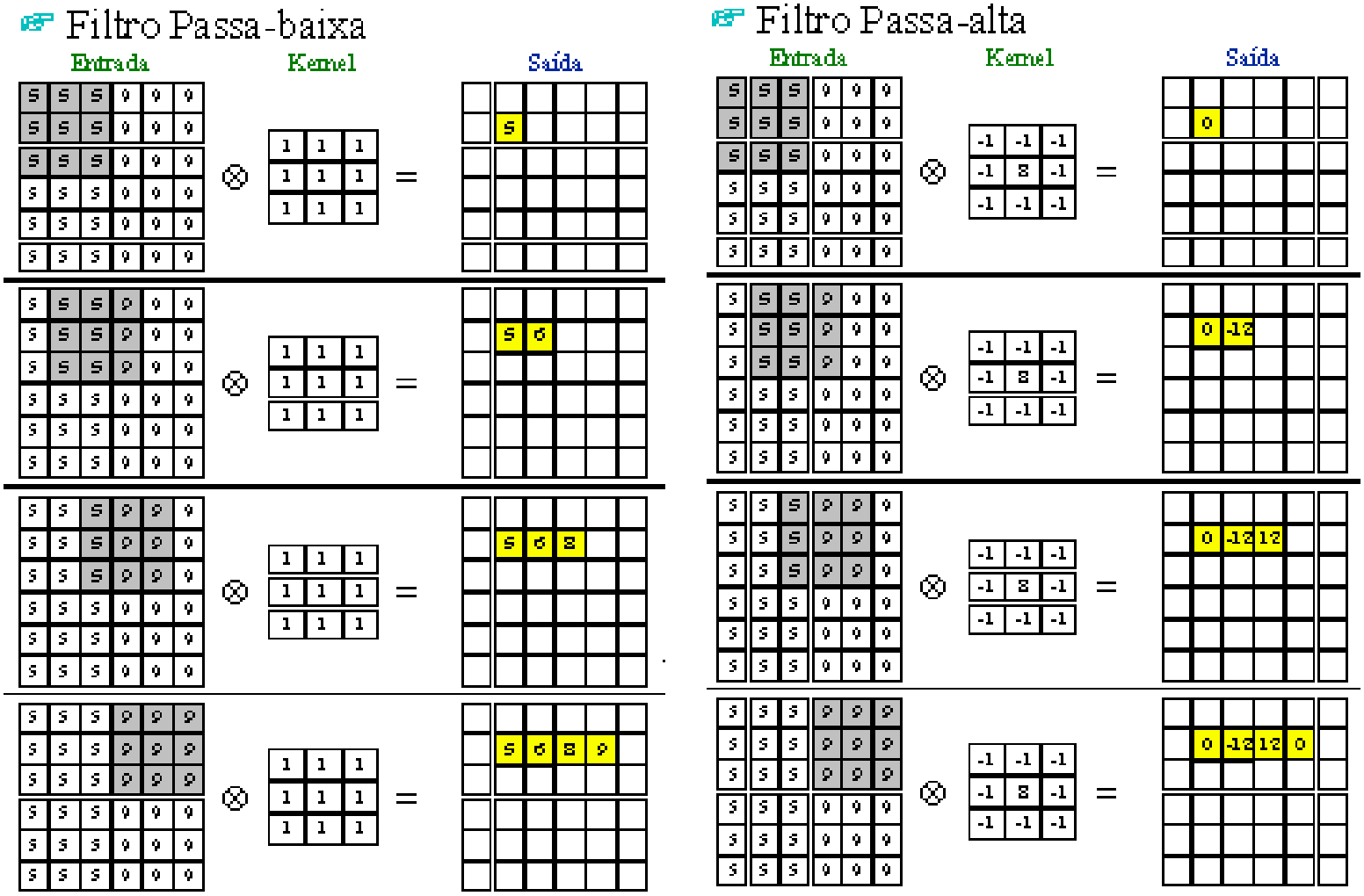
- Complexidade?
- Valores Negativos?
- O que fazer na borda?

- Convolução:

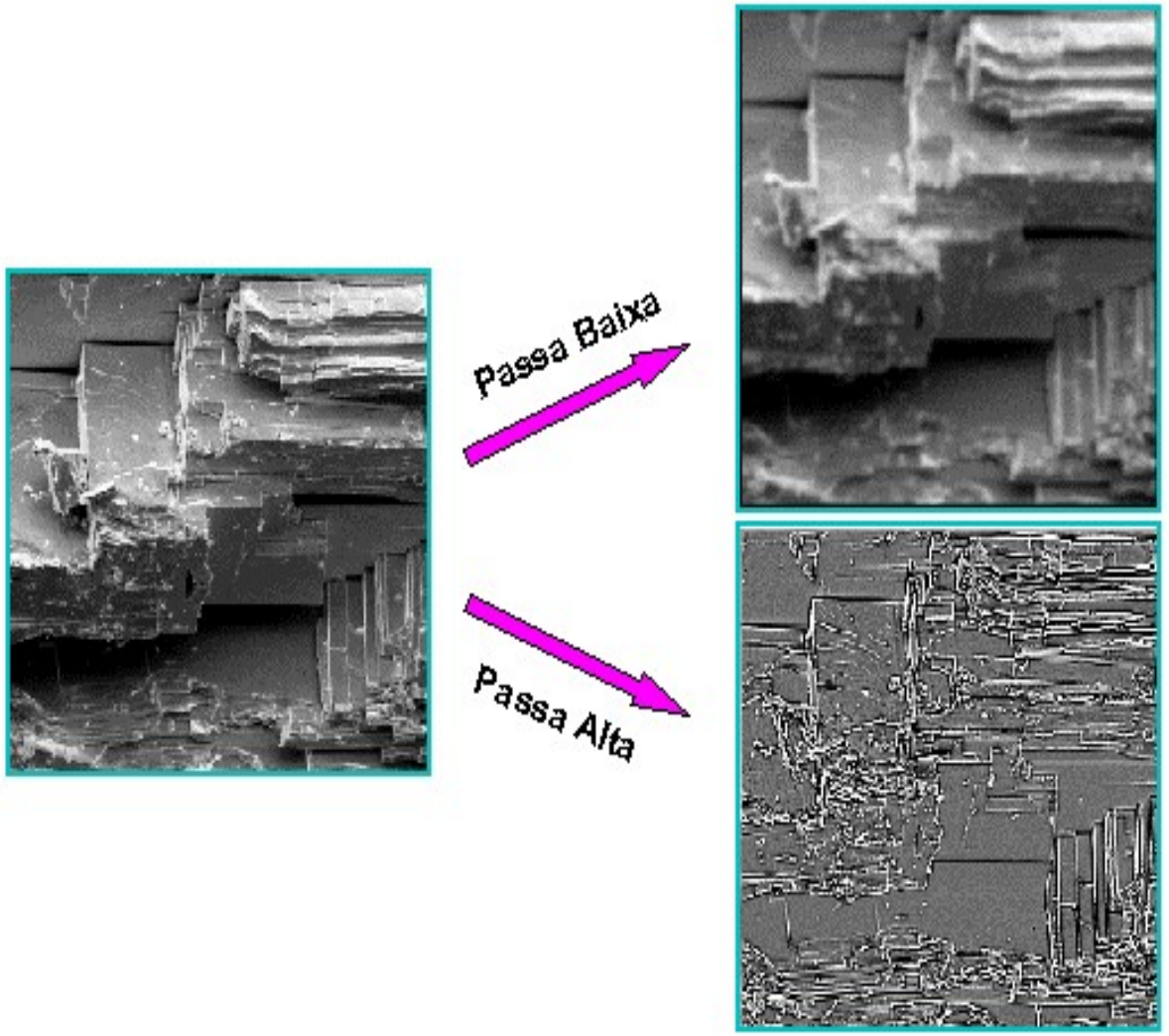
$$g(x,y) = \sum_{i=-m}^{m} \sum_{j=-n}^{n} f(i,j)h(x-i, y-j)$$



Convolução


Filtro Passa-baixa
Entrada
Kernel
Saída
Filtro Passa-alta
Entrada
Kernel
Saída

Filtragem



Convolução

- Filtros (máscaras) comuns:
  - Média (passa baixa)
  - Laplaciana (passa alta)
  - Gaussiana
  - Sobel
  - Realce
  - Nitidez (*sharpen*)
  - Artísticos

Média ou Box

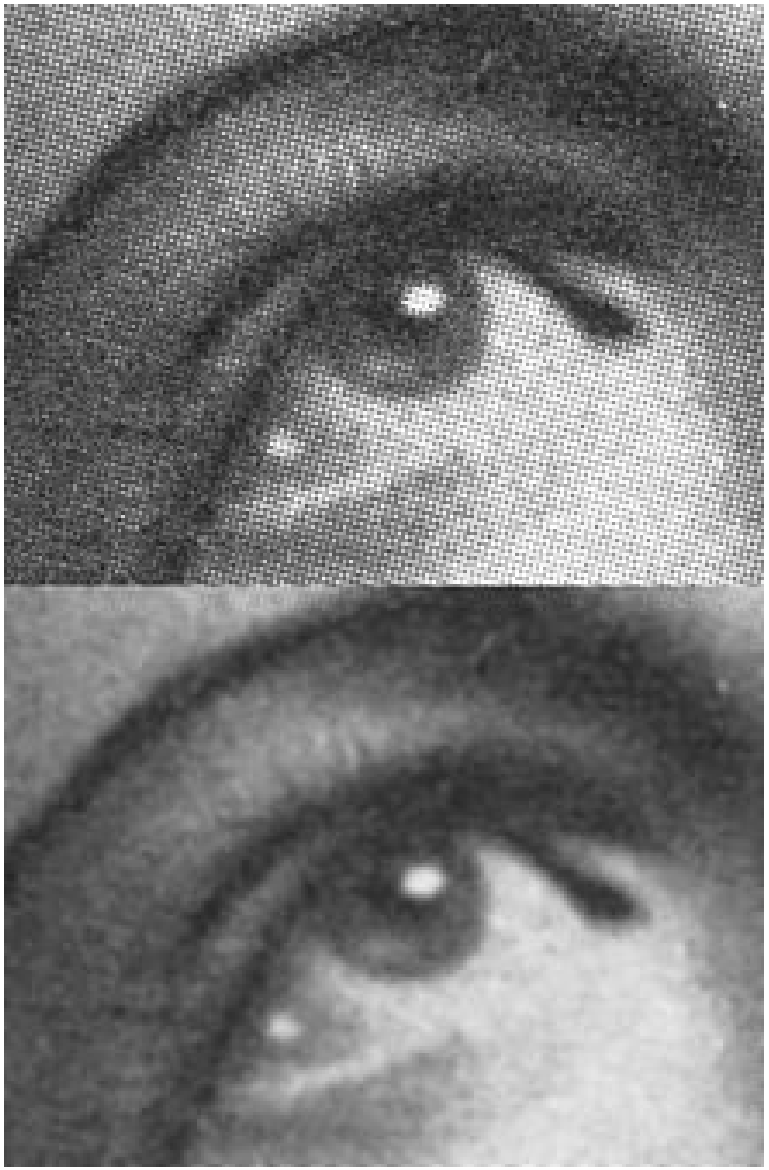
- O que acontece se fizermos uma convolução usando uma máscara como essa abaixo?

