

# Grupo de Estudos de Visão Computacional

Guilherme Reis  
Hugo Pessoni

Apoio



## Introdução a Redes Neurais Convolucionais (CNNs)

Goiânia  
2024



# Sumário

1. Introdução e Objetivos
2. Contextualização Histórica e Motivação
3. Arquitetura Geral
4. Seção prática.

# O que vocês precisam saber até aqui.

1. Devem “saber”:

- i. filtros
- ii. Redes Neurais (MLPs)
- iii. Backpropagation
- iv. Gradiente
- v. Regularização
- vi. Função de Ativação - ReLU



0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	29
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4
0	22	206	252	246	251	241	100	24	113	255	245	255	194	9	0
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0
0	218	251	250	137	7	11	0	0	0	2	62	255	250	129	3
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4
0	18	146	250	255	247	255	255	249	255	240	255	129	0	5	0
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1
0	0	5	5	0	0	0	0	0	0	14	1	0	6	6	0

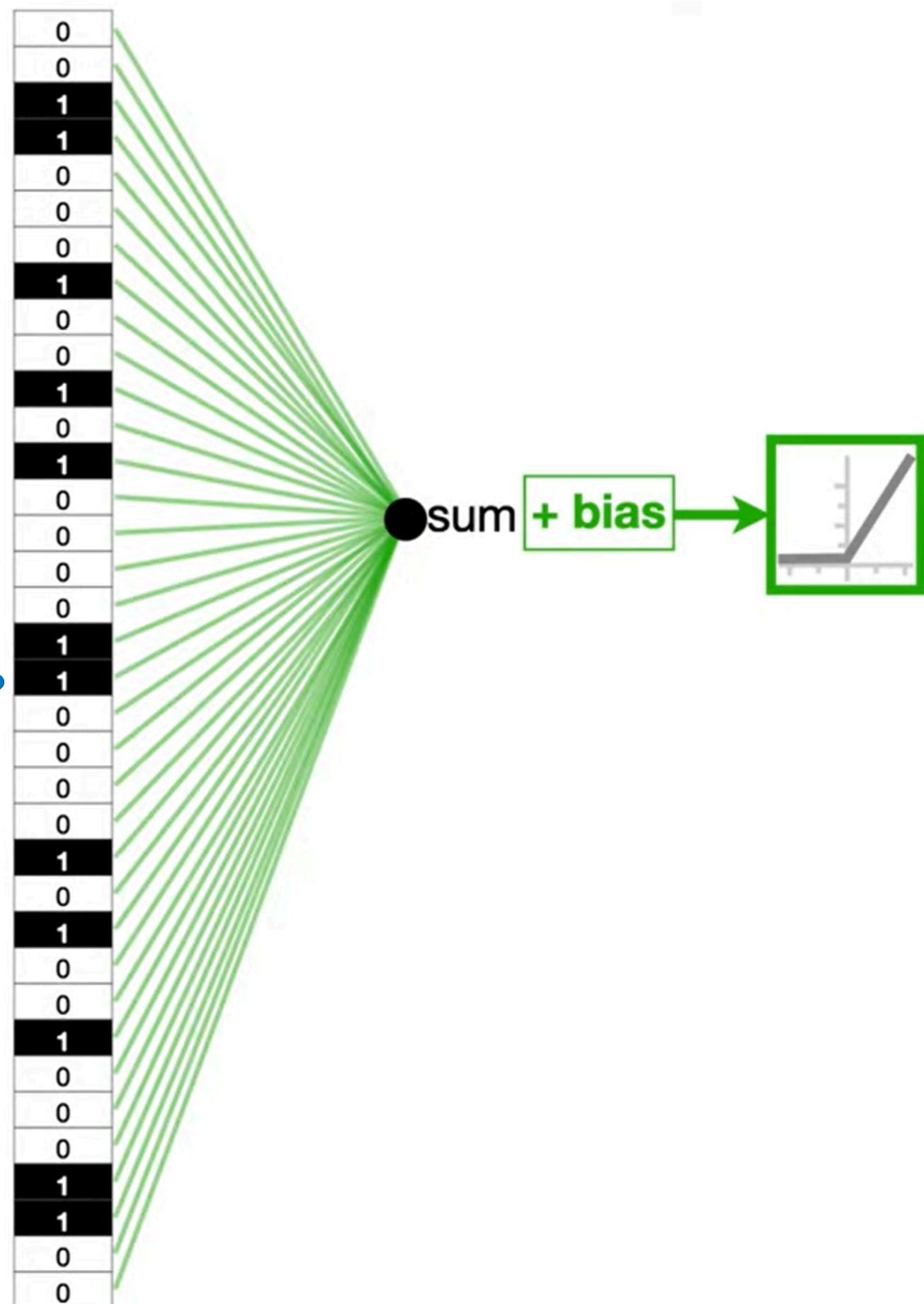
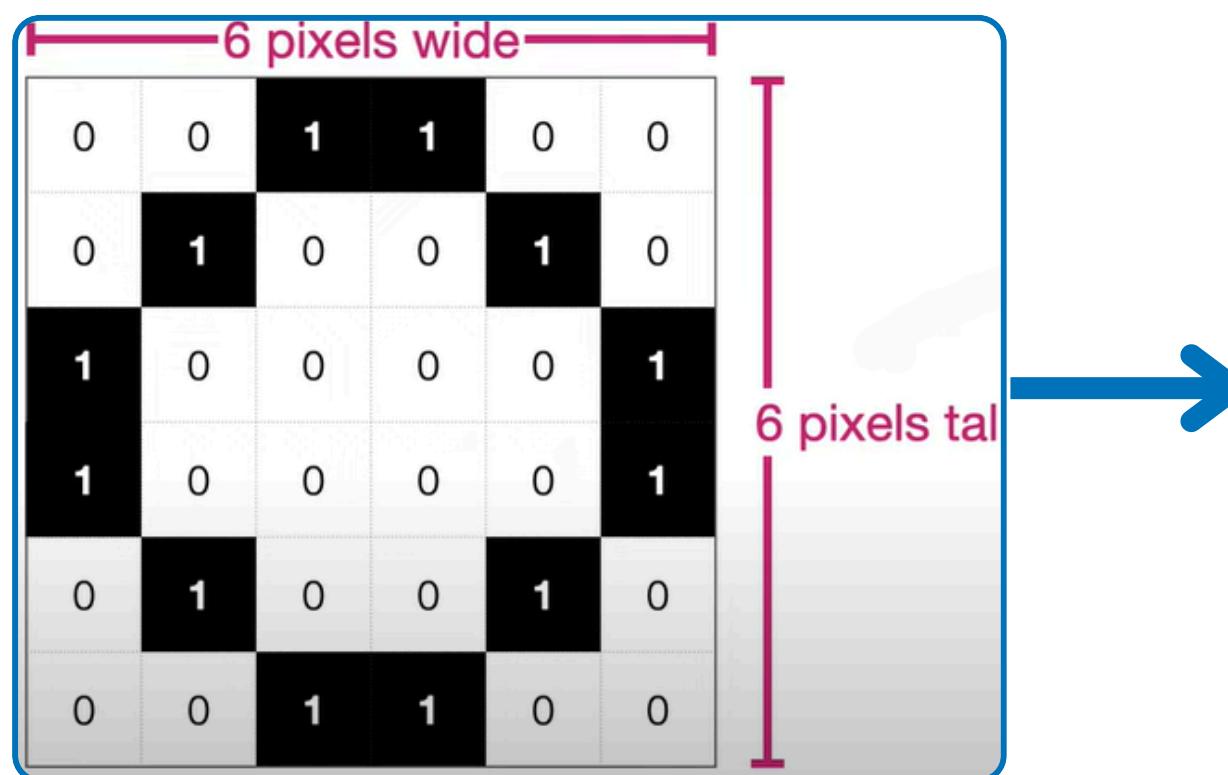
0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	29
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4
0	22	206	252	246	251	241	100	24	113	255	245	255	194	9	0
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0
0	218	251	250	137	7	11	0	0	0	2	62	255	250	129	3
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4
0	18	146	250	255	247	255	255	249	255	240	255	129	0	5	0
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1
0	0	5	5	0	0	0	0	0	0	14	1	0	6	6	0

# Por que precisamos de CNNs se já temos MLPs?

1. Devem “saber”:
  - i. Redes Neurais (MLPs)
  - ii. Backpropagation
  - iii. Gradiente
  - iv. Regularização
  - v. Função de Ativação - ReLU
2. Vamos brincar de classificação, quero diferenciar **O** e **X** para um jogo da velha contra o computador.
3. Como é um tela, vemos essas imagens formadas por pixels, correto?