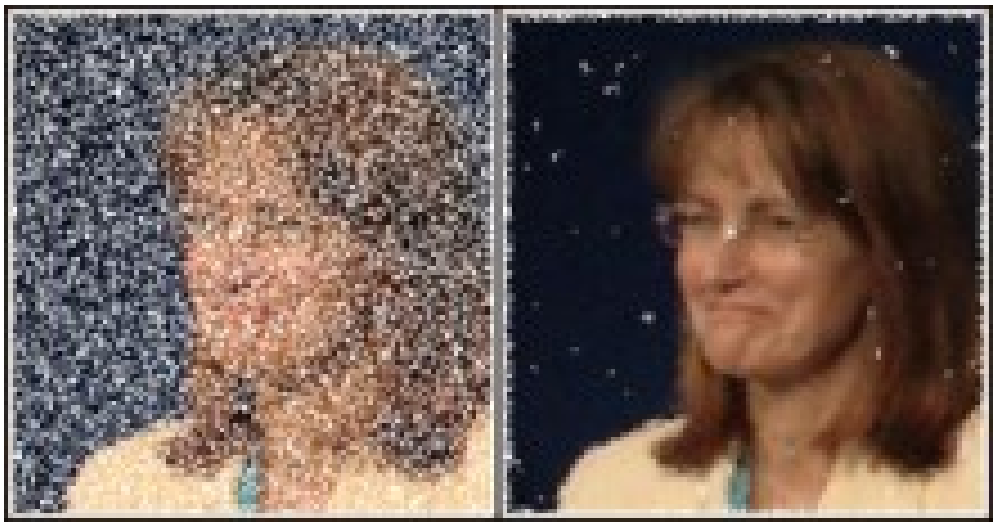
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

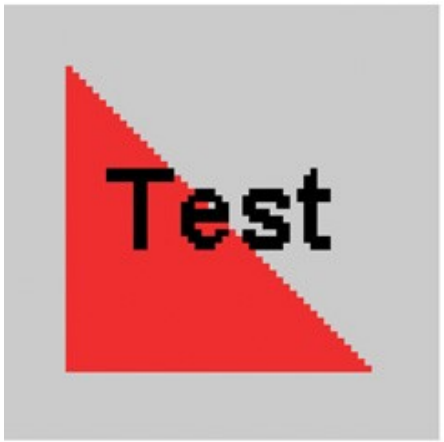
Departamento de Ciência da COMPUTAÇÃO

UDESC Joinville

Resultado







(a)

(b)

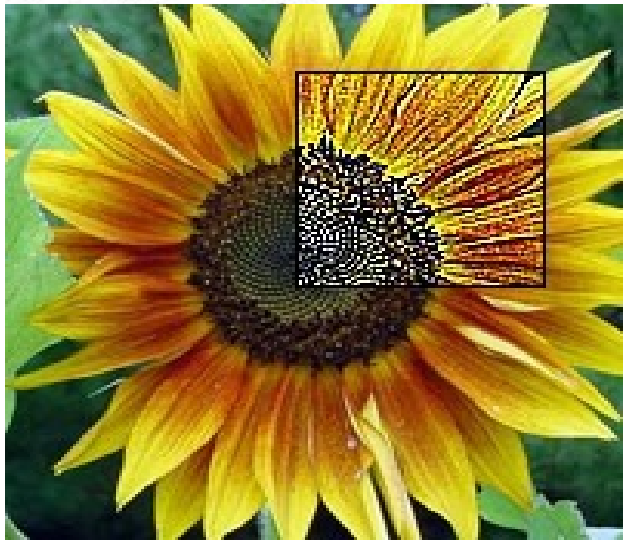
Resultado (máscara 5x5)

Nitidez (*sharpen*)

- O que acontece se fizermos uma convolução usando uma máscara como essa abaixo?

-1	-1	-1
-1	9	-1
-1	-1	-1

Resultado

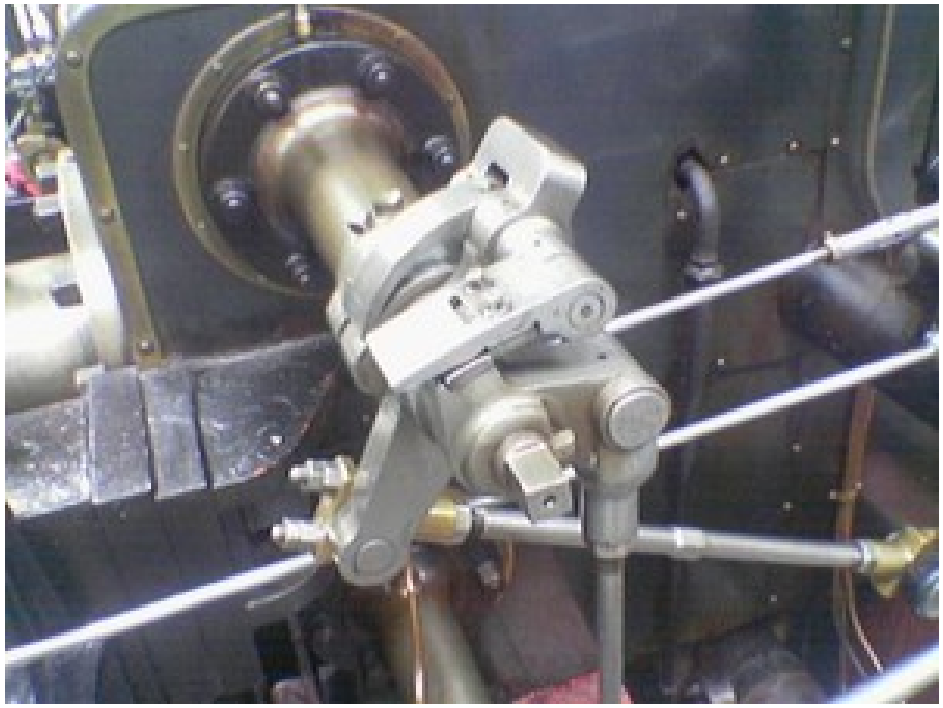


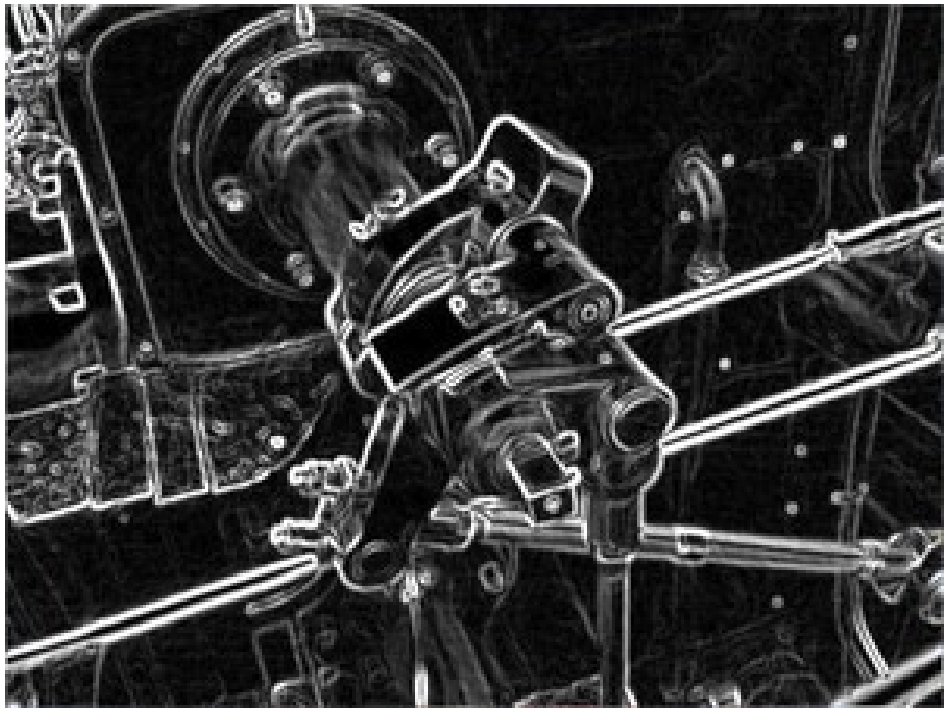
Resultado

Sobel

$$Gx = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

Sobel (borda)





Outros

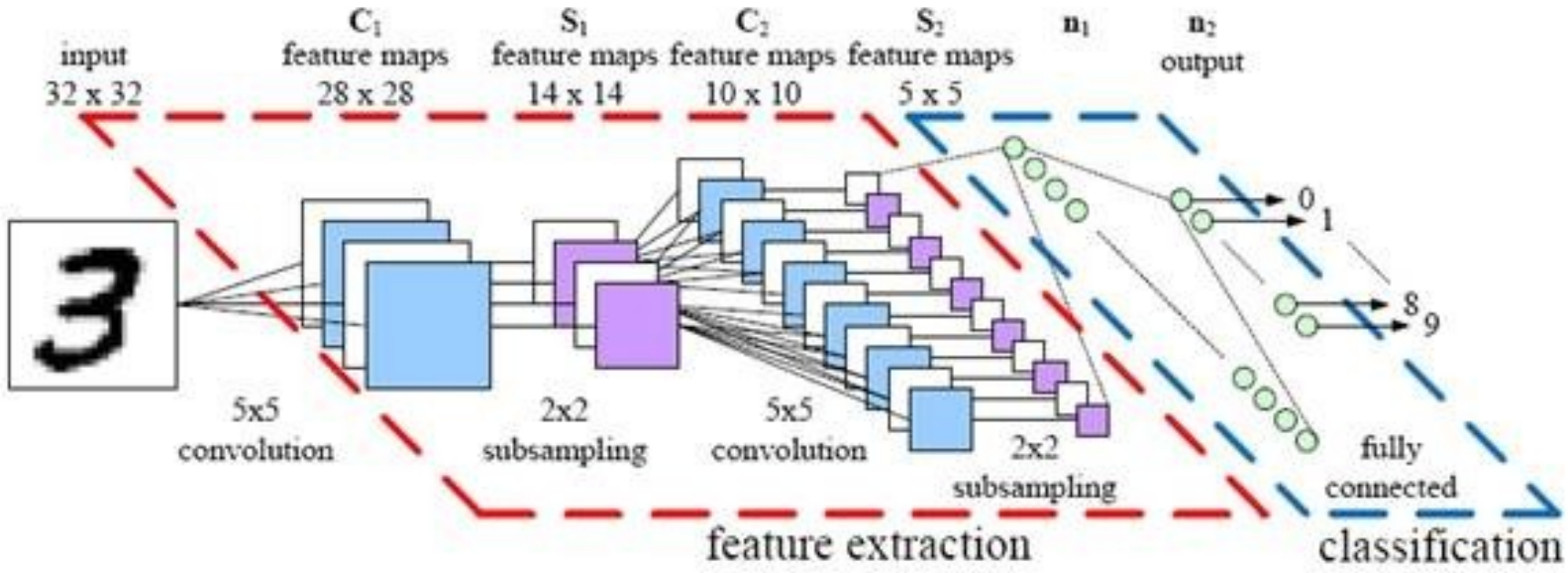
Departamento de Ciência da COMPUTAÇÃO

UDESC Joinville

Rede CNN

(*Convolutional Neural Network*)

Rede CNN (LENET-5)



PEEMEN, Maurice; MESMAN, Bart; CORPORAAL, Henk. **Speed sign detection and recognition by convolutional neural networks**. In: Proceedings of the 8th International Automotive Congress. 2011. p. 162-170.

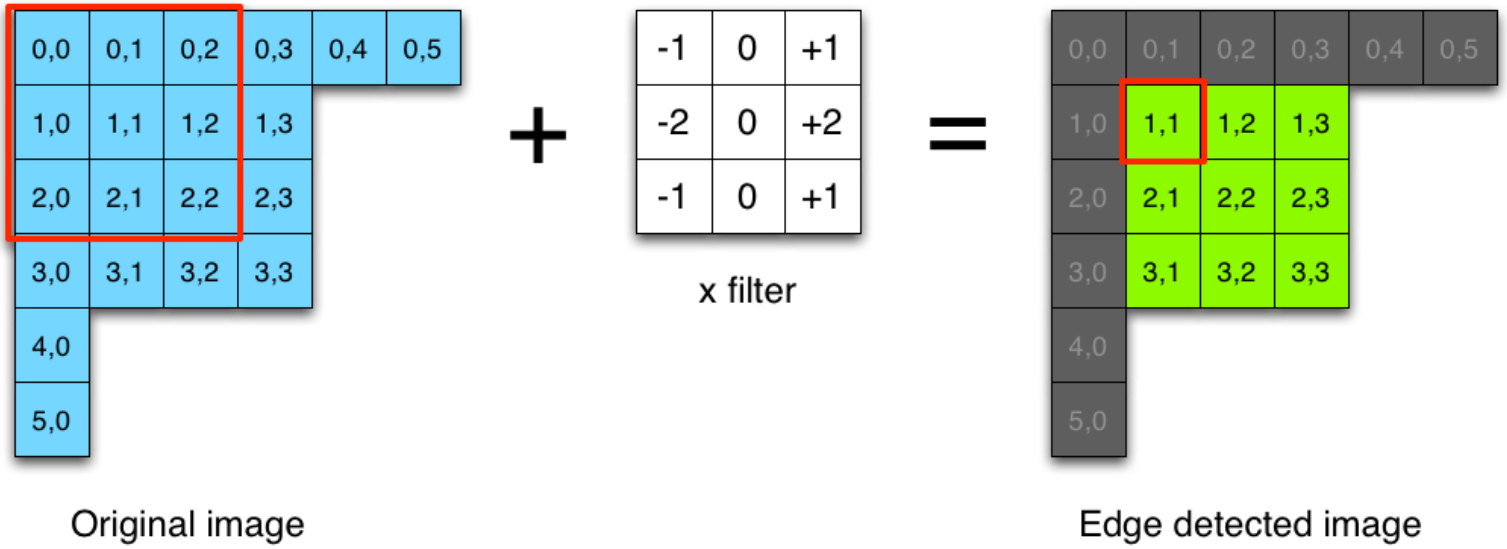
CNN

- Inspiradas no modelo biológico da visão
- Usa conceito de Redes Multi-Camadas
- Idealizada no ínicio do anos 90 [Lecun], e vasta aplicação após 2006 devido a "popularização" de GPU's
- Treinamento requer alto custo computacional e uma base de dados grande (podendo necessitar de aumento de dados artificiais)