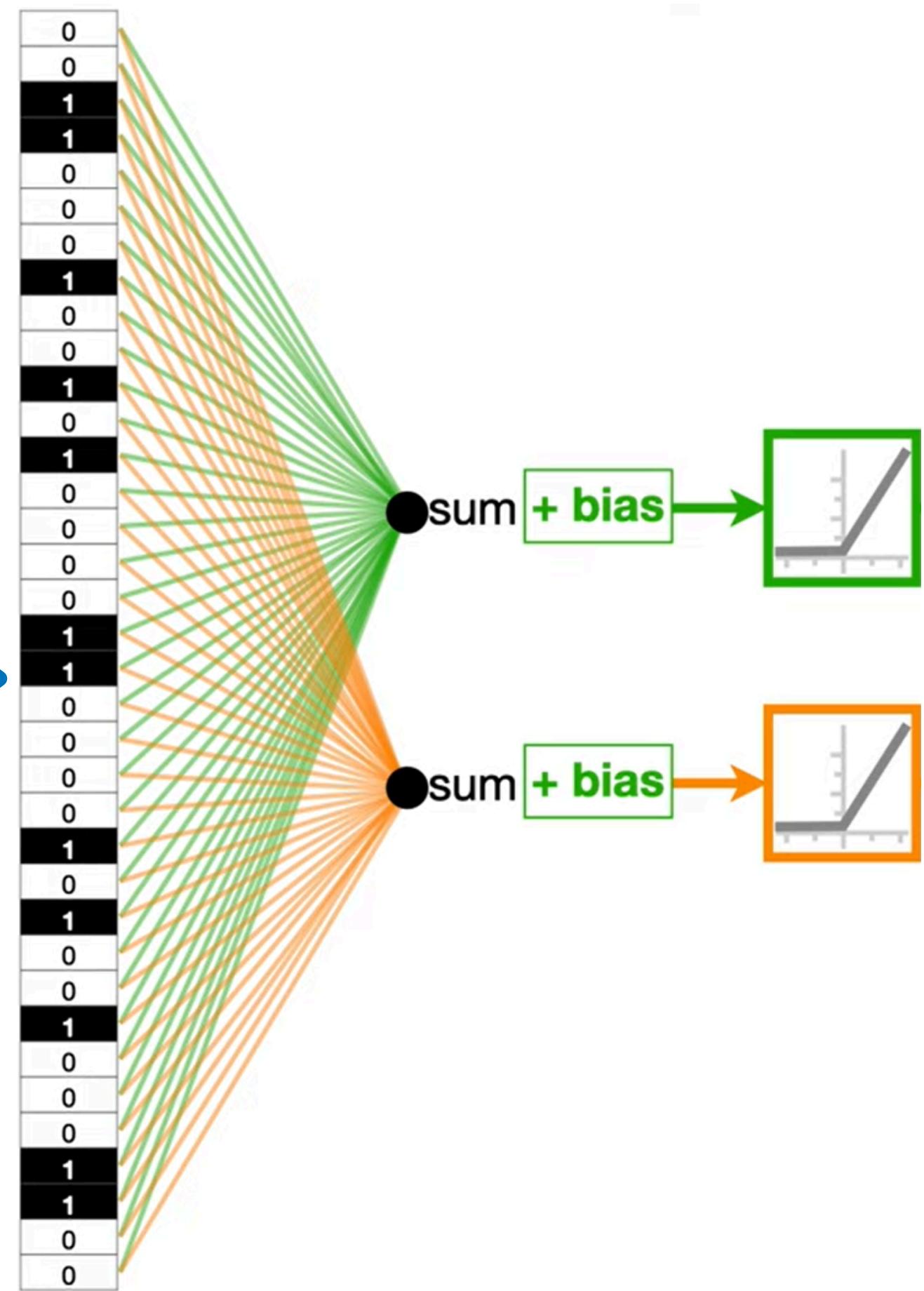
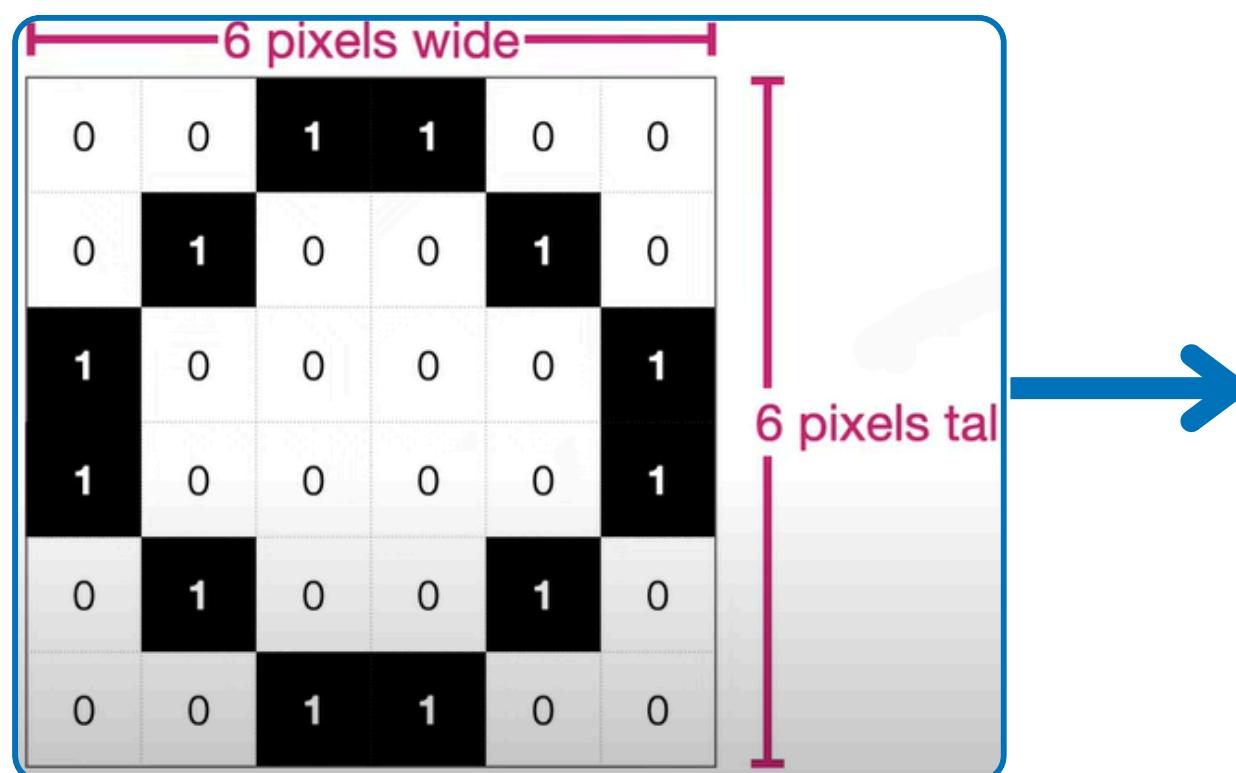
# Por que precisamos de CNNs se já temos MLPs?

1. Vamos começar pelo 0



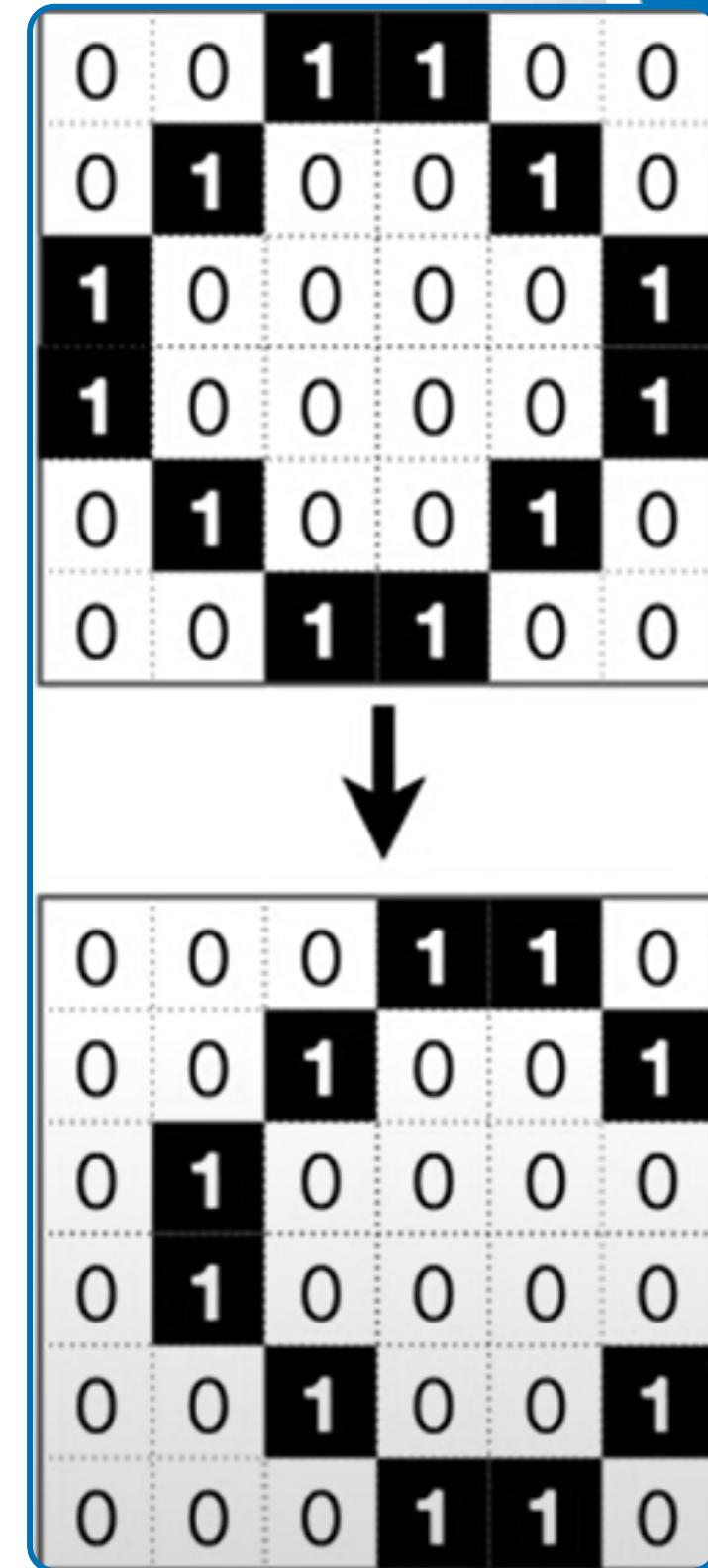
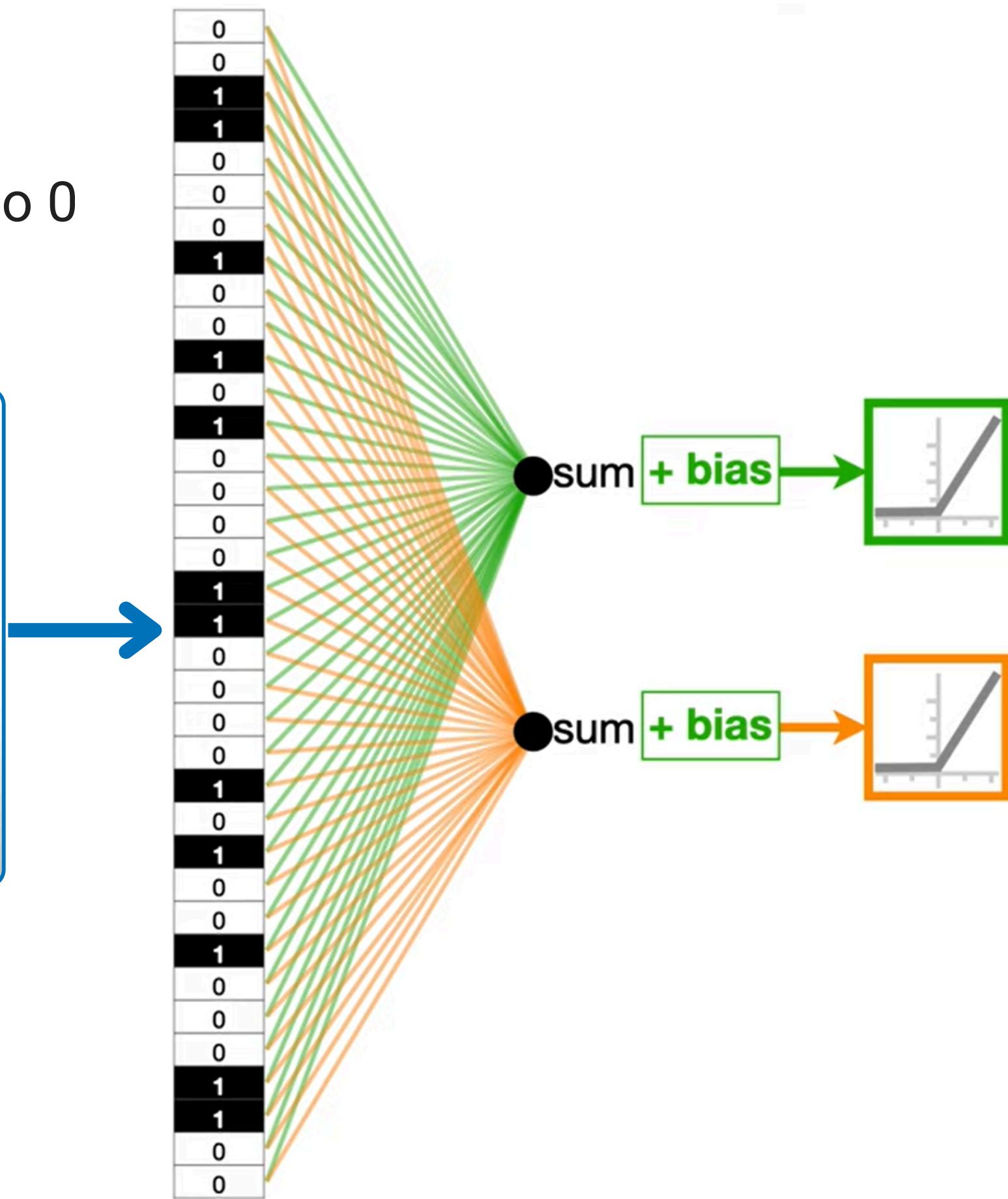
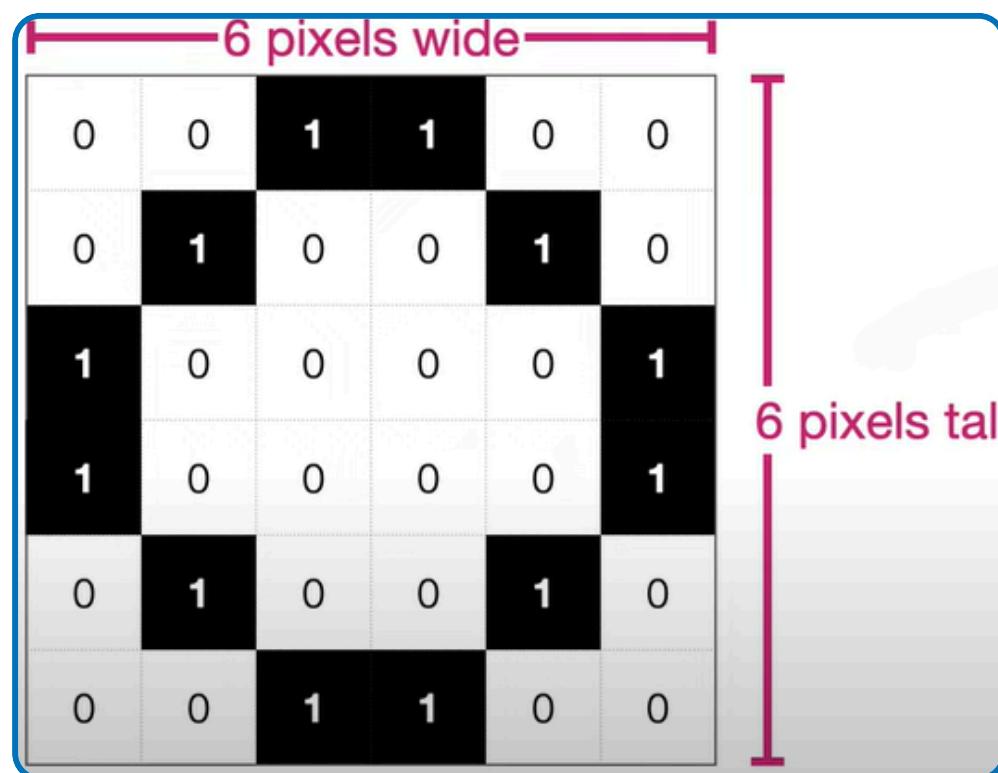
# Por que precisamos de CNNs se já temos MLPs?