Camadas

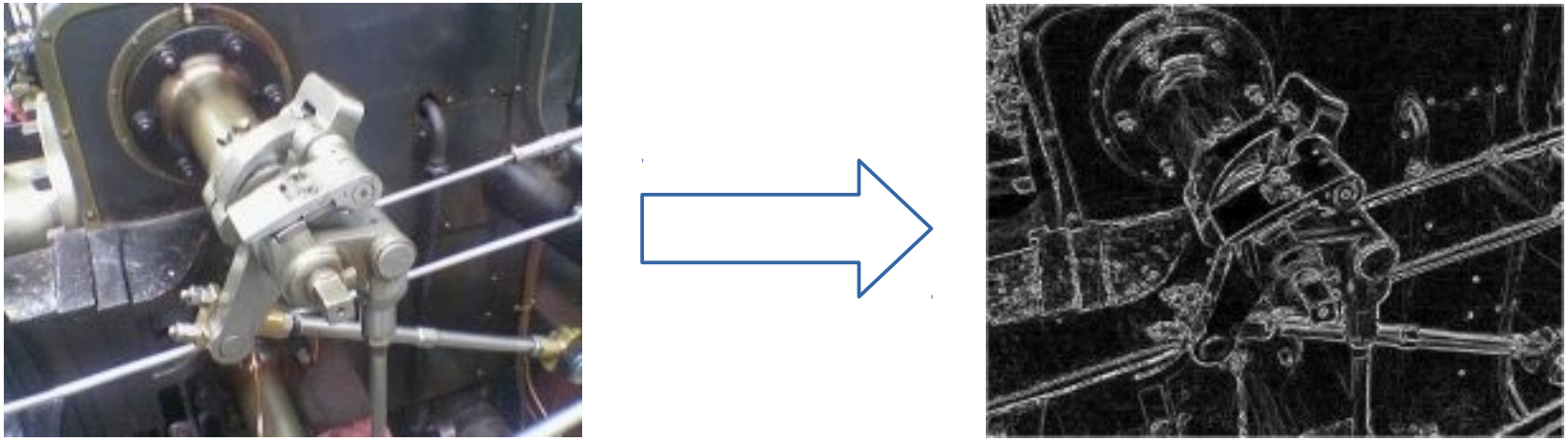
- Convolucional : Definem os filtros
  (Aprendizado / BackPropagation)
- Ativação: Neurônios
  (Relu / Sigmoid / TangH)
- ReLU (*Rectified Linear Units*)
  (max(0,x), tanh(x), sigmoid,...)
- Pooling : Reduzem as escalas
  (Max, Media, etc..)
- Fully-Connected (FC): Camada que determina as classes (Classificador)

Convolução

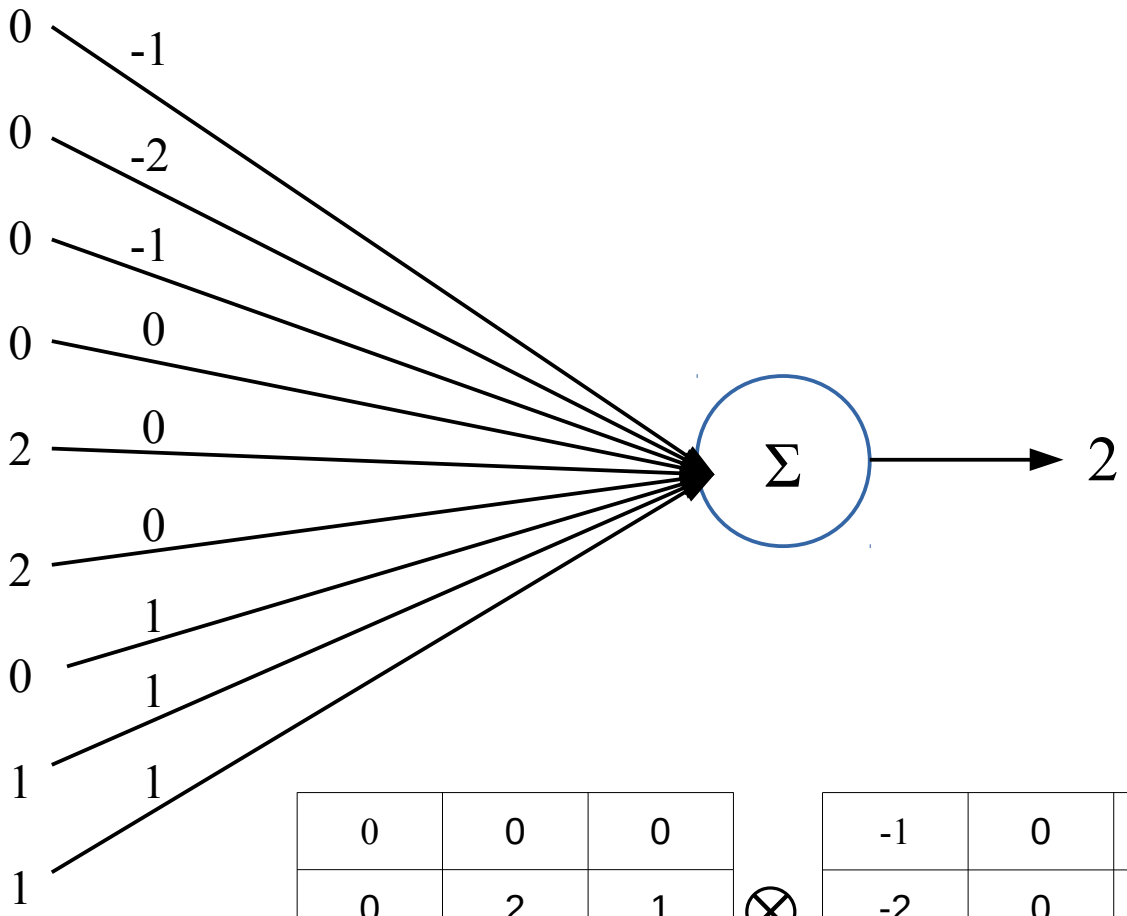


Original image

Edge detected image



Convolução



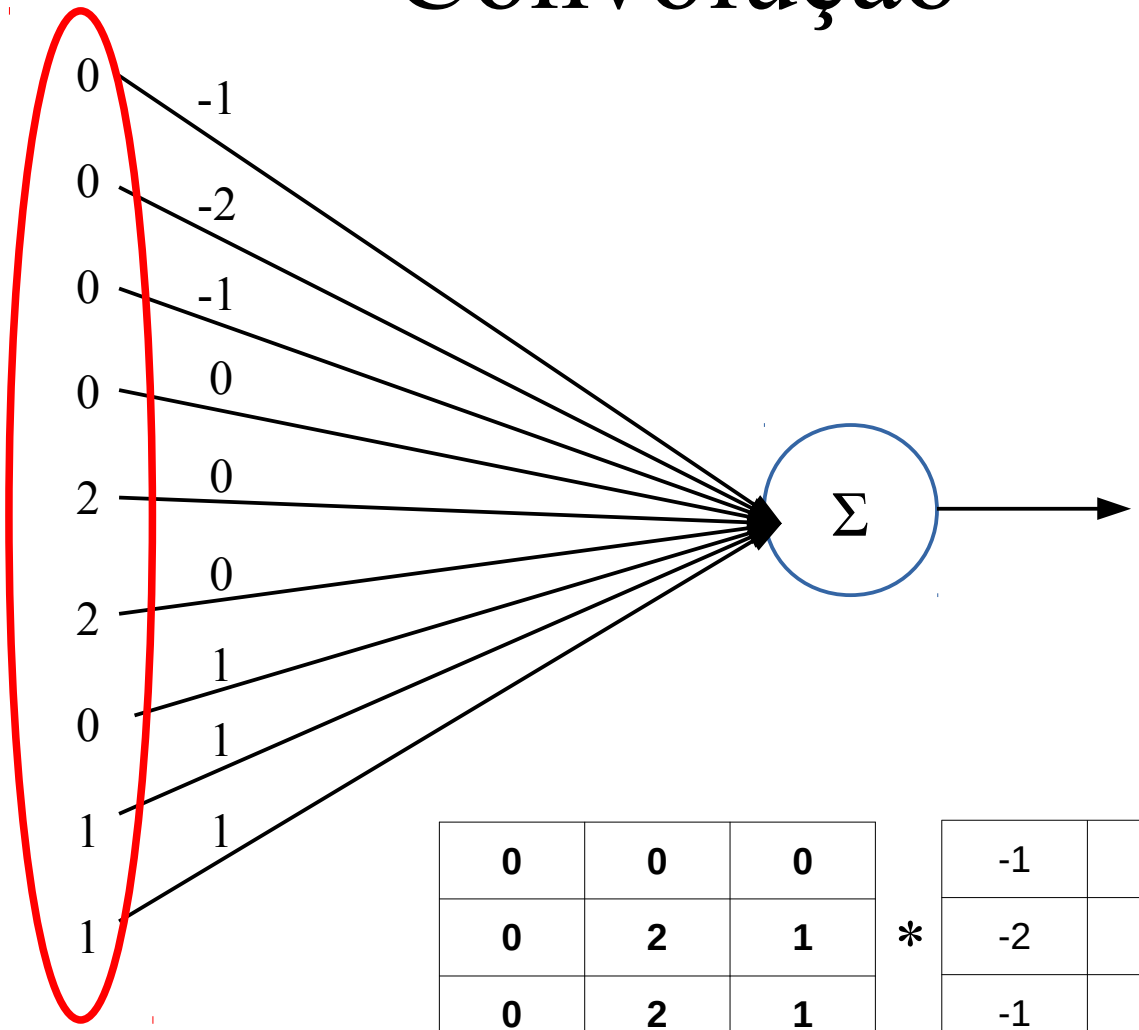
0	0	0
0	2	1
0	2	1

$\otimes$

-1	0	1
-2	0	1
-1	0	1

$= 2$

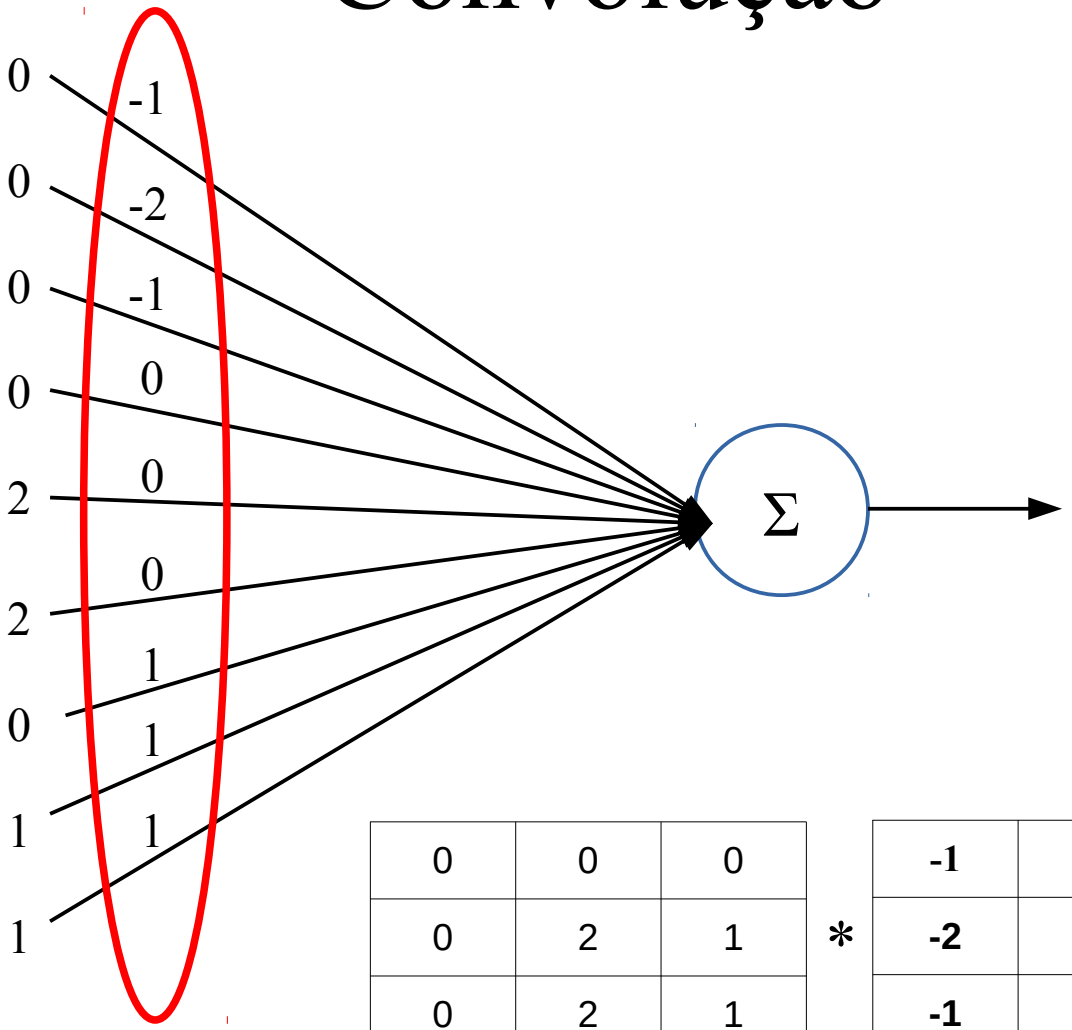
Convolução



0	**0**	**0**		-1	0	1
0	**2**	**1**	*	-2	0	1
0	**2**	**1**		-1	0	1

Padrão de entrada (pixels)