1. Vamos começar pelo 0



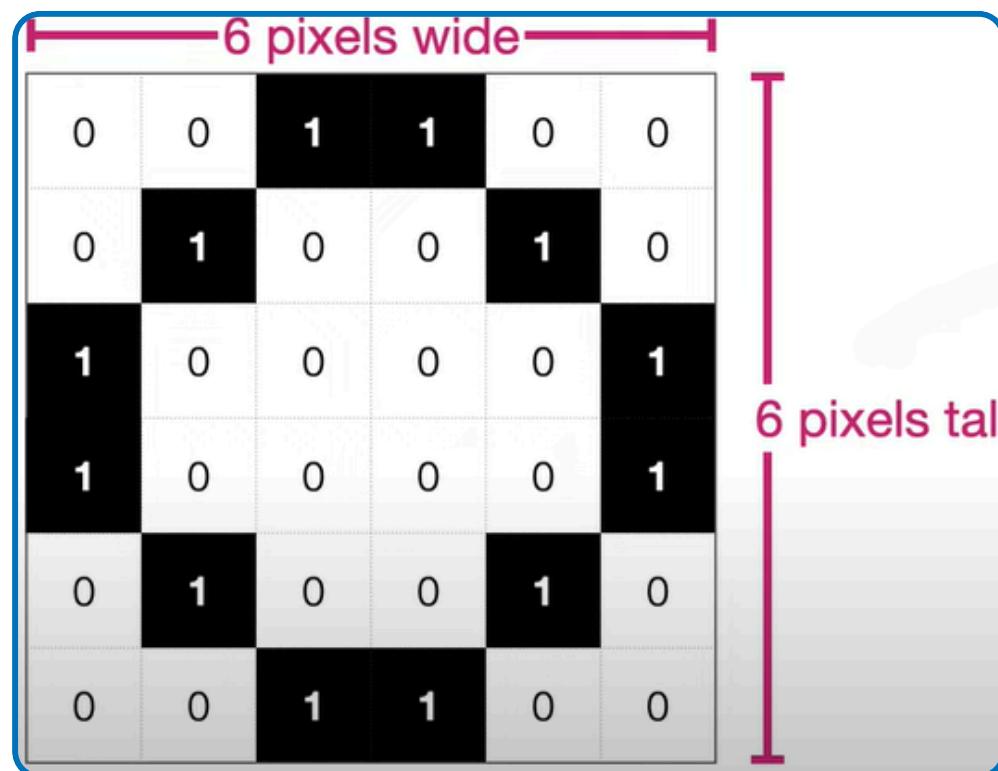
# Por que precisamos de CNNs se já temos MLPs?

1. Vamos começar pelo 0

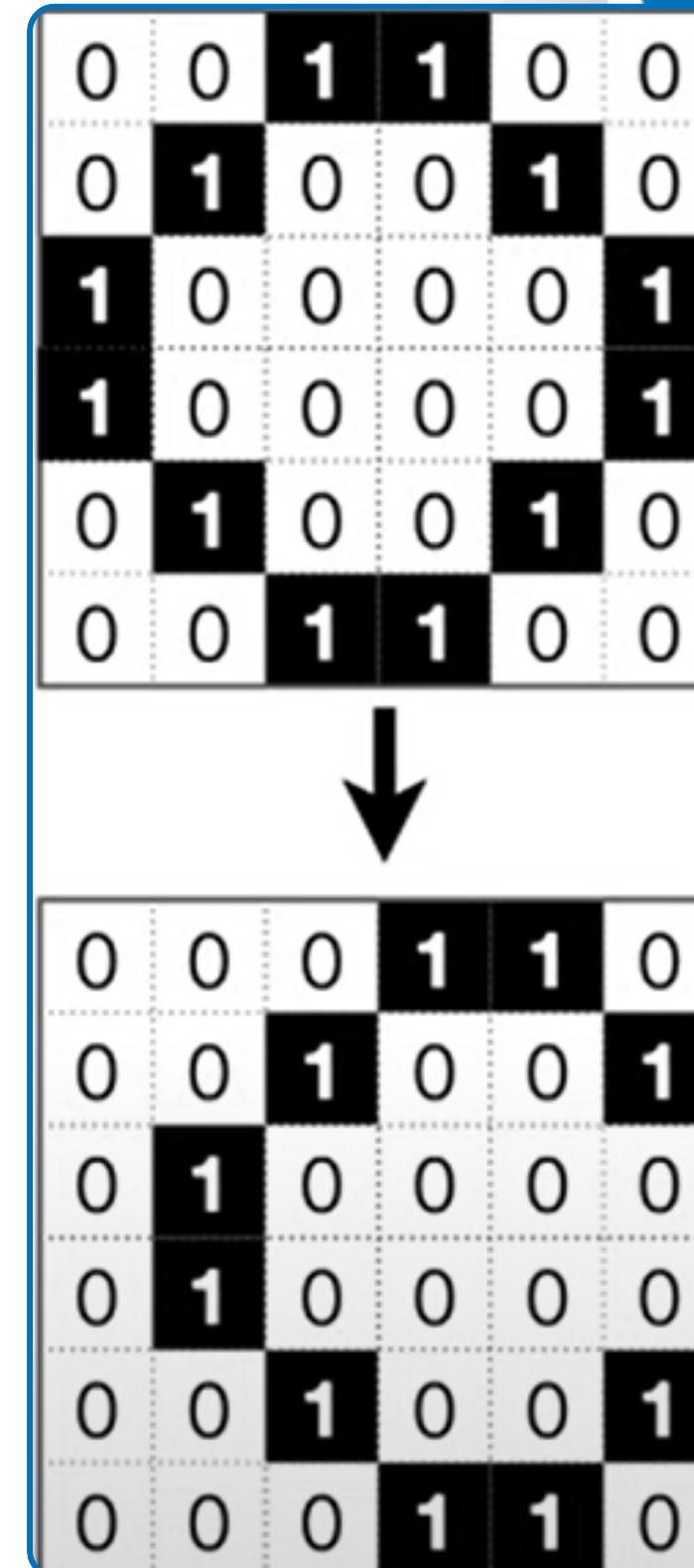
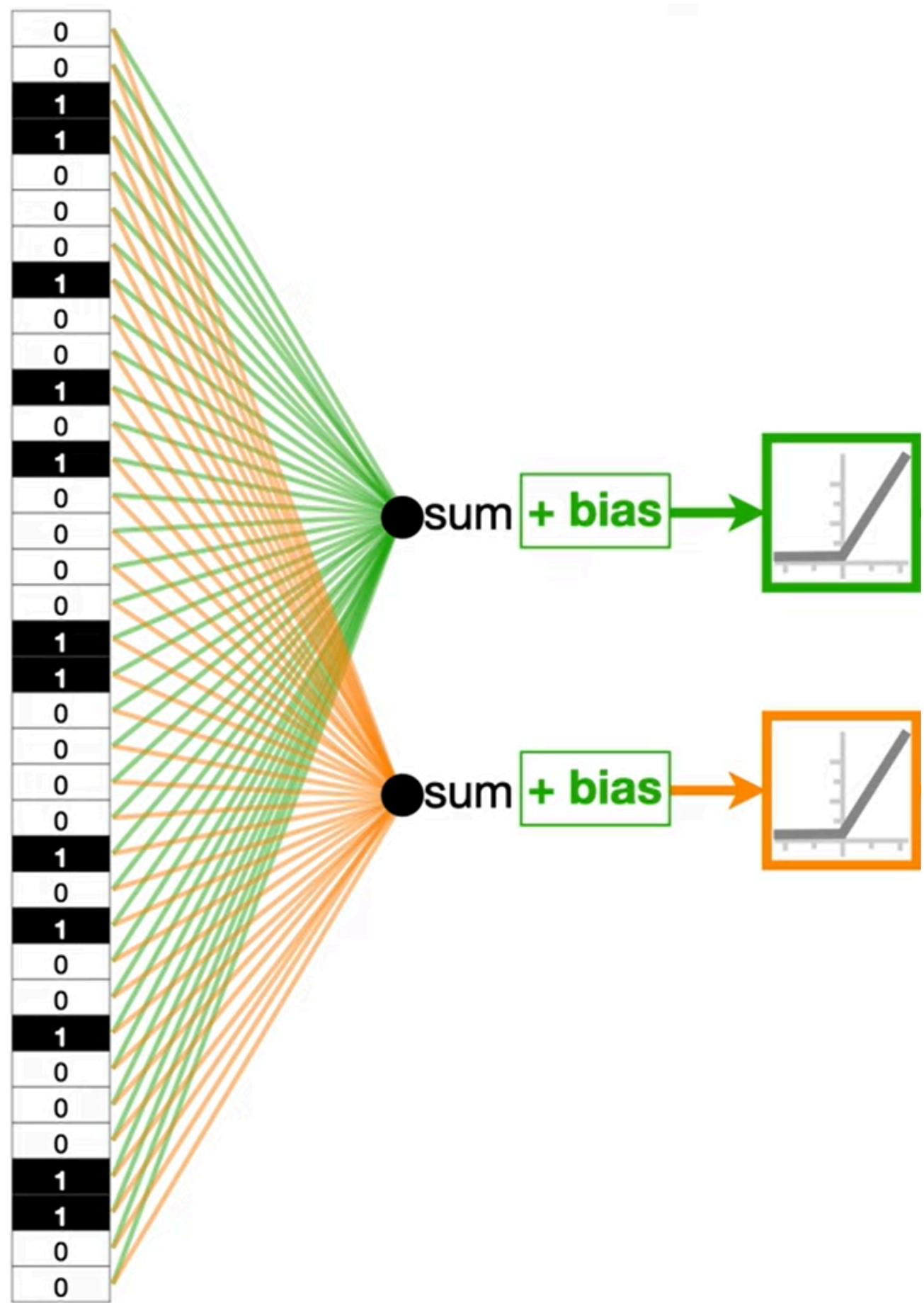


# Por que precisamos de CNNs se já temos MLPs?

1. Vamos começar pelo 0



- $1080 \times 720 \times 3 = 2.332.800$



# Por que precisamos de CNNs se já temos MLPs?

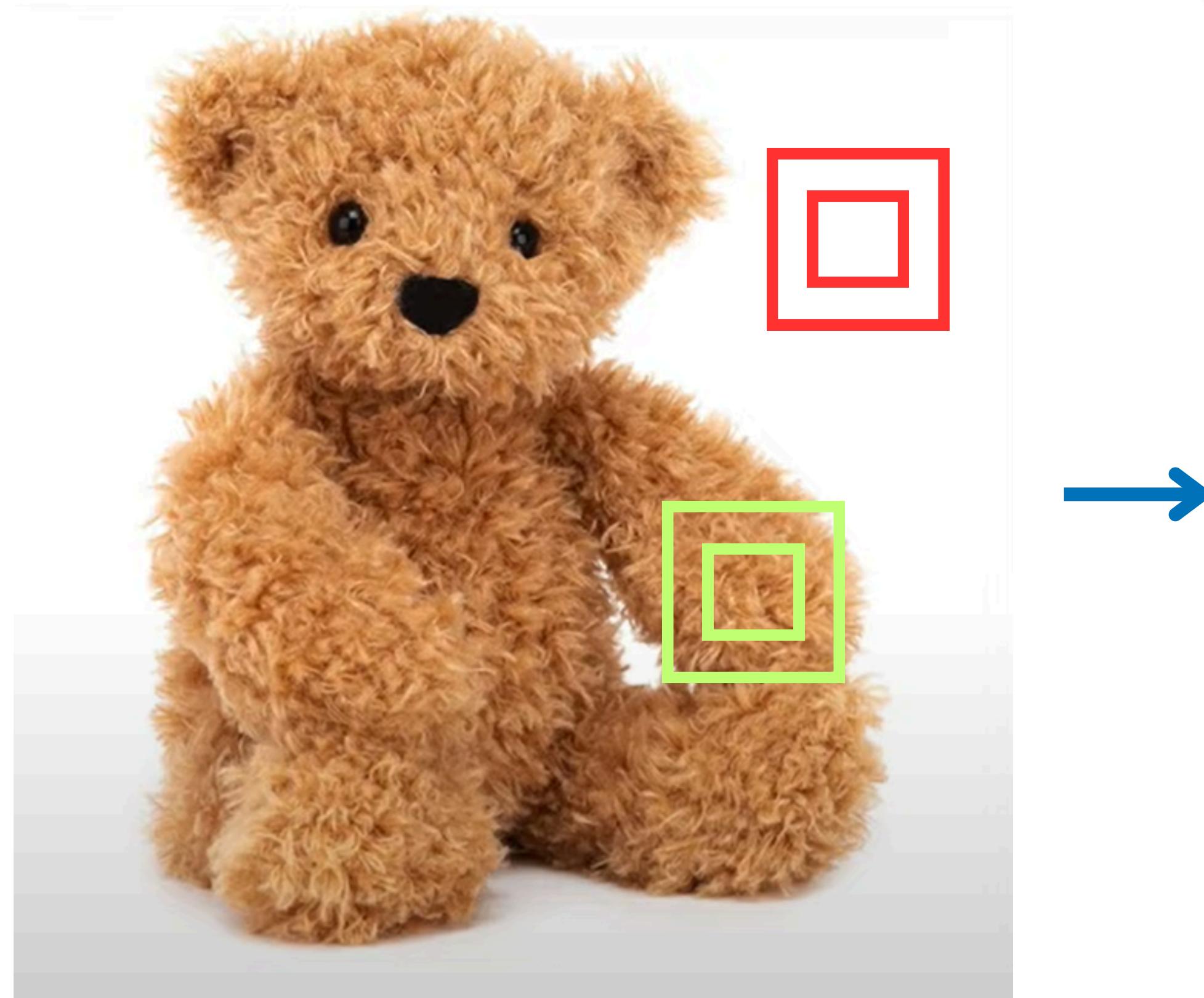


# Por que precisamos de CNNs se já temos MLPs?



# Por que precisamos de CNNs se já temos MLPs?

- Perda de correlação  
de pixels

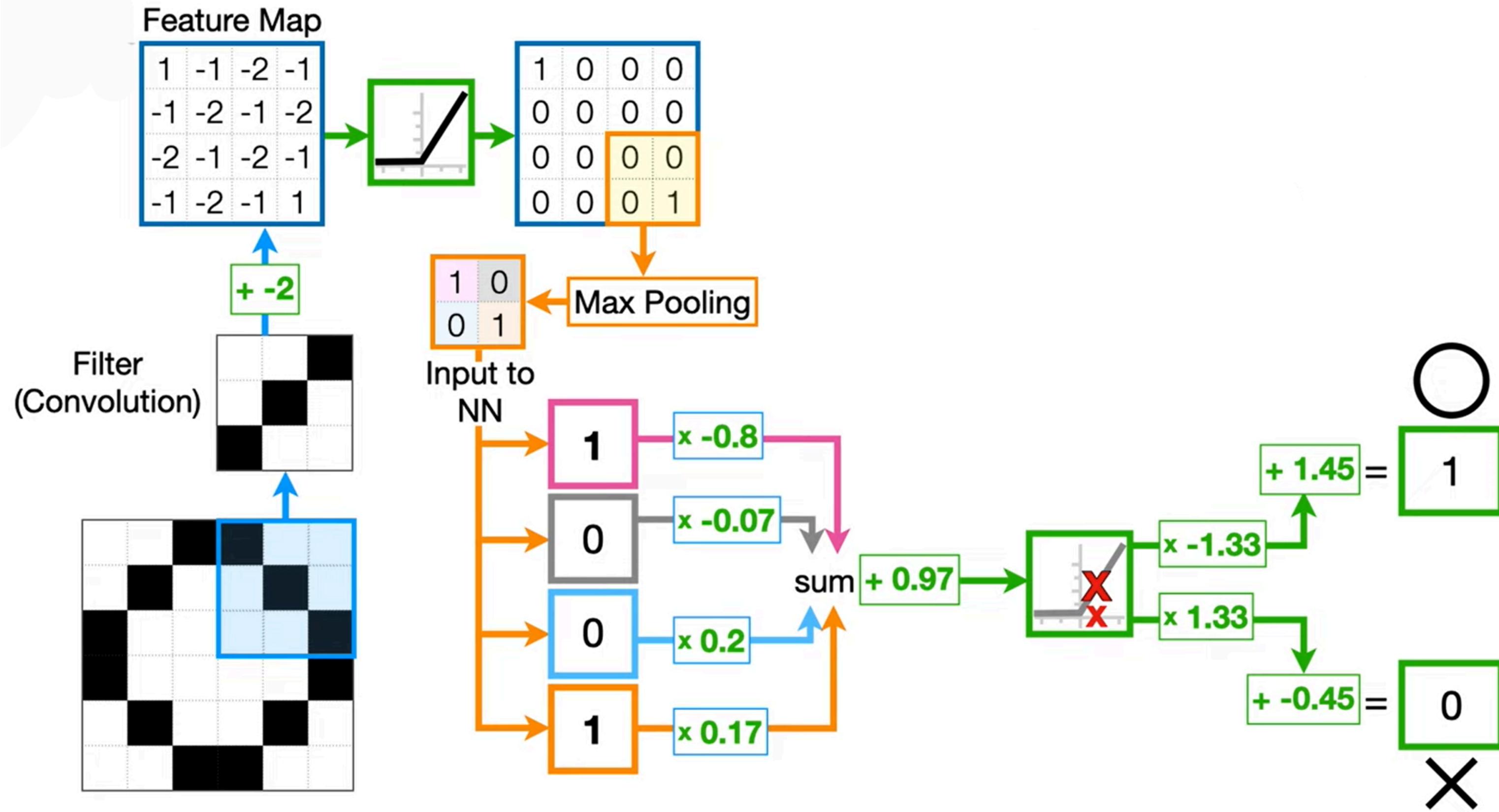


0
0
1
1
0
0
0
1
0
0
1
1
0
0
1
1
0
0
0
0
1
1
0
0
0
0
1
0
1
0
0
1
1
0
0
0
1
1
0
0
0
1
1
0
0
0
1
1
0
0
0
1
1
0
0
0

# Por que precisamos de CNNs se já temos MLPs?

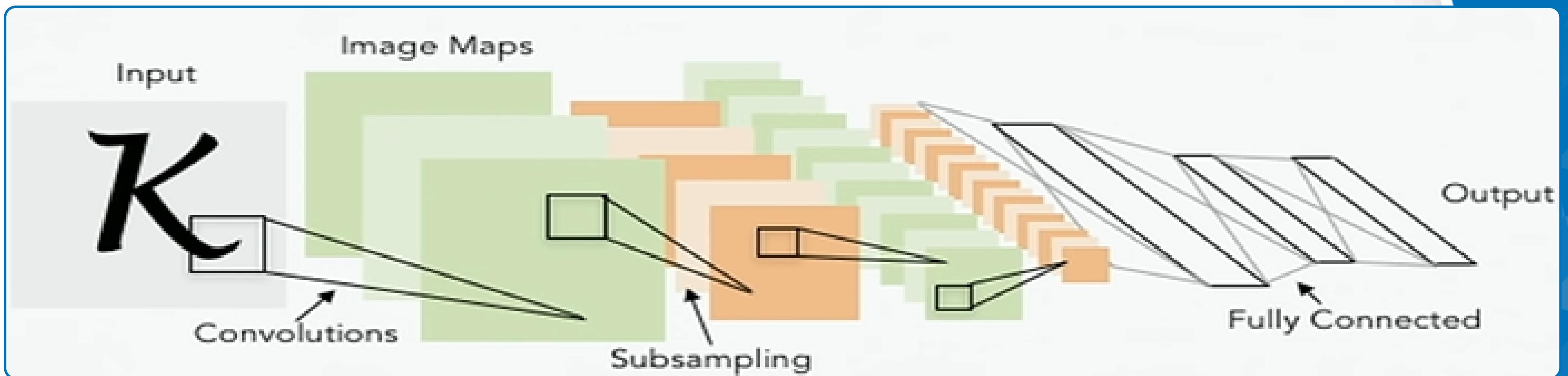
1. Reduzir a quantidade de neurônios;
2. Precisa que tenha tolerância quanto a movimentação de pixels na imagem;
3. Precisa que colete correlações entre pixels;