Convolução



0	0	0		*	**-1**	**0**	**1**
0	2	1			**-2**	**0**	**1**
0	2	1			**-1**	**0**	**1**

Pesos (kernel)

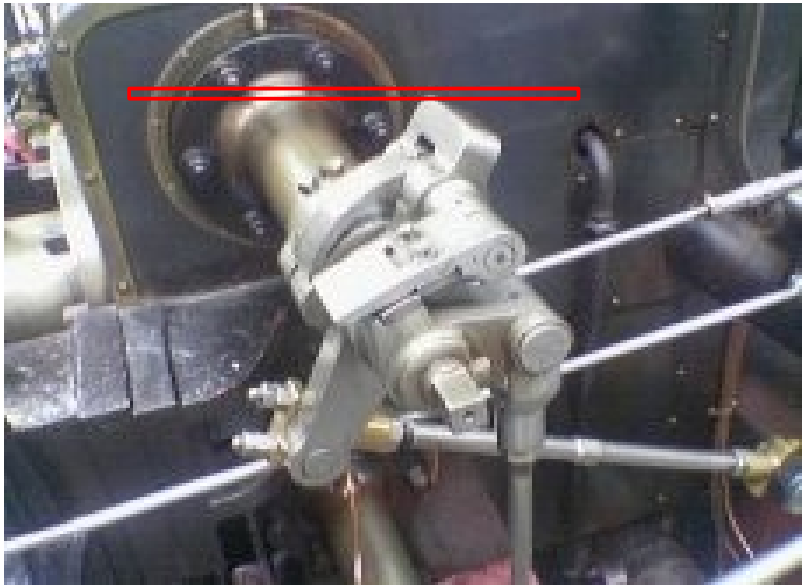
Convolução

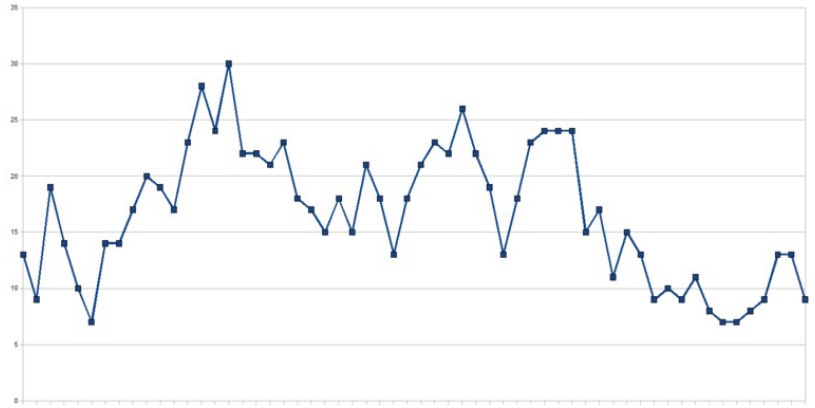
- Todos os neurônios desta camada compartilham os mesmos pesos a fim de realizar uma convolução (como vimos anteriormente);
- Todos os neurônios detectam a mesma característica (bordas, por exemplo) em todas as posições da imagem. Isso pode ser realizado em paralelo, se for utilizada uma GPU, por exemplo;
- O número de parâmetros livres é reduzido, reduzindo o processamento no aprendizado.

Convolução

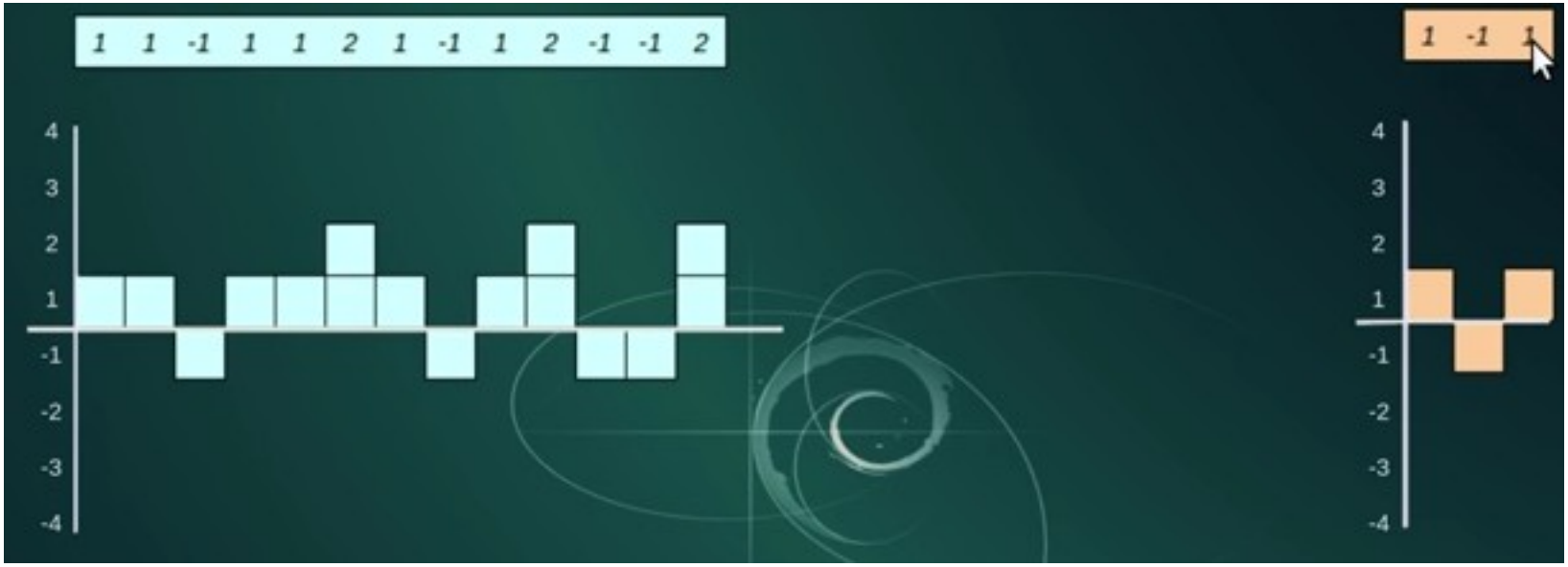
- O número de convoluções a serem utilizadas é um parâmetro da rede;
- Em vez de testarmos diferentes *kernels* (tipos de convolução), a rede vai aprender quais as melhores configurações de "filtros" serão utilizados nas imagens;
- Em uma imagem colorida, o filtro é aplicado em cada um dos canais. Podendo gerar uma saída para cada canal ou realizada uma operação (média) para reduzir o processamento (tamanho da rede).