## CNN



# Contextualização Histórica

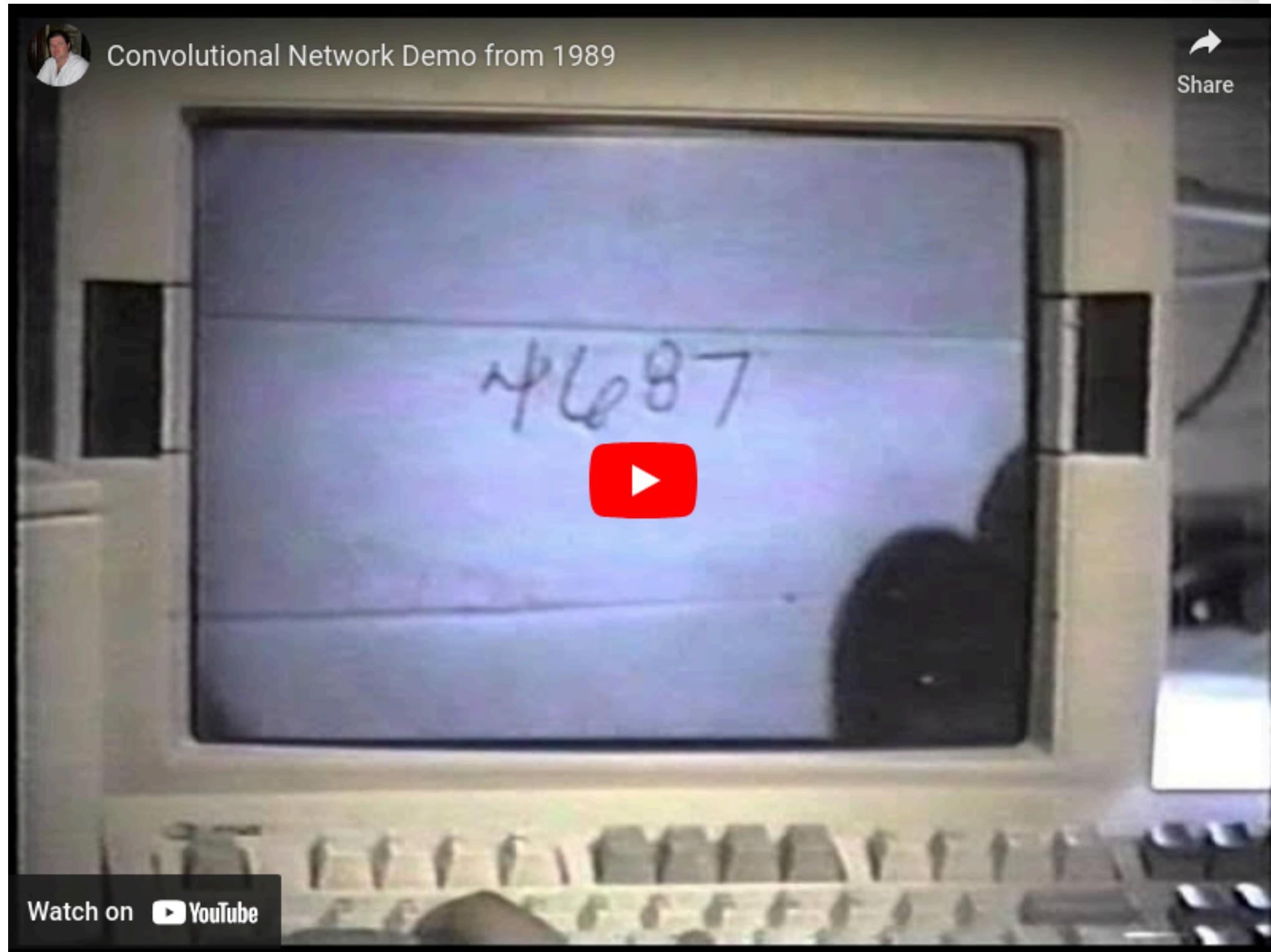
1. Um dos primeiros trabalhos foi de Yann LeCun em 1989 para reconhecimento de dígitos para cheques, porém somente em 1998 é que podemos considerar uma aplicação de sucesso.



# Contextualização Histórica

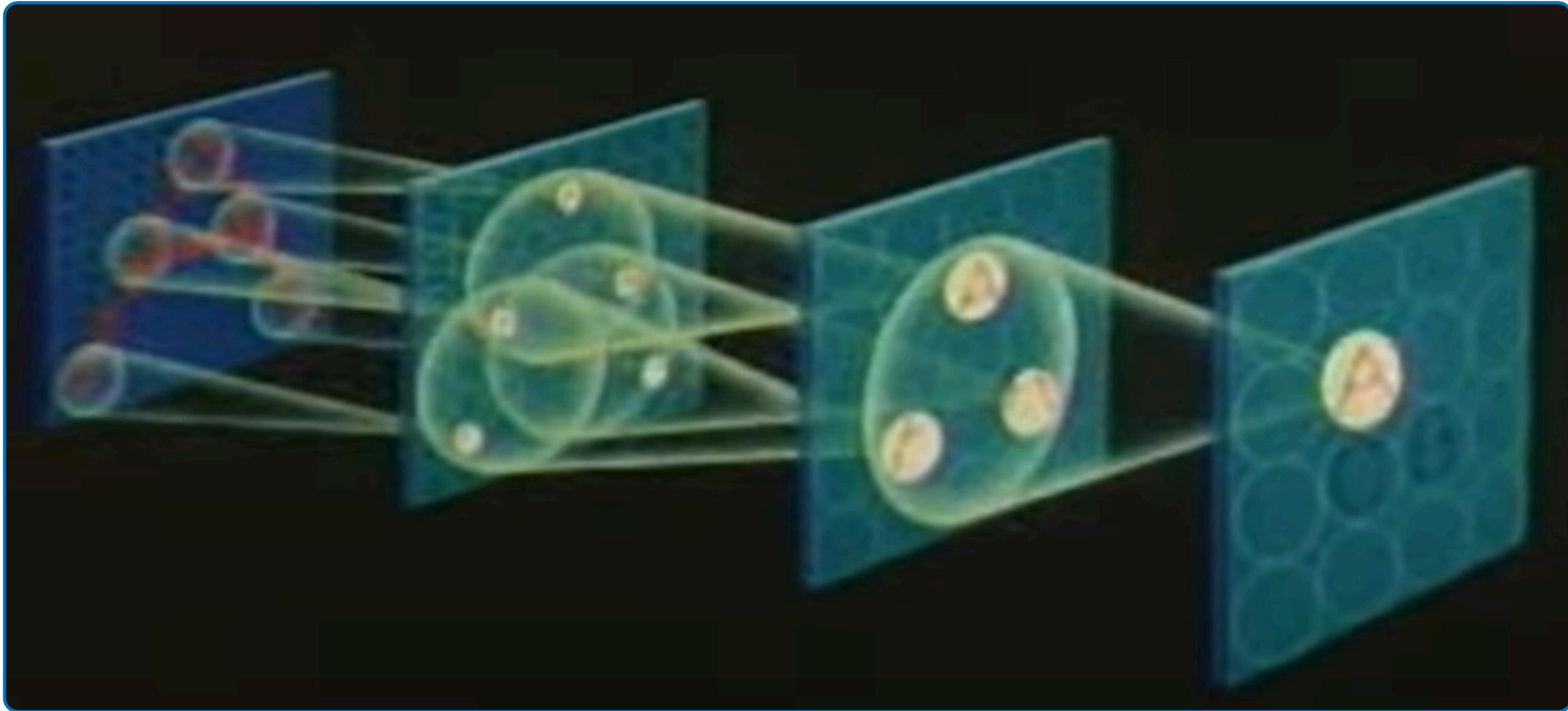
1. Um dos primeiros trabalhos foi de Yann LeCun em 1989 para reconhecimento de dígitos para cheques.