Camada Convolucional

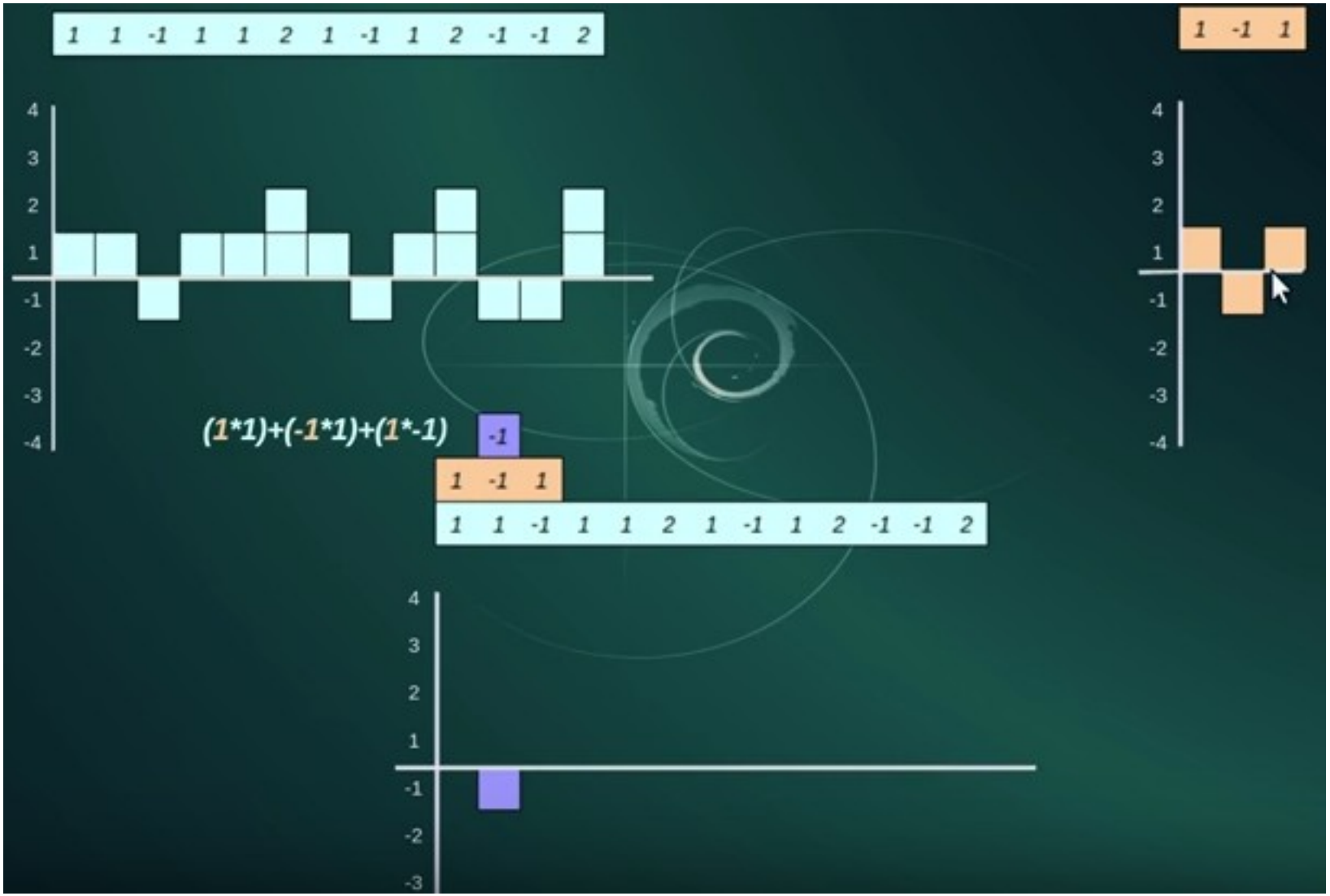




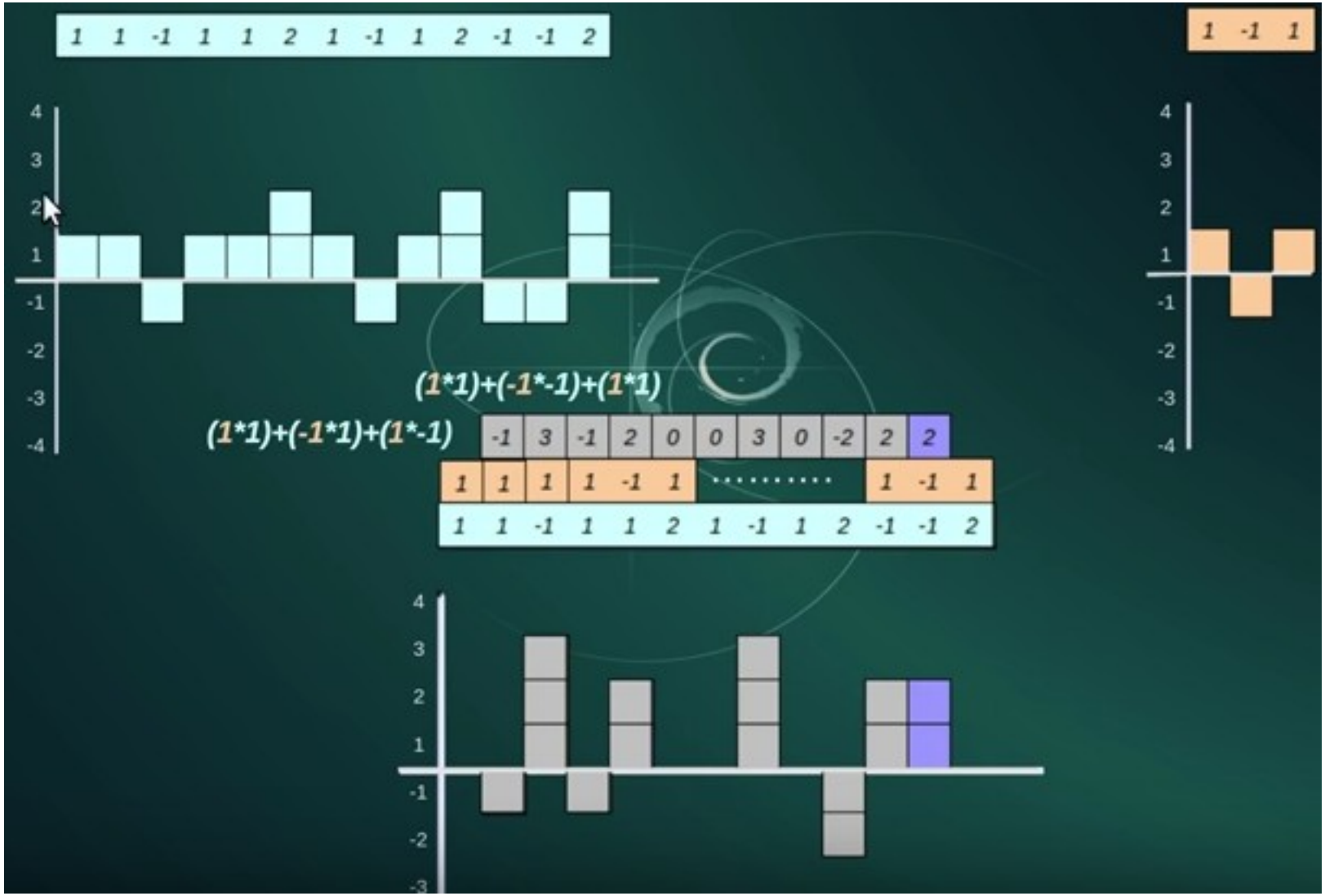
Camada Convolucional



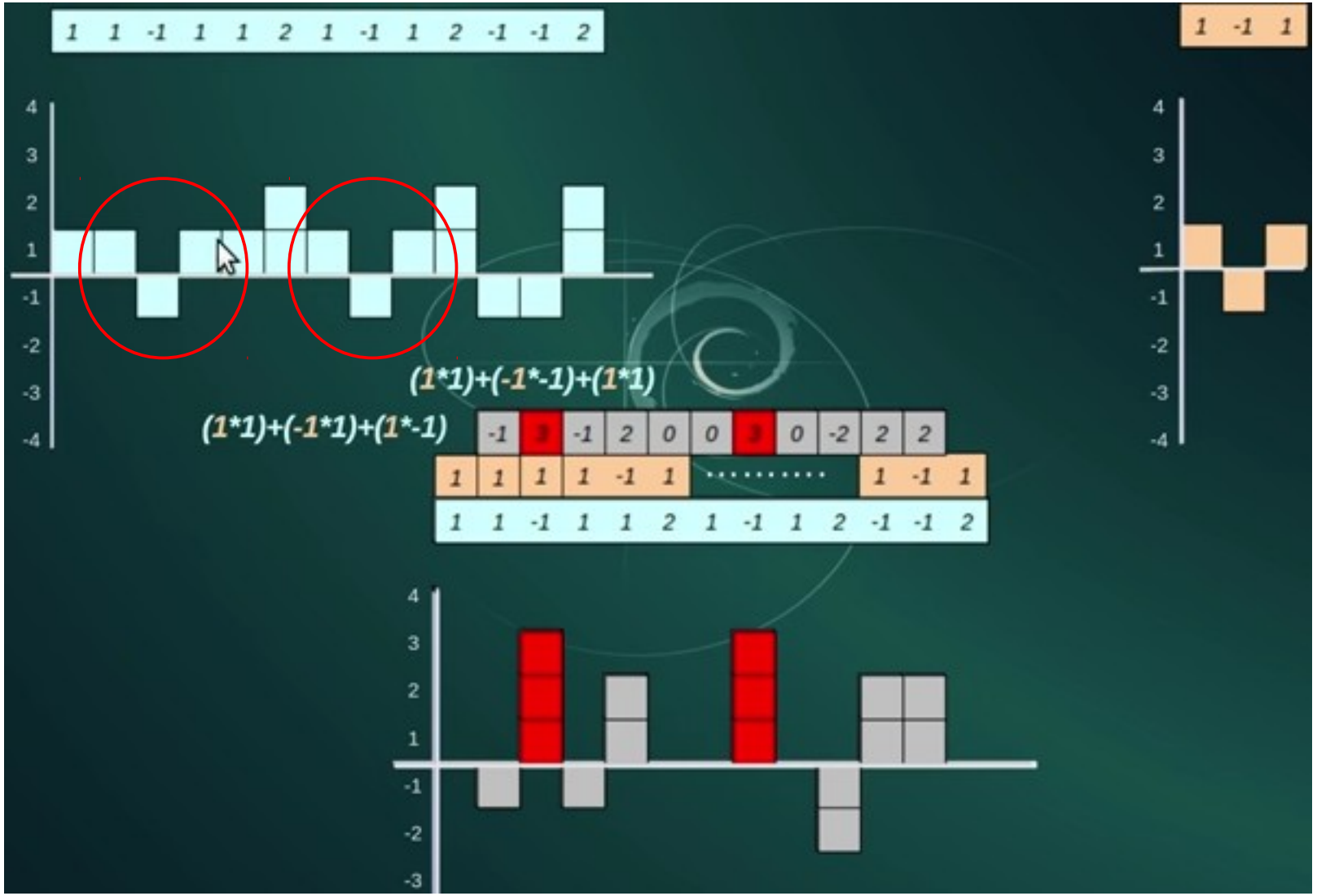
Camada Convolucional



Camada Convolucional

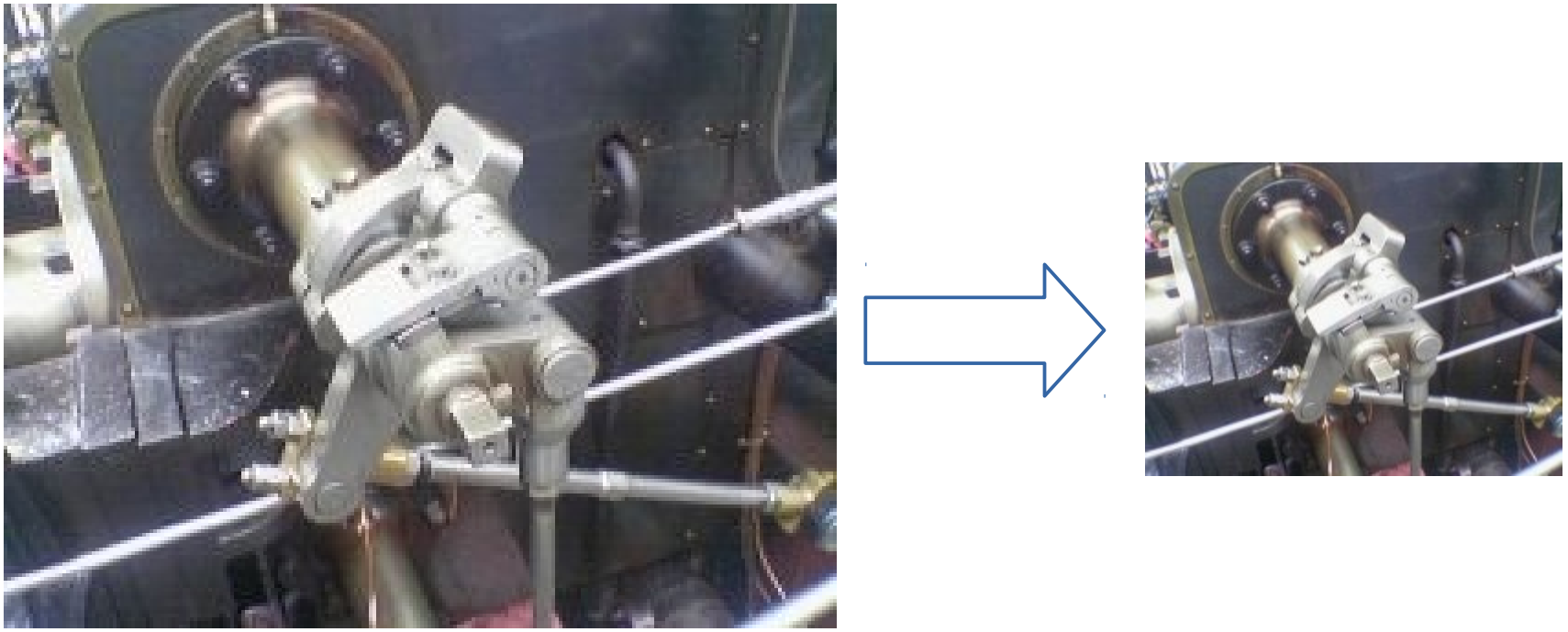


Camada Convolucional



Pooling

- Os pooling's são necessários para reduzir a quantidade de características por filtro (redução de escala). A Subamostragem dos pixels não altera os objetos.



ReLU (*Rectified Linear Units*)

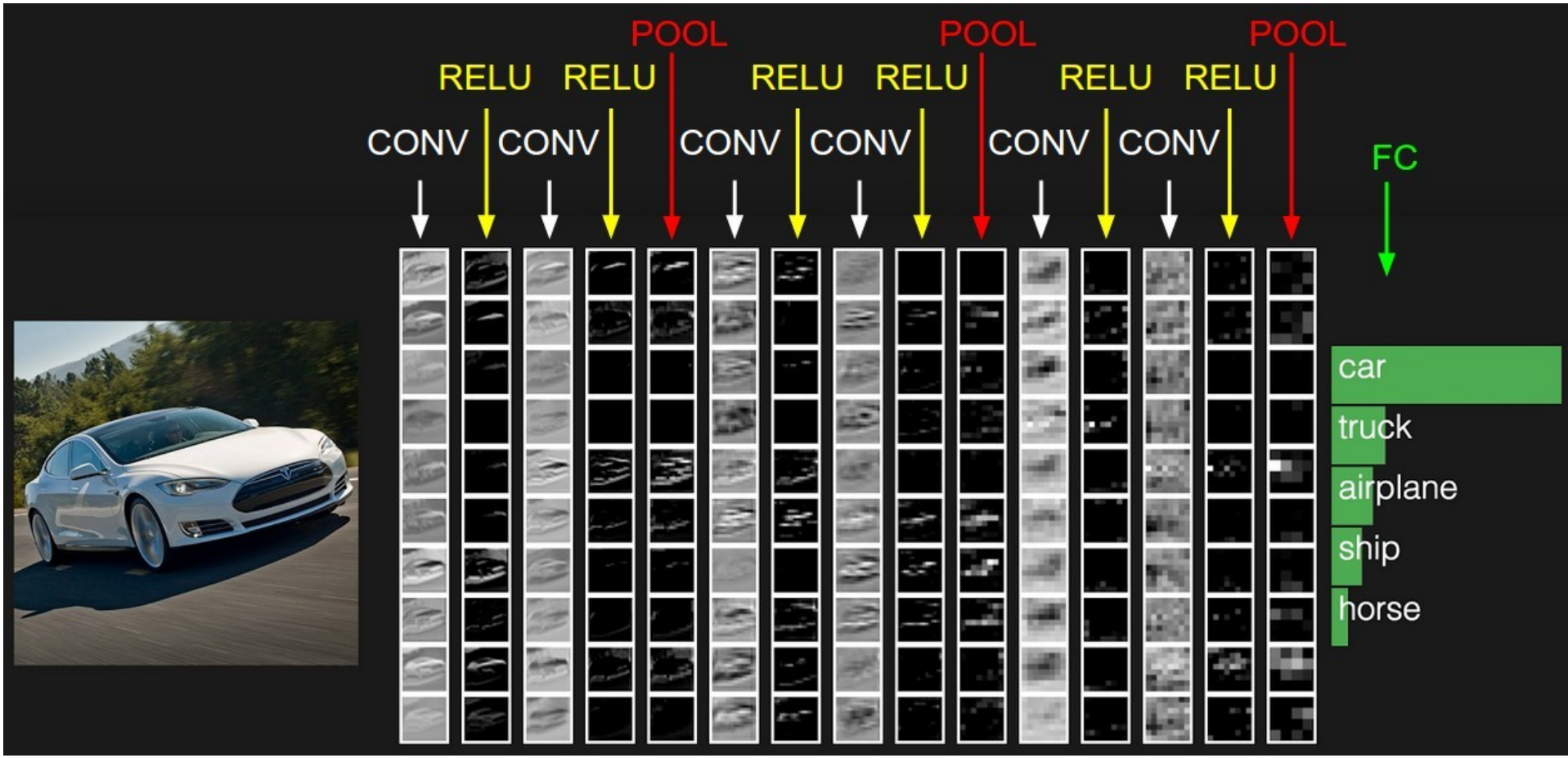
- Camada que aplica uma função de ativação sem saturação, aumentando as propriedades não lineares da função de decisão para toda a rede sem afetar os valores obtidos pela camada convolucional;
- Normalmente é utilizada a função ReLU $f(x)=max(0,x)$, mas podem ser usadas funções como $f(x)=|tanh(x)|$ ou $f(x)=(1+e^{-x})^{-1}$ (função sigmoide). A ReLU é favorita por ser muitas vezes mais rápida que as outras.

ReLU (*Rectified Linear Units*)

- Camada que aplica uma função de ativação sem saturação, aumentando as propriedades não lineares da função de decisão para toda a rede sem afetar os valores obtidos pela camada convolucional;
- Normalmente é utilizada a função ReLU $f(x)=max(0,x)$, mas podem ser usadas funções como $f(x)=|tanh(x)|$ ou $f(x)=(1+e^{-x})^{-1}$ (função sigmoide). A ReLU é favorita por ser muitas vezes mais rápida que as outras.

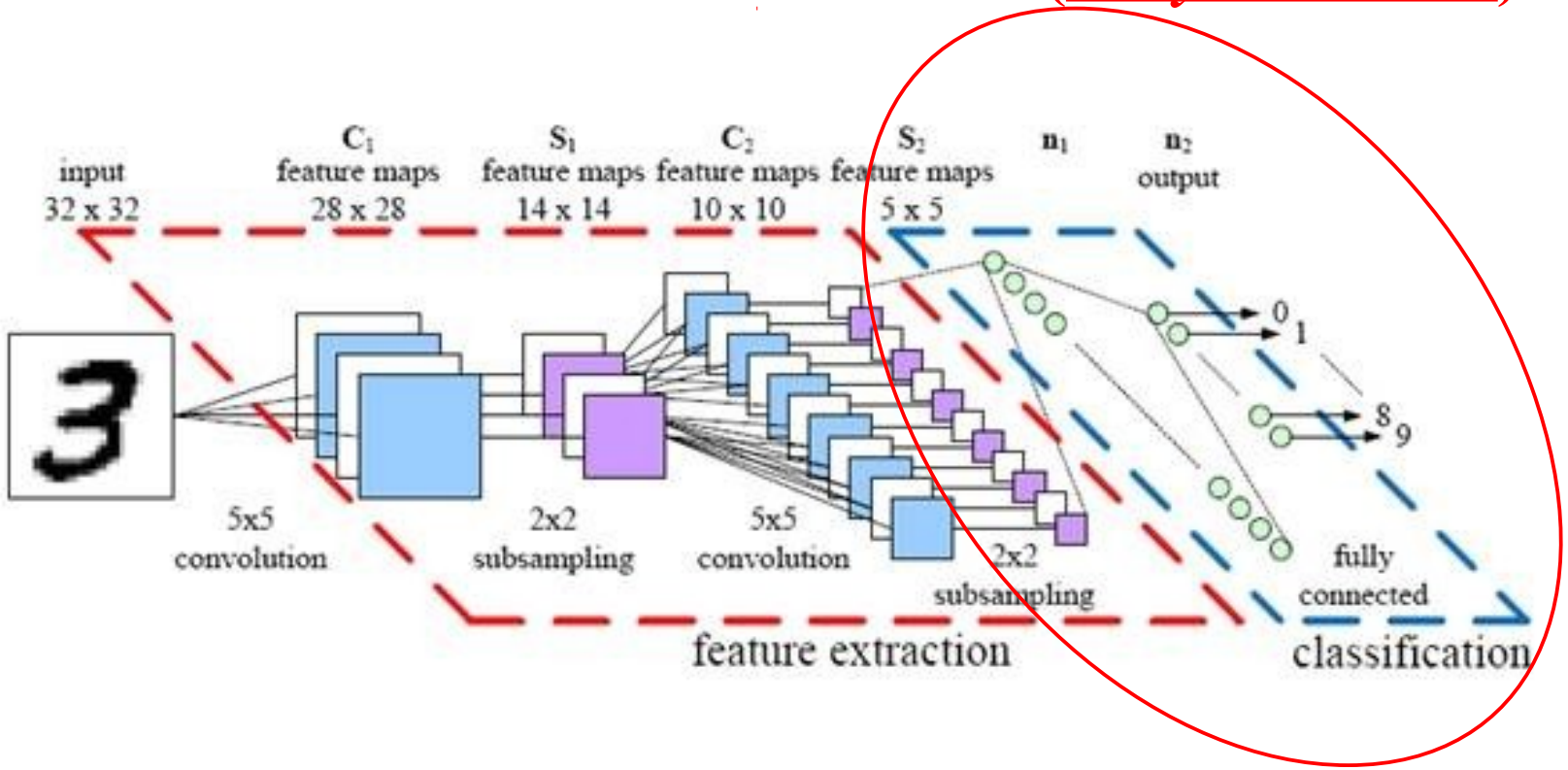
Lembram dos picos da camada convolucional?

CNN


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

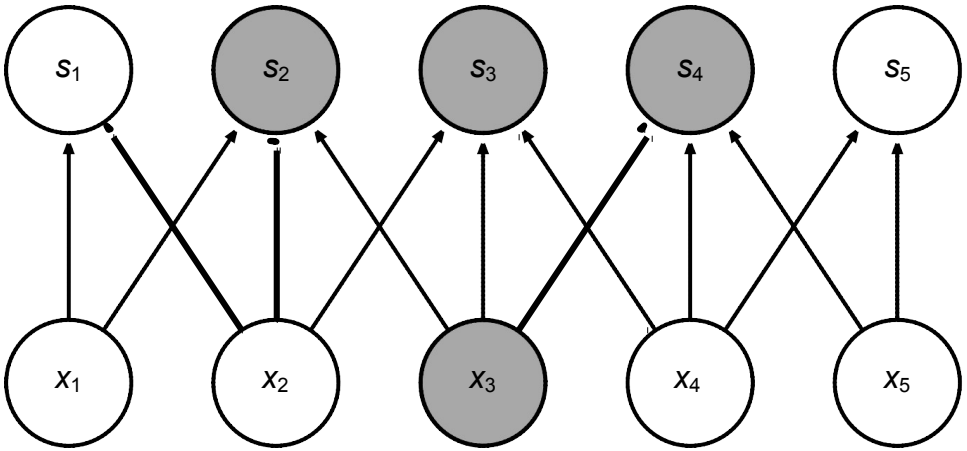
Rede CNN (LENET-5)

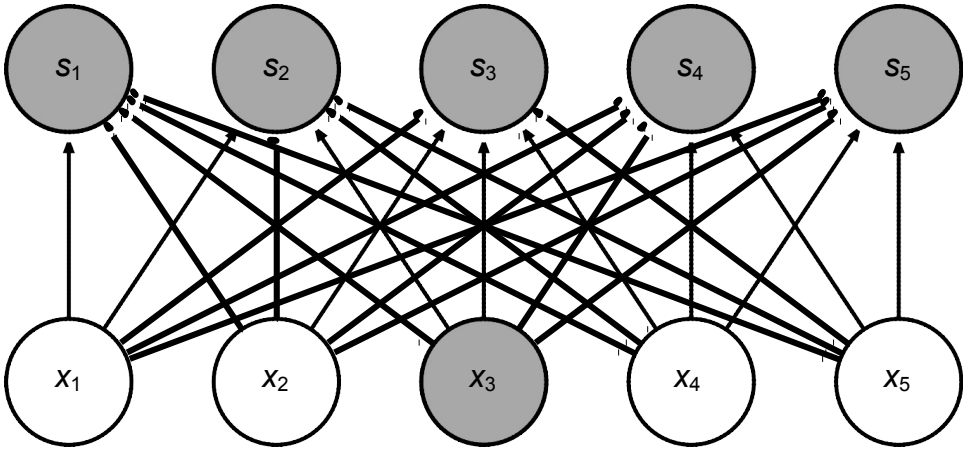
MLP densa (*Fully-Connected*)



PEEMEN, Maurice; MESMAN, Bart; CORPORAAL, Henk. **Speed sign detection and recognition by convolutional neural networks**. In: Proceedings of the 8th International Automotive Congress. 2011. p. 162-170.

Rede esparsa ou densa





Treinamento

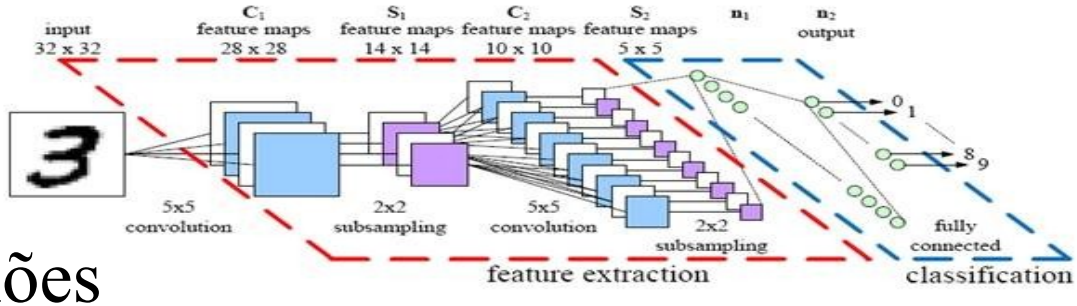
- Inicialização dos filtros e camada FC
- Fase de Aprendizado
  (Pode demorar horas ou dias dependendo da configuração da rede - uso extenso de GPUs ou nuvens)
- Validação e Backpropagation
- Armazenamento dos pesos e filtros ao longo do processo

CNNs

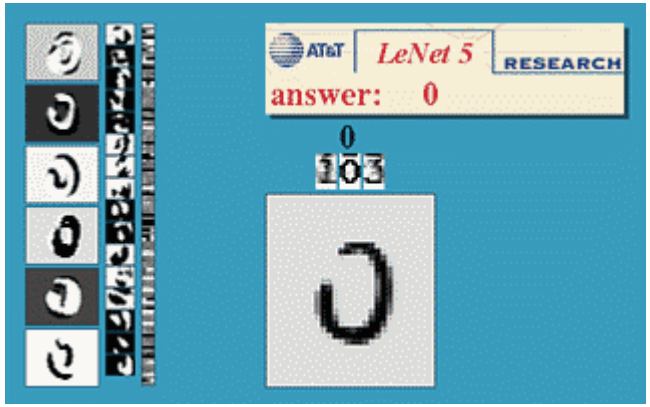
- Redes
  - LENET 5
  - AlexNet
  - GoogLeNet
- Ferramentas
  - Google Cloud Vision
  - IBM Watson Visual Recognition
  - Clarifai

LENET 5

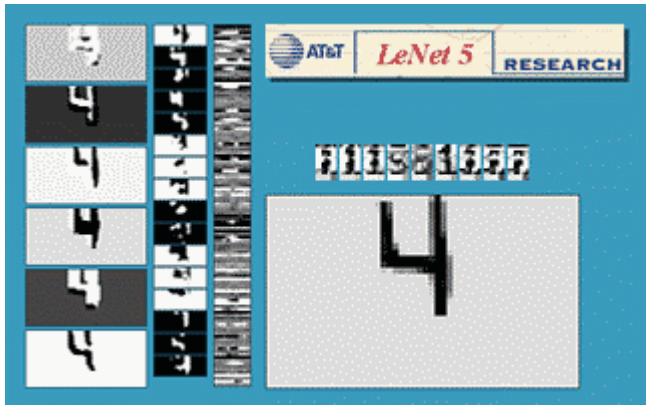
- Primeira CNN implementada e testada com sucesso (Bell Labs) / Yan Lecun – 1998
- Reconhecimento de Dígitos Manuscritos
  - MNIST DATASET (10 Classes [0-9])
  - 60 K Treinamento
  - 10 K Teste
  - 0.95% (erro)
  - ~345 K de conexões
  - ~60 K parâmetros