LeNet-5 em 1998



# De onde veio a ideia?

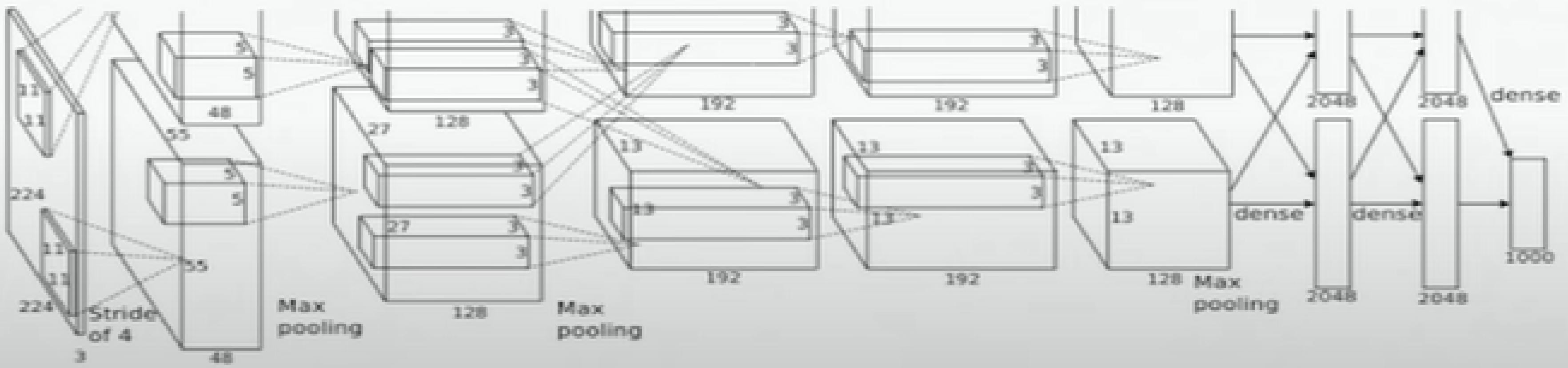
1. Neocognitron - 1986



# Contextualização Histórica

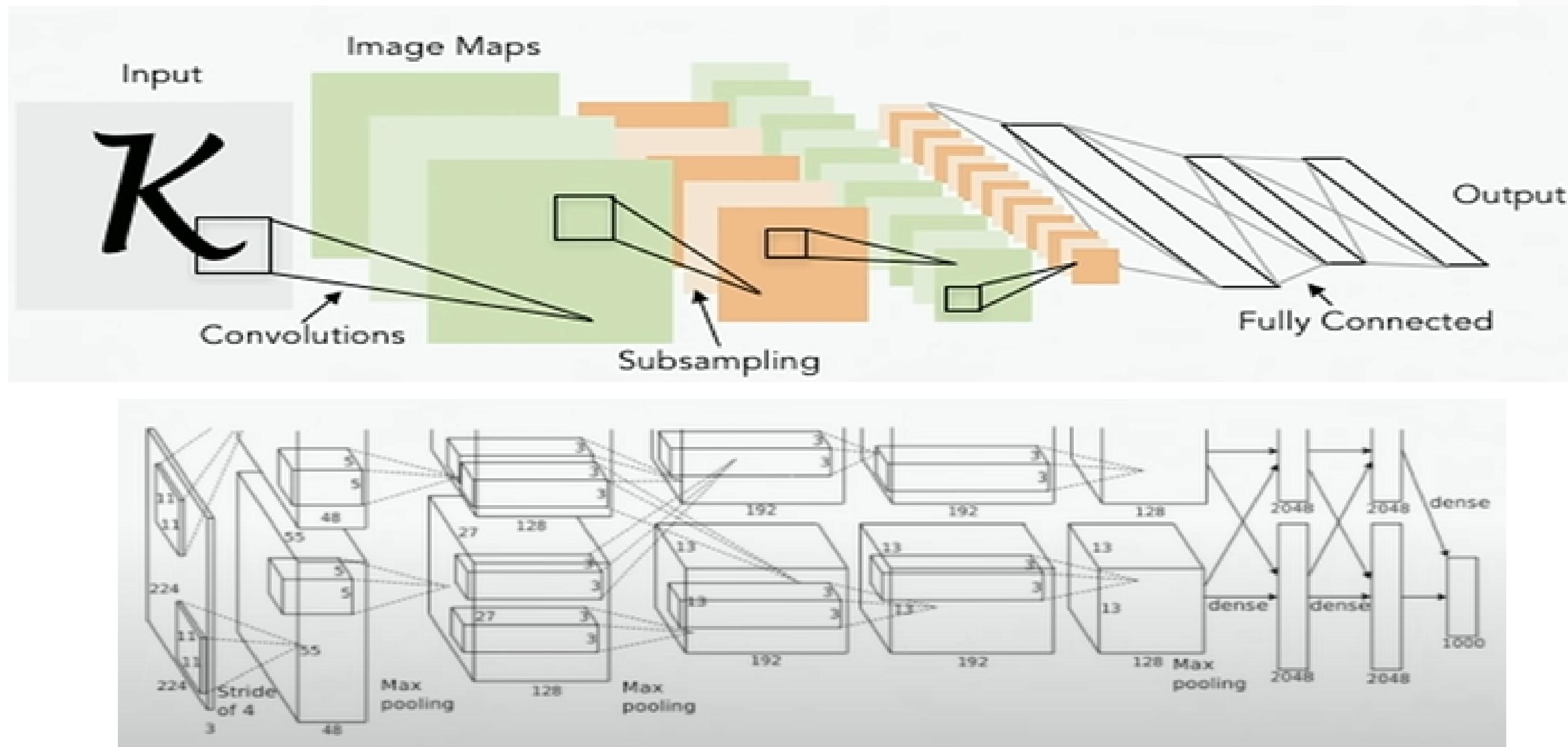
## 1. Desafio Imagenet 2012 por Krizhevsky;

- AlexNet - GPUs e Dropout no ImageNet Challenge



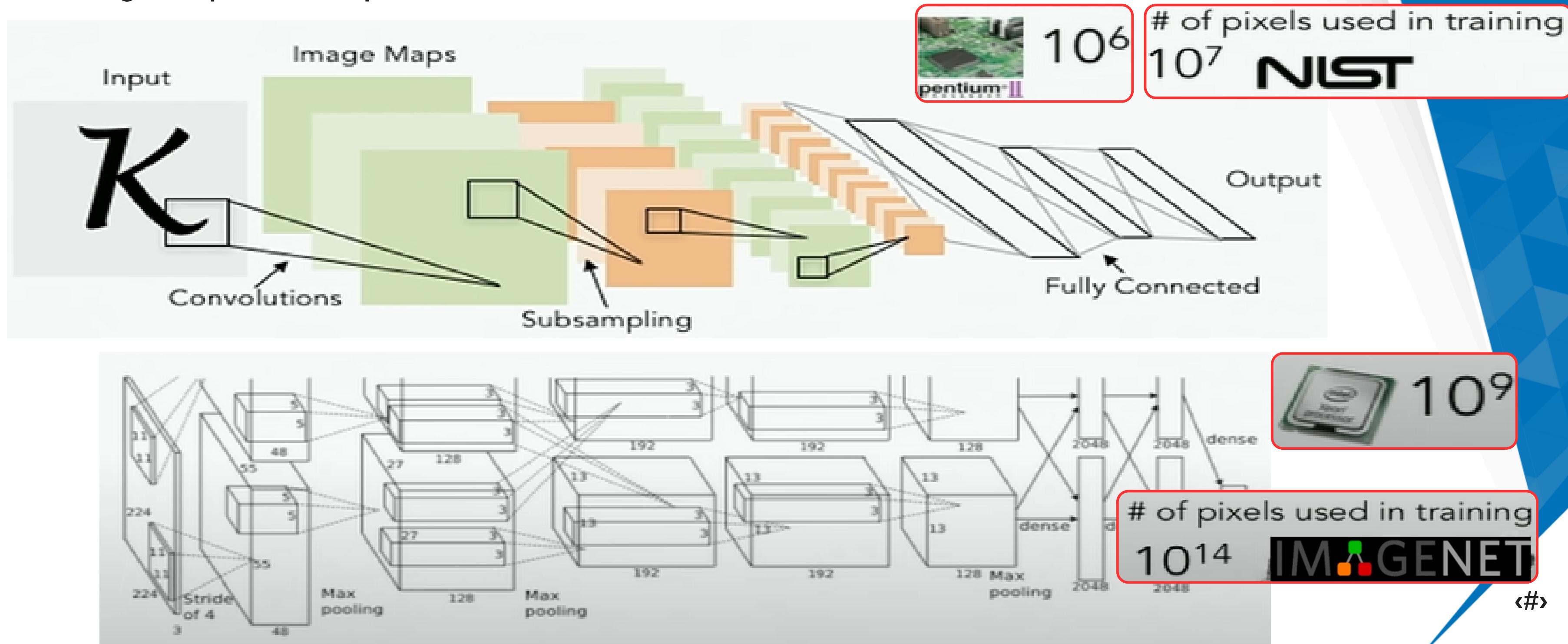
# Contextualização Histórica

1. Um dos primeiros trabalhos foi de Yann LeCun em 1989 para reconhecimento de dígitos para cheques.
2. Desafio Imagenet 2012 por Krizhevsky;

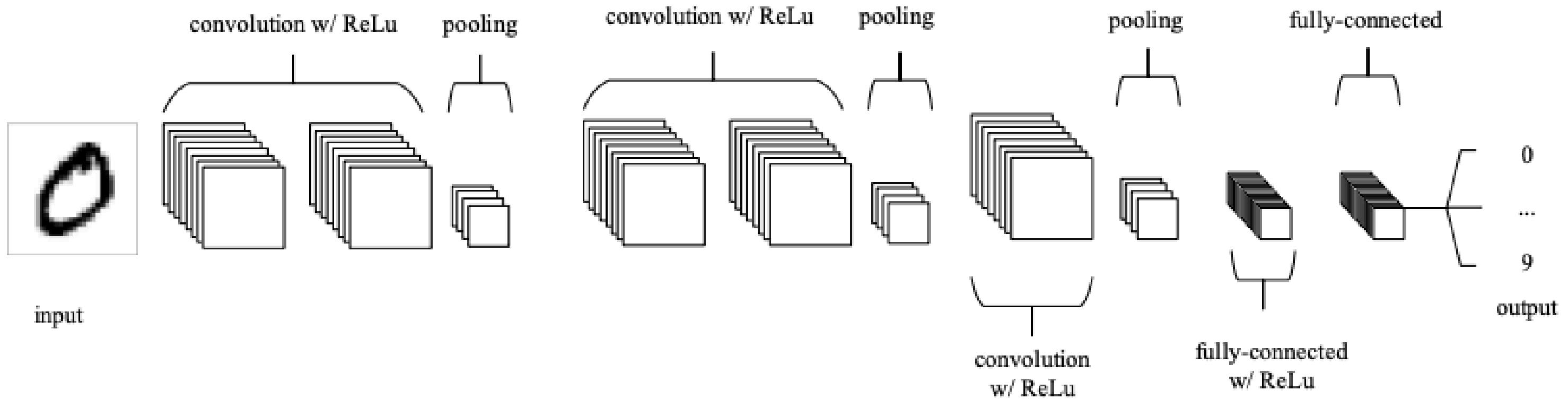


# Contextualização Histórica

1. Desafio Imagenet 2012 por Krizhevsky;
2. Um dos primeiros trabalhos foi de Yann LeCun em 1989 para reconhecimento de dígitos para cheques. [Link](#)



# Arquitetura geral da CNN



# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

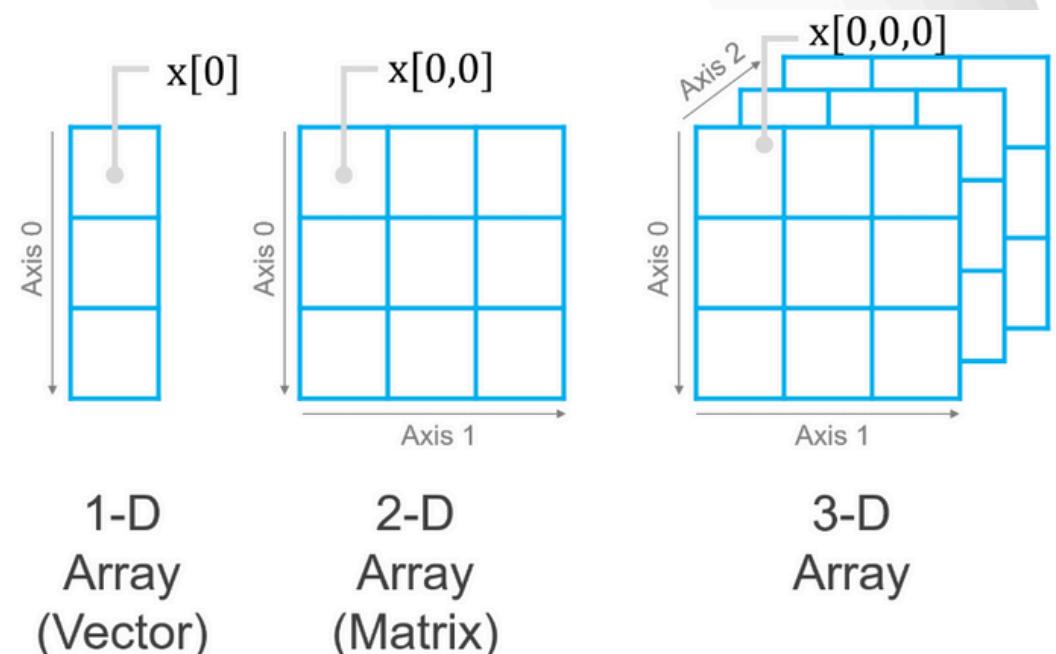
- Camada de Entrada (***Input layer***)
  - Responsável por receber os **dados brutos**, como imagens ou sequências.
- Camada de Convolução (***Convolution layer***)
  - Realiza operações de convolução para **extrair características**, como bordas ou padrões locais, utilizando filtros ou kernels.
- Camada de Pooling (***Pooling layer***)
  - Reduz a **dimensionalidade** espacial dos dados (como altura e largura), mantendo as informações mais relevantes.
- Camada Totalmente Conectada (***Fully connected layer***)
  - Conecta todos os neurônios, permitindo que a rede faça **previsões** baseadas nos recursos (informações) extraídos.

# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

Camada de Entrada (*Input layer*):

- Como é a entrada?
  - Ex: Uma imagem em escala de cinza de  $28 \times 28$  pixels.
- Diferença em relação às Redes MLP:
  - Diferente de uma MLP, a entrada não é achatada (transformada) em um vetor 1D - shape(784,).
  - Em vez disso, é apresentada à rede como uma matriz - shape(28, 28).
- Benefício desse formato:
  - Preservar o formato original da imagem (matriz) facilita a captura das relações espaciais na imagem, como proximidade e padrões visuais.