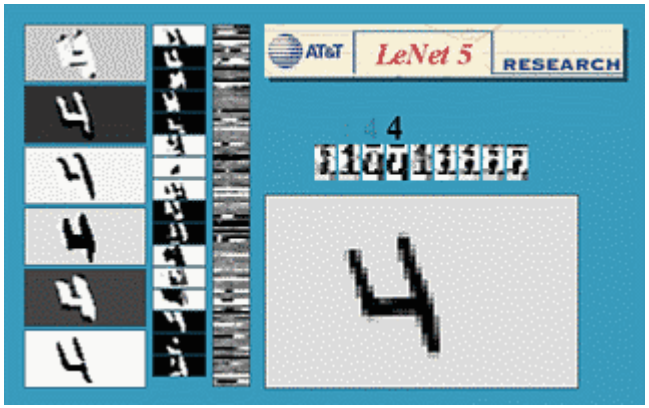


LENET 5

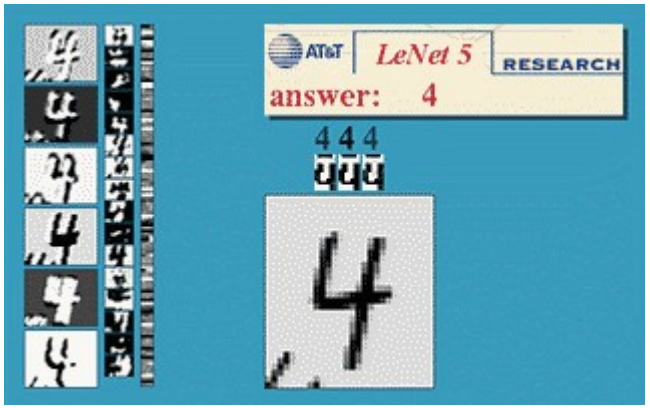

AT&T
LeNet 5
RESEARCH
answer: 0
0

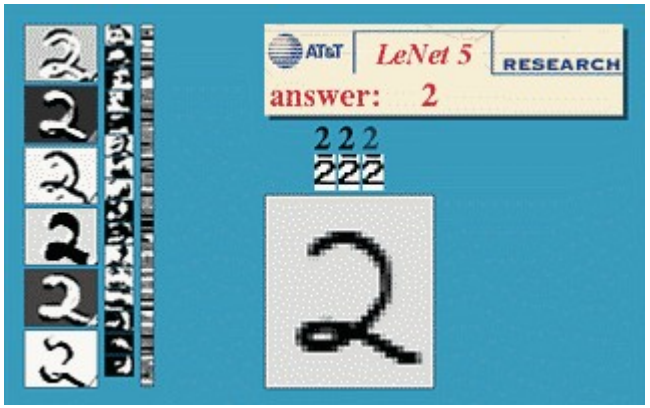

AT&T
LeNet 5
RESEARCH
4

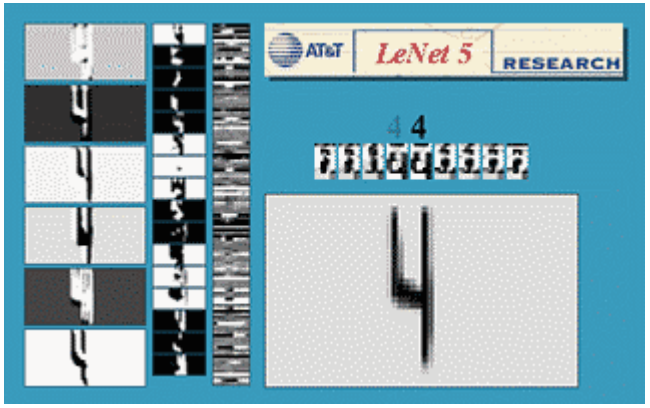

AT&T
LeNet 5
RESEARCH

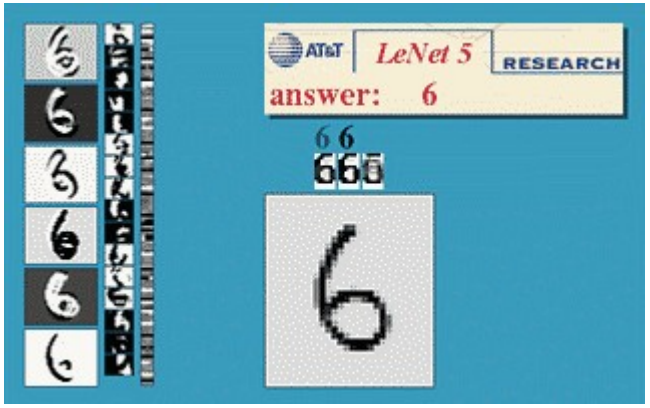

AT&T
LeNet 5
RESEARCH
4

LENET 5


AT&T
LeNet 5
RESEARCH
answer: 4

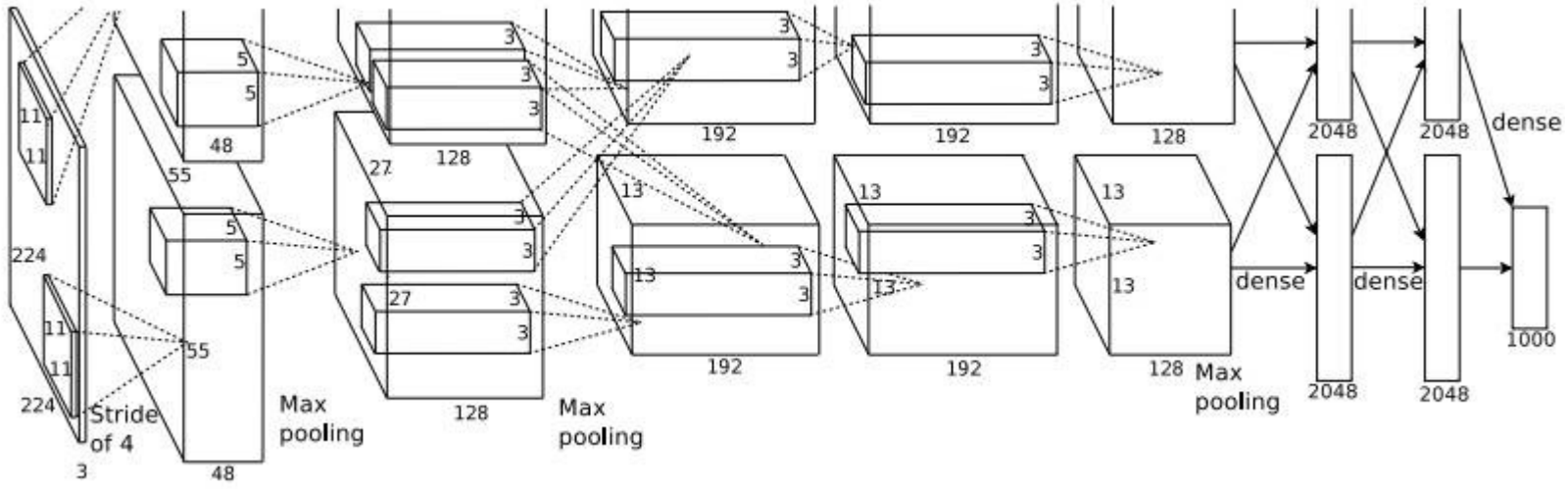

AT&T
LeNet 5
RESEARCH
answer: 2


AT&T
LeNet 5
RESEARCH


AT&T
LeNet 5
RESEARCH
answer: 6

AlexNet

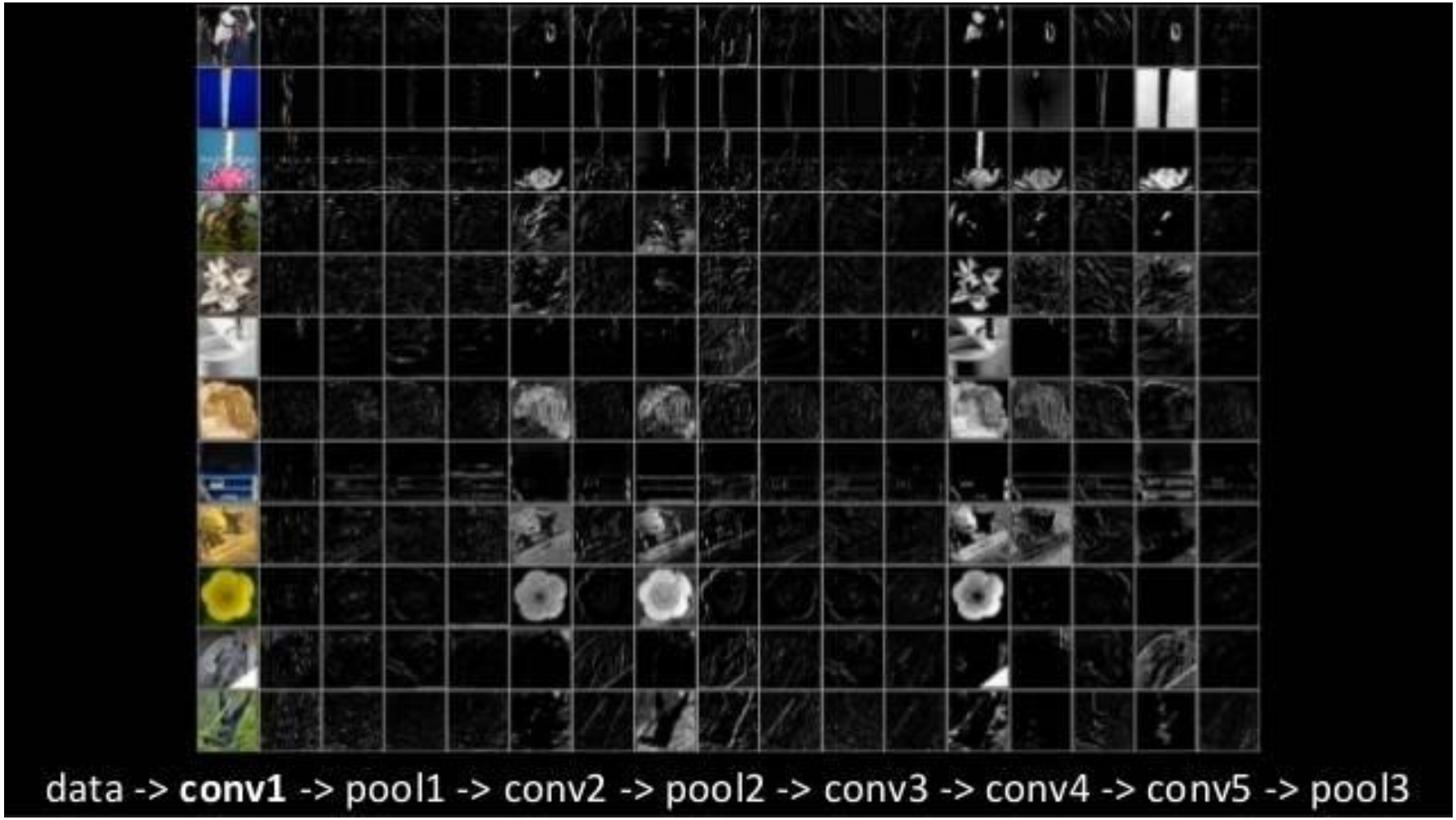
- Alex Krizhevsky – 2012 (Krizhevsky Net)
- Imagenet 2012 Challenge (1000 classes)
- Vencedor - erro 15.3% (2° SIFT Based - 26.2%)
- 1.2 M Treinamento
- 50 K Validação
- 150 K Teste