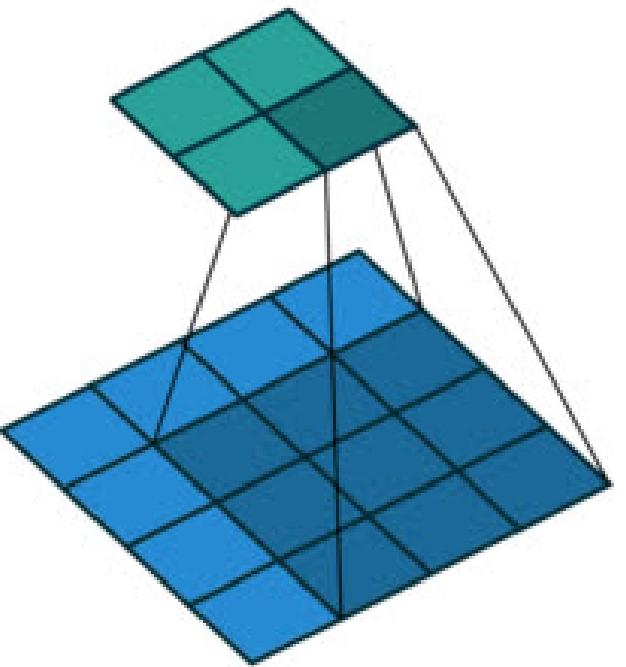


# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

Camada de Convolução (***Convolution layer***):

- Filtros (Kernels)
  - A rede é composta por múltiplos filtros.
  - Para imagens 2D, os filtros também são 2D.
- Exemplo de Filtro
  - Consideramos um filtro de  $3 \times 3$  ("kernel 3", totalizando 9 valores).
  - Os valores do filtro são definidos aleatoriamente, podendo ser 0 ou 1.
- Processo de Convolução:
  - Posicionar o filtro > multiplicar > somar > mover o filtro = ***feature map***



$$\begin{matrix} 1 & 0 & -1 & 14 & 19 \\ 1 & 0 & -1 & 196 & 169 \\ 1 & 0 & -1 & 41 & 111 \\ 209 & 30 & 201 & 23 & 203 \\ 58 & 79 & 218 & 59 & 118 \end{matrix} * \begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix} = \begin{matrix} 2 \end{matrix}$$

The diagram illustrates a convolution operation. On the left is a 5x5 input feature map with values ranging from 1 to 209. In the center is a 3x3 kernel with values 1, 0, -1 repeated three times. An asterisk (\*) indicates the multiplication of the input by the kernel. An equals sign (=) indicates the result. To the right is a single value, 2, representing the output of the convolution step. The bottom right corner of the slide has a page number <#>.

# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

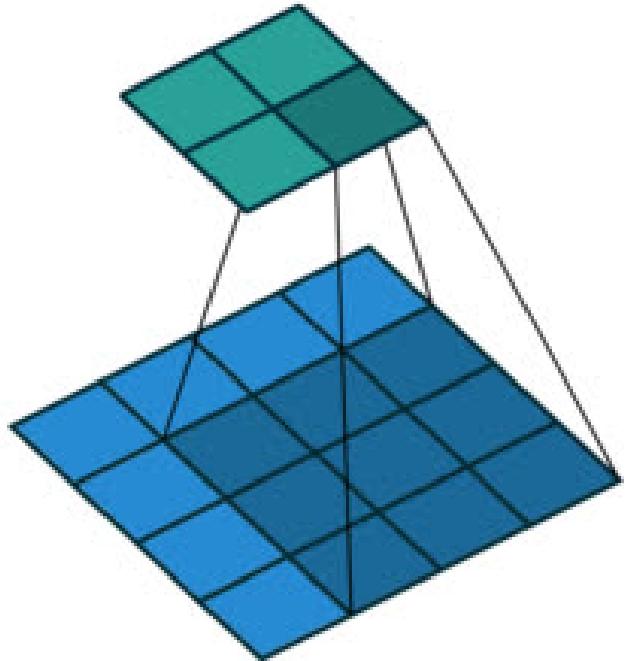
Camada de Convolução (***Convolution layer***): Parâmetros

- **Tamanho do Kernel:**
  - Dimensão do filtro que extrai características da imagem na convolução.
- **Stride:**
  - Passo de deslocamento do kernel sobre a entrada durante a convolução.
- **Padding:**
  - Adição de bordas na entrada para preservar dimensões ou controlar o tamanho da saída.
- **Função de ativação:**
  - Operador não linear que introduz complexidade e permite modelar relações não lineares.

# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

Camada de Convolução (***Convolution layer***): Parâmetros



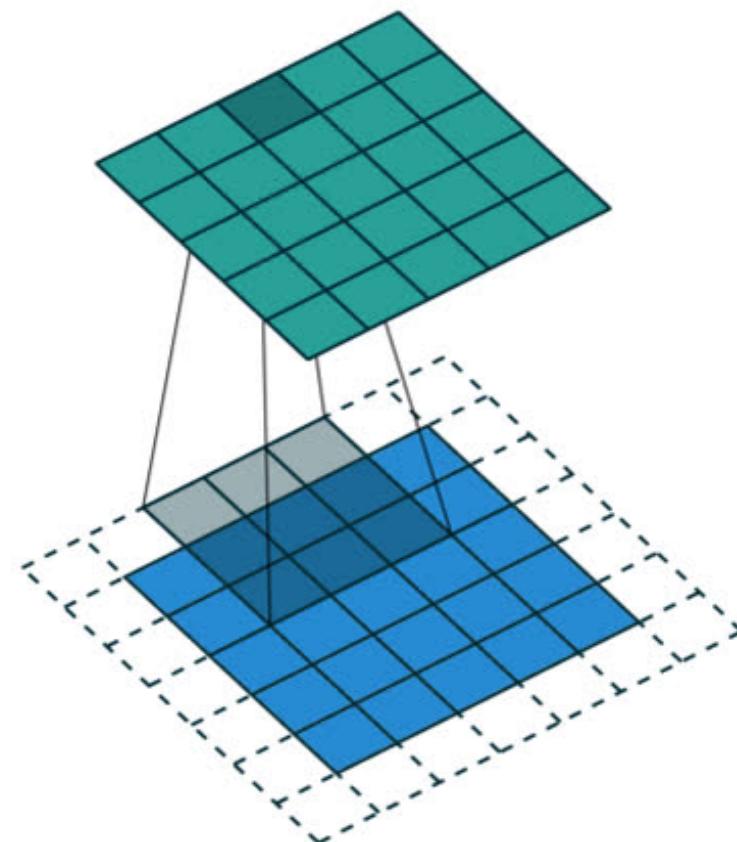
Ao aplicar um kernel com tamanho 3 em uma imagem com tamanho  $4 \times 4$ , obtemos um feature map de tamanho  $2 \times 2$ , ou seja, perdemos o tamanho da imagem

# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

Camada de Convolução (***Convolution layer***): Parâmetros

**Padding:**



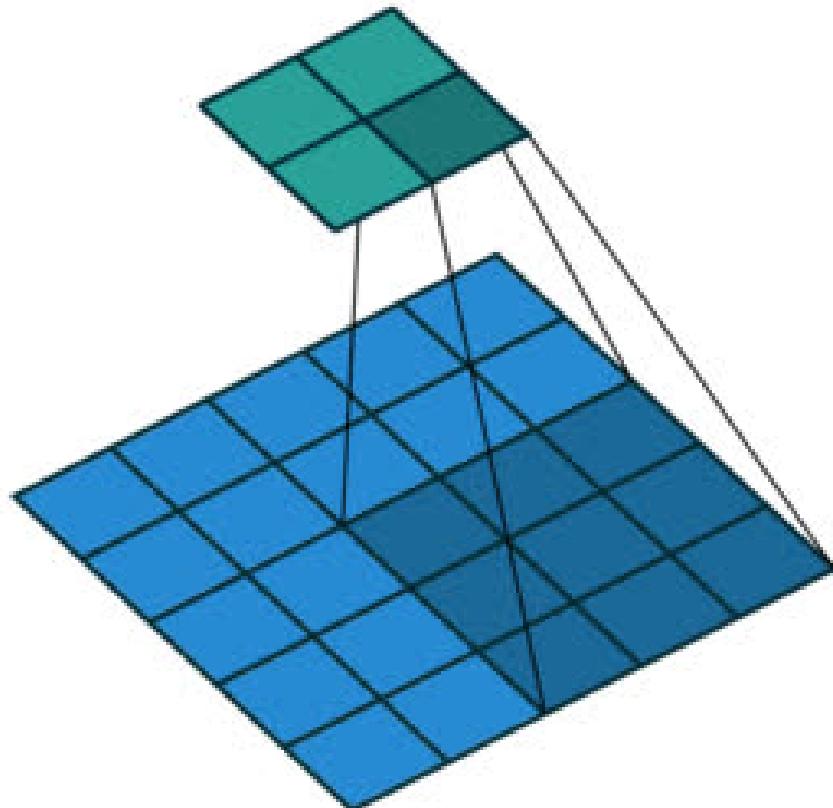
Usamos o parâmetro Padding para preservar o tamanho da imagem, isso é feito aumentando o tamanho da imagem original antes da convolução.

# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

Camada de Convolução (***Convolution layer***): Parâmetros

**Stride:**



Usamos o parâmetro Stride para aumentar ou diminuir o passo em que o kernel percorre a imagem.  
Se o Stride aumenta, o tamanho do feature map diminui.

# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

Camada de Convolução (***Convolution layer***): Parâmetros

Função de ativação:

0	1	-4
0	2	0
-1	4	3



Filter output

0	1	0
0	2	0
0	4	3

Filter output after ReLU

Usamos a função de ativação para introduzir não-linearidade na rede.

# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

Camada de Pooling (***Pooling layer***):

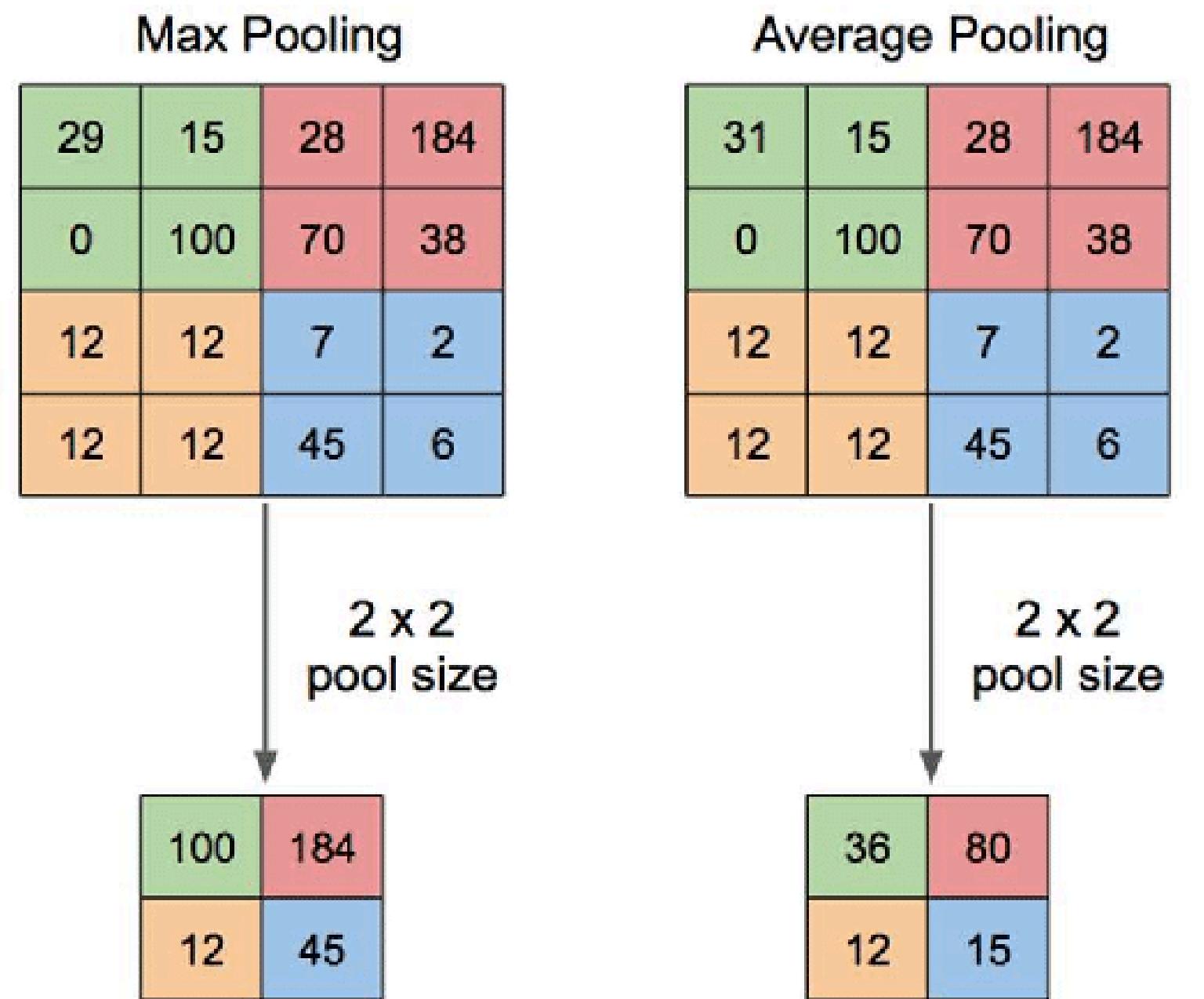
- Função da Camada de Pooling
  - Realiza downsampling para reduzir a dimensionalidade espacial da entrada.
- Benefícios:
  - Redução do Número de Parâmetros: Menos dados para processar.
  - Diminuição do Tempo de Treinamento/Cálculo: Reduz a carga computacional.
  - Menor Probabilidade de Overfitting: Simplifica o modelo, tornando-o mais generalizável.
- Pooling Mais Popular: Max Pooling
  - Utiliza um filtro  $2 \times 2$  com um stride de 2.
  - Retorna o maior valor ao deslizar sobre os dados de entrada.

\*Essa técnica mantém as informações mais significativas enquanto reduz a complexidade do modelo.

# Principais camadas da CNN

Uma **CNN** típica possui **4 camadas principais**:

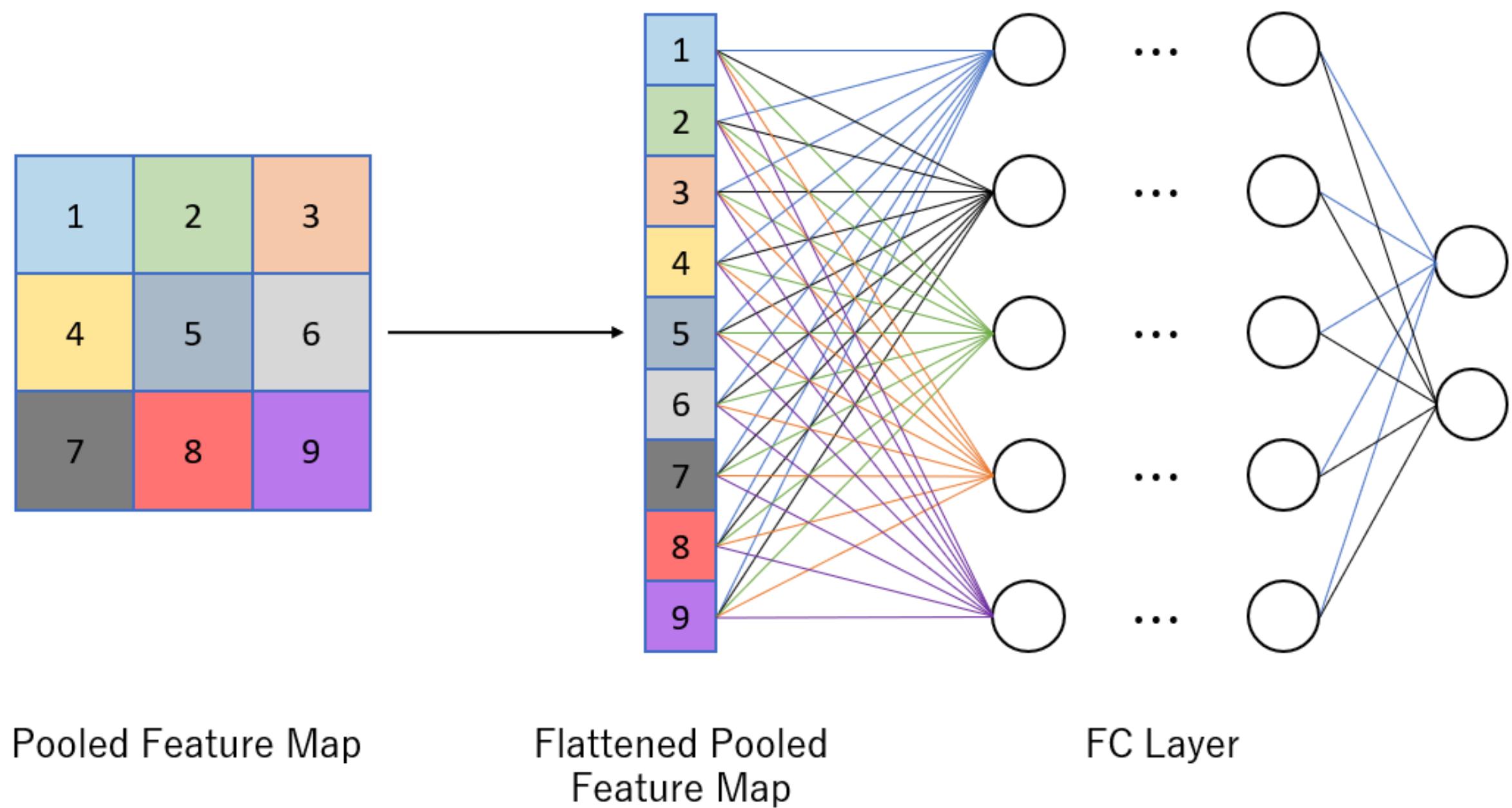
Camada de Pooling (***Pooling layer***):



# Principais camadas da CNN

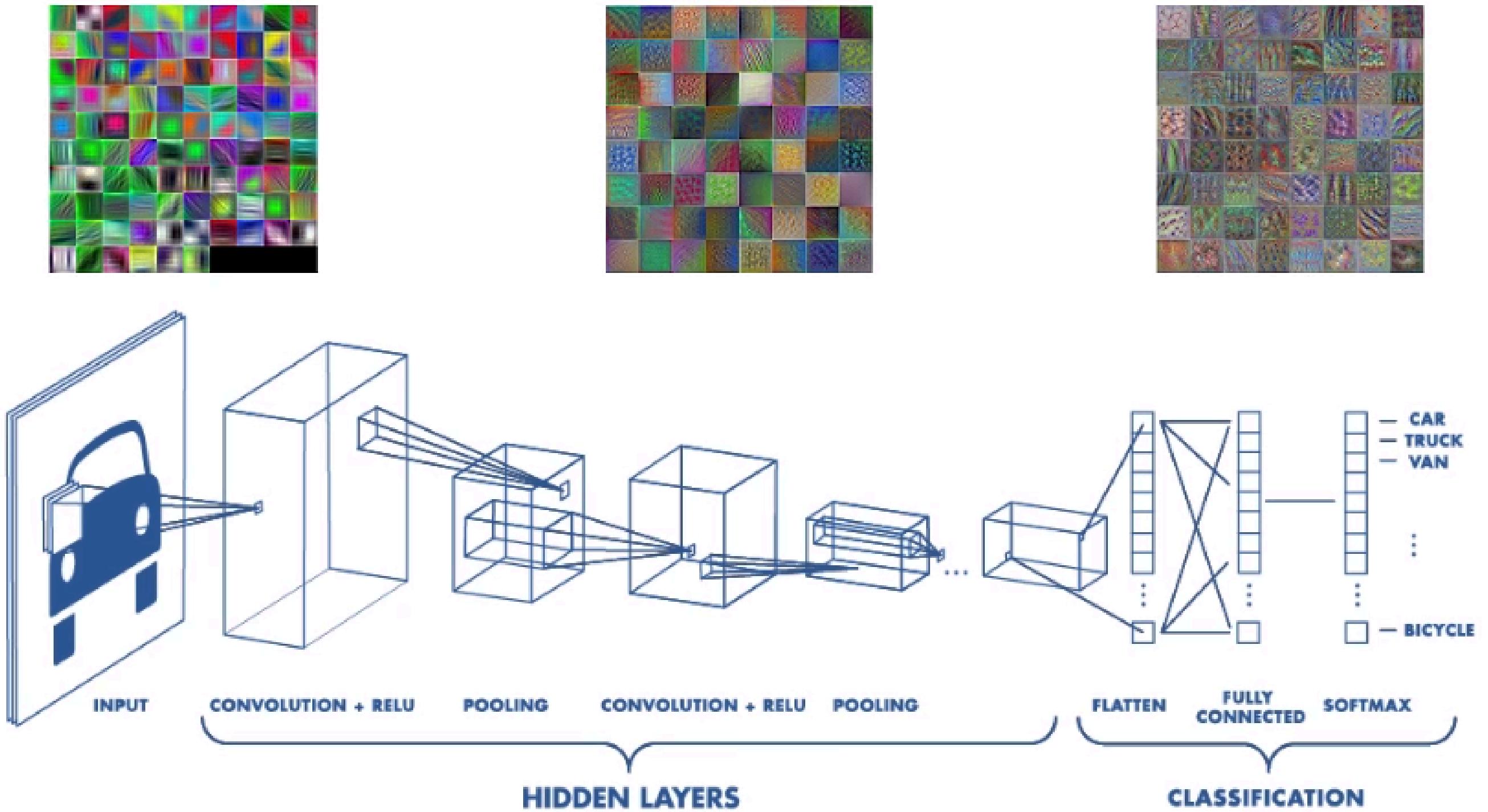
**Uma CNN típica possui 4 camadas principais:**

# Camada Totalmente Conectada (*Fully connected layer*)



# Principais camadas da CNN

Uma CNN típica possui 4 camadas principais:



# Principais camadas da CNN

Para brincar

[https://adamharley.com/nn\\_vis/cnn/3d.html](https://adamharley.com/nn_vis/cnn/3d.html)

[CNN Explainer](#)

# Obrigado!

Dúvidas ou sugestões:  
[email@apresentadores](mailto:email@apresentadores)

**INF**  
INSTITUTO DE  
INFORMÁTICA