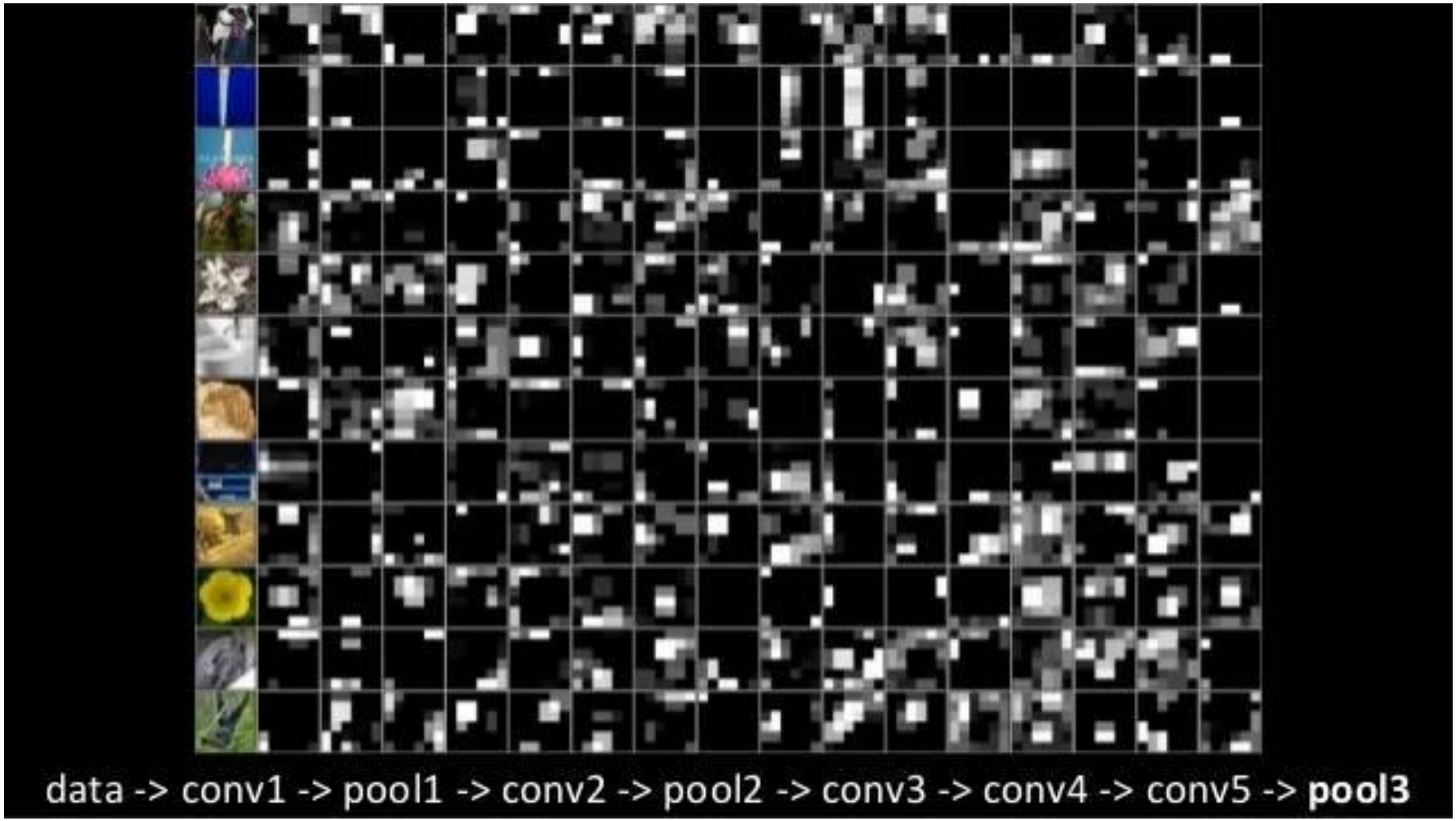
AlexNet

- Neurônios por Camada: ~253k, ~186k, ~64k , ~64k, ~43k, ~4k, ~4k, ~1k
- 60 M de Parâmetros
- Treinamento: 6 dias, 2 NVidia GTX 580 3GB

AlexNet



AlexNet

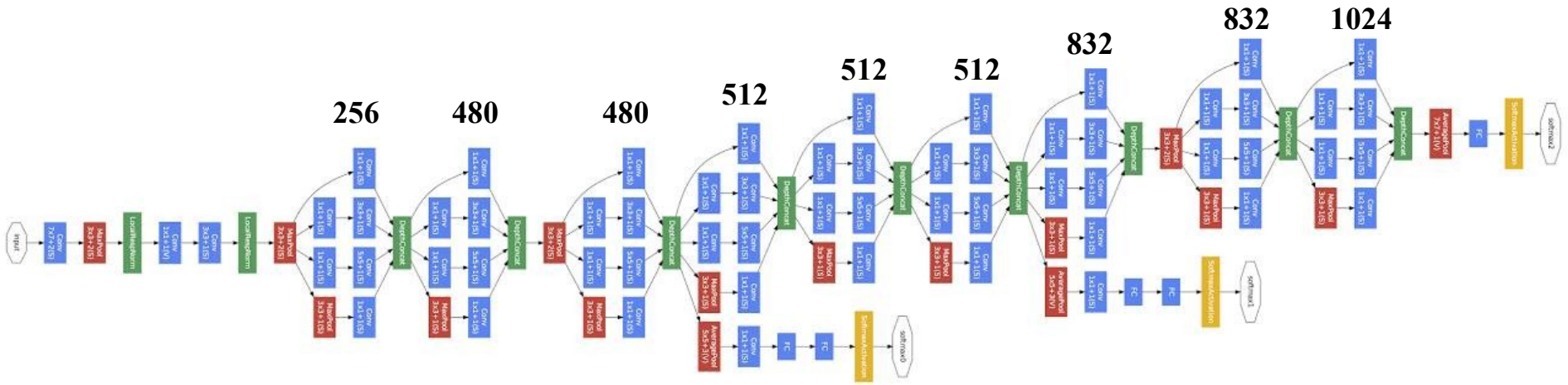


Departamento de Ciência da COMPUTAÇÃO

UDESC Joinville

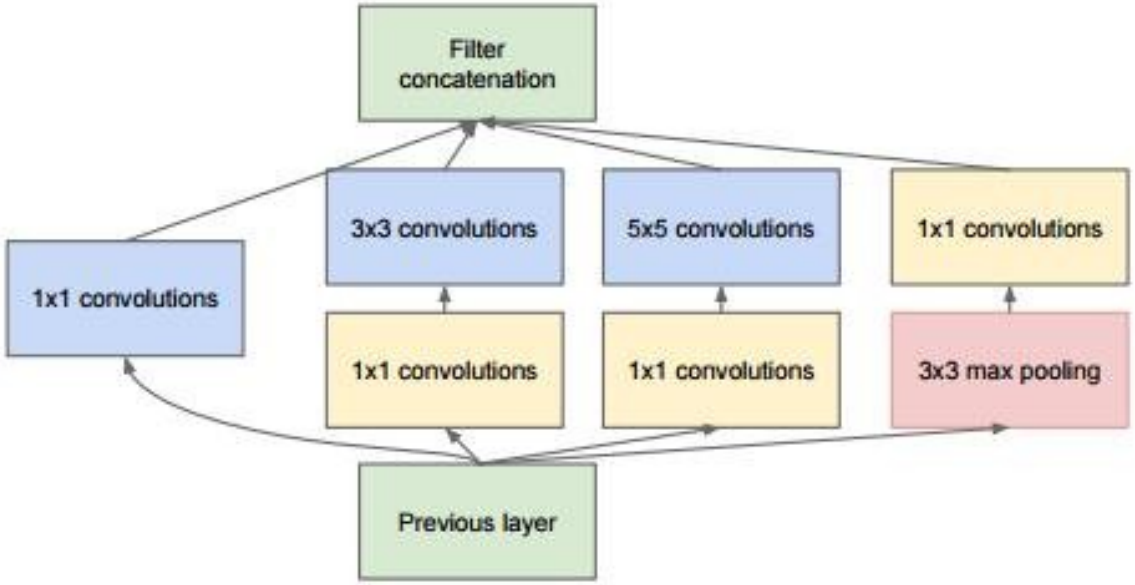
GoogLeNet

- Custo computacional “apenas” 2x mais que AlexNet
- Possui 5 camadas de convolução seguidas por pooling
- Conceito de inceptions (redes dentro de redes)
- Imagenet 2014 Challenge (1000 classes)
  - 1.2 M Treinamento, 50 K Validação, 100 K Teste
- Vencedor - erro 6.67% (2° e 3° também CNN)



GoogLeNet - Inceptions

- Redes dentro de Redes
- 256 a 1024 filtros



GoogLeNet

- Em 2012, o Google utilizou uma parte da Imagenet, para treinar sua rede. O subconjunto era composto por imagens de cães, aprendendo a extrair características com base somente em... cães!

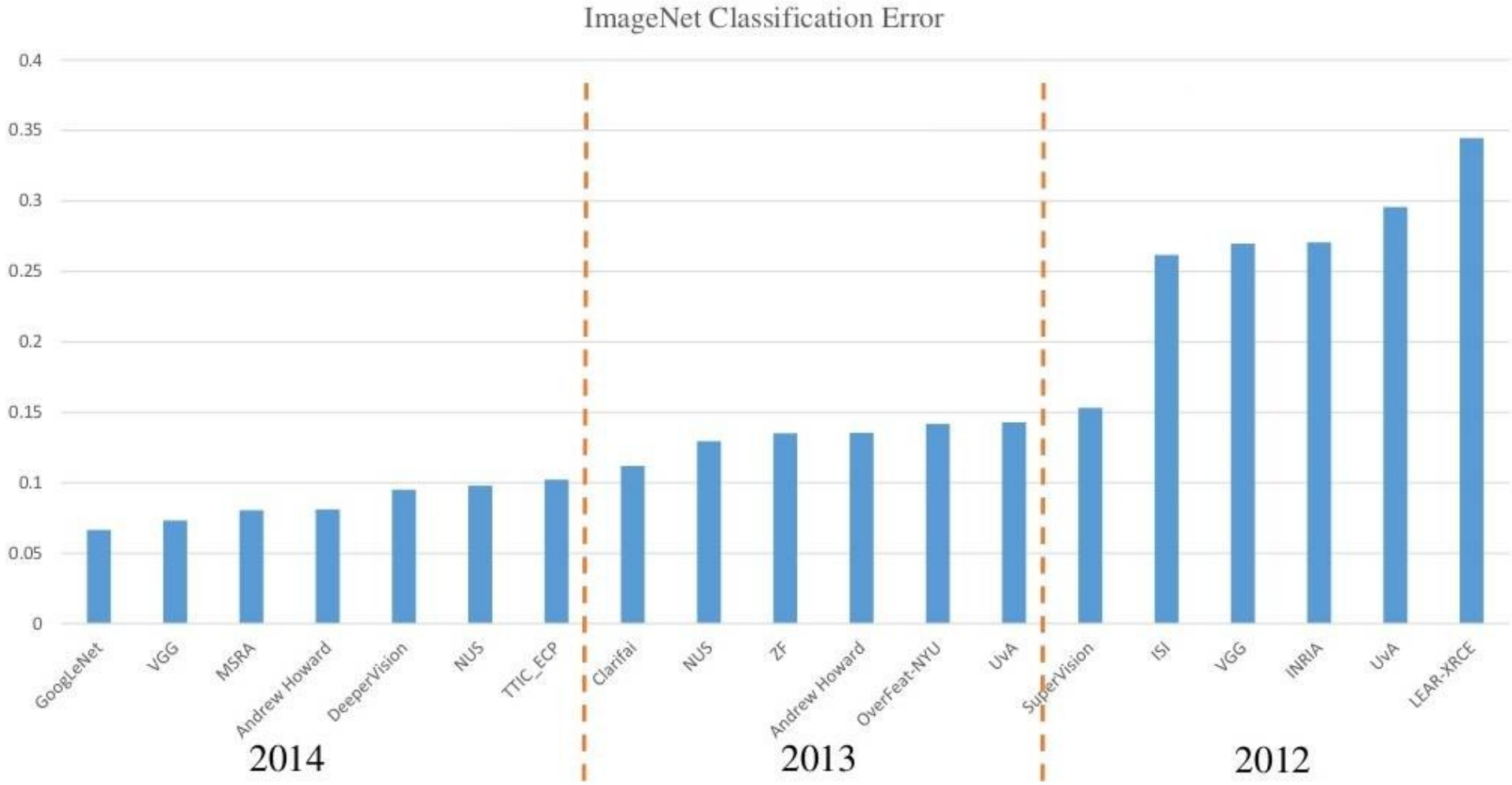



Departamento de Ciência da COMPUTAÇÃO

Resultados ImageNet

UDESC Joinville

- **1000** categories and **1.2** million training images



Mais recentes: WMW **0.02251** (2017), Trimps-Soushen **0.02991** (2016) e MSRA **0.03567** (2015)

Utilizando CNNs

- Construir uma CNN a partir do zero pode ser uma tarefa cara e demorada. Para isso existe série de APIs que visam permitir que as organizações obtenham insights sem necessidade de implementar uma CNN ou necessitar de experiência em Visão Computacional.
- Google Cloud Vision
  - É um serviço de reconhecimento visual do Google que utiliza uma API REST. Esta API é baseada na estrutura TensorFlow de código aberto. Ele detecta rostos e objetos individuais e contém um conjunto de rótulos bastante abrangente.
- IBM Watson Visual Recognition
  - É parte do Watson Developer Cloud e vem com um enorme conjunto de classes embutidas. Foi construído para treinar classes personalizadas com base nas imagens que você fornece. Ele também suporta uma série de características como a detecção de NSFW e OCR, como o Google Cloud Vision.

Utilizando CNNs

- Clarifai
  - É um serviço de reconhecimento de imagens que também utiliza uma API REST. Possui uma série de módulos que ajudam a adaptar seu algoritmo a assuntos específicos, como alimentos, viagens e casamentos.
- Embora esses serviços sejam adequadas para aplicações comuns, melhores resultados podem ser obtidos desenvolvendo uma solução personalizada para tarefas específicas. Existem diversas bibliotecas otimizadas que permite que desenvolvedores se concentrem nos modelos de treinamento: Caffe, Torch, DeepLearning4J, Keras, MxNet, TensorFlow, entre outras.

Desvantagens

- Em relação à quantidade de memória, uma CNN não é muito maior que uma MLP normal. Mas as camadas convolucionais demandam um tempo computacional elevado, usando mais de 67% do tempo.
- As CNNs são aproximadamente 3X mais lentas que uma rede totalmente conectada do mesmo tamanho (mesma quantidade de pesos a ajustar).
- Requerem muitos dados para treinamento (podendo necessitar *Data Augmentation*).