

INF721

2023/2

UFV

Aprendizado em Redes Neurais Profundas

A13: Redes Neurais Convolucionais

Logística

Avisos

- ▶ Projeto P3: Regularização e Otimização será publicado até sexta-feira!

Última aula

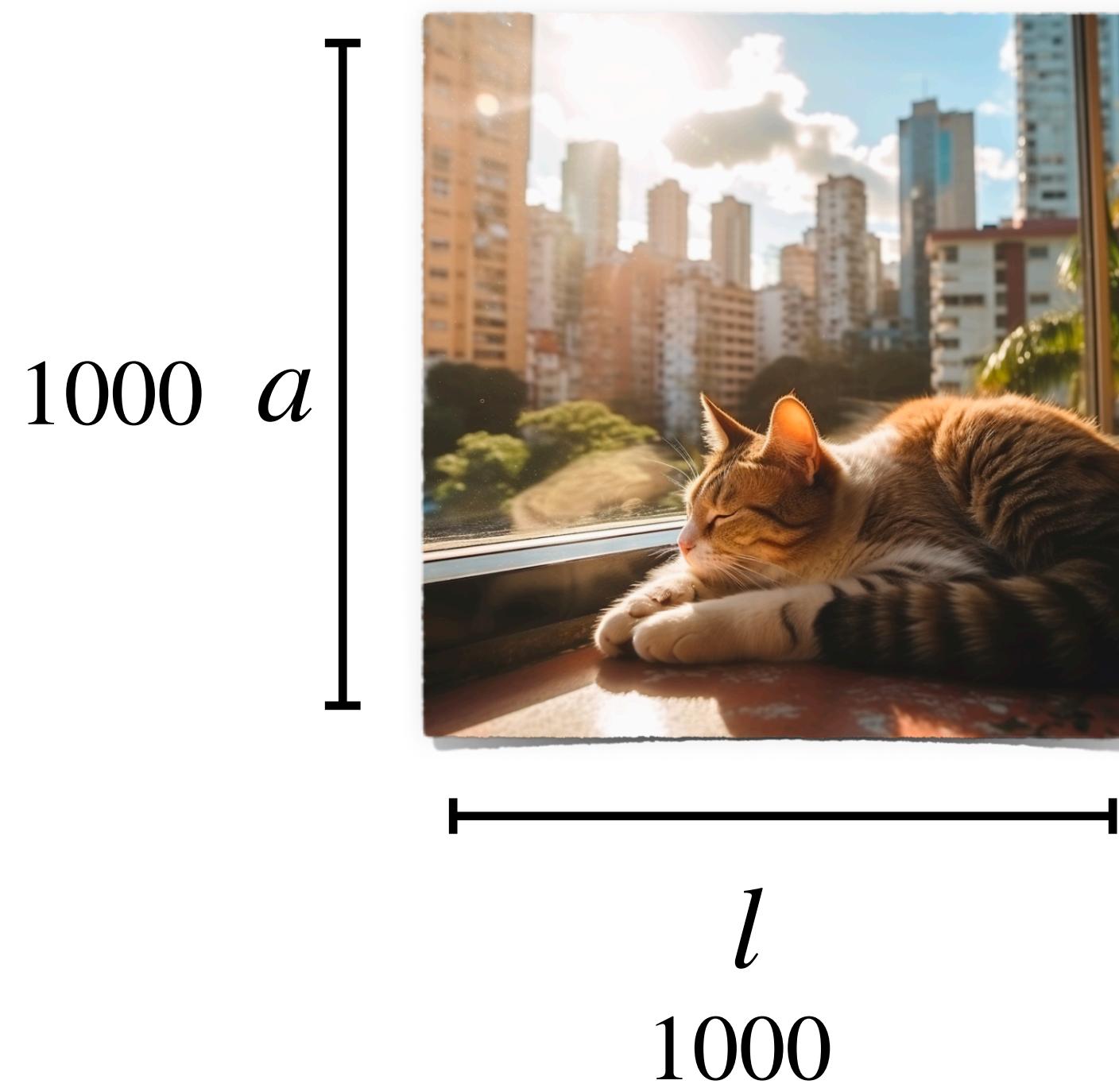
- ▶ MLP em Pytorch

Plano de Aula

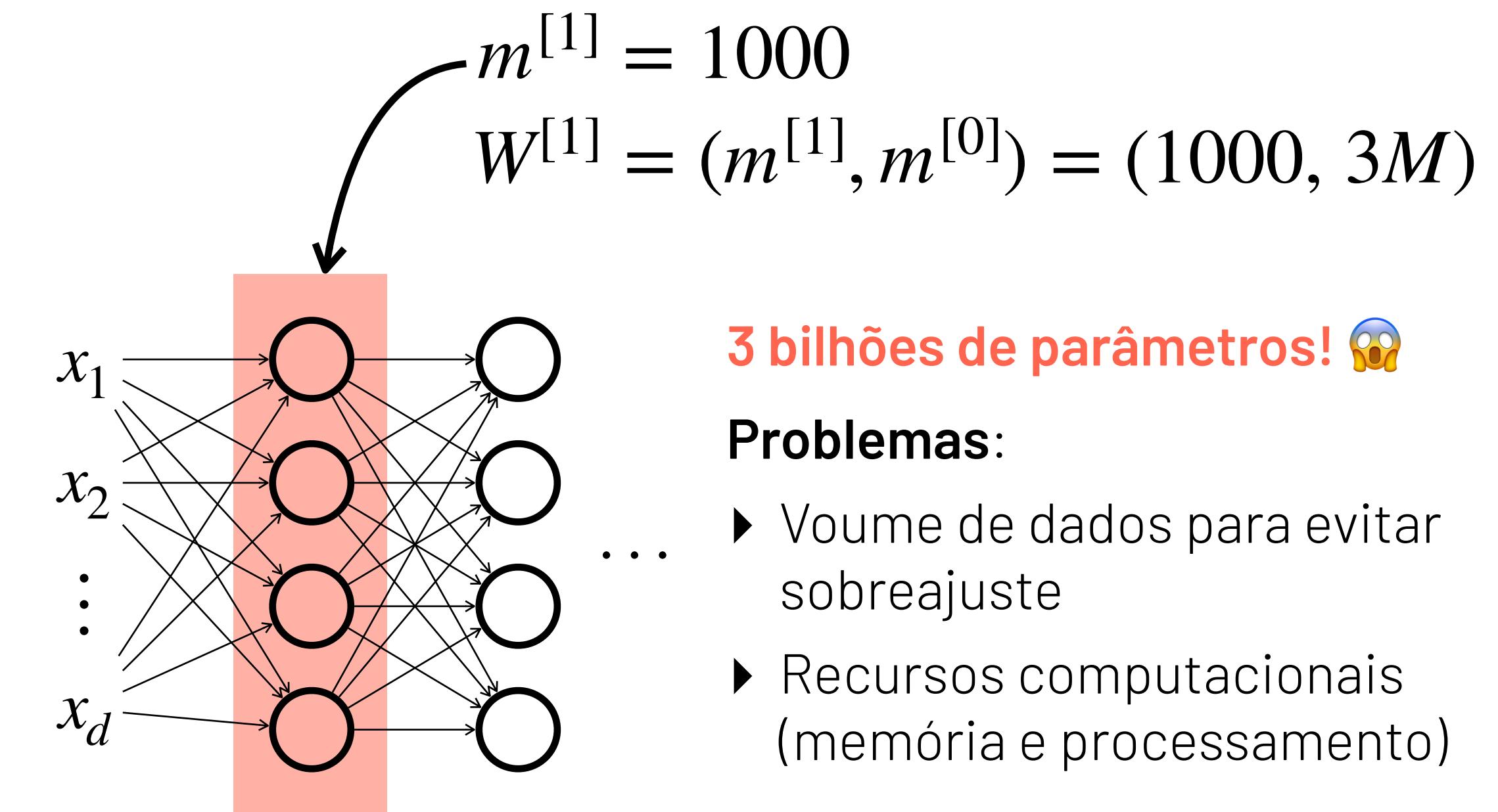
- ▶ Explosão de Parâmetros
- ▶ Filtros
- ▶ Convoluções
 - ▶ Preenchimento (Padding)
 - ▶ Convoluções Passadas (Strided Convolutions)
- ▶ Convoluções em Volumes
- ▶ Camadas de Padding
- ▶ Redes Neurais Convolucionais (CNN)

Explosão de parâmetros

Para processar imagens com MLPs, temos que transformá-las em vetores de características.



$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$



$$d = a \times l \times 3$$

$$d = 1000 \times 1000 \times 3 = 3M$$

3 bilhões de parâmetros! 🤯

Problemas:

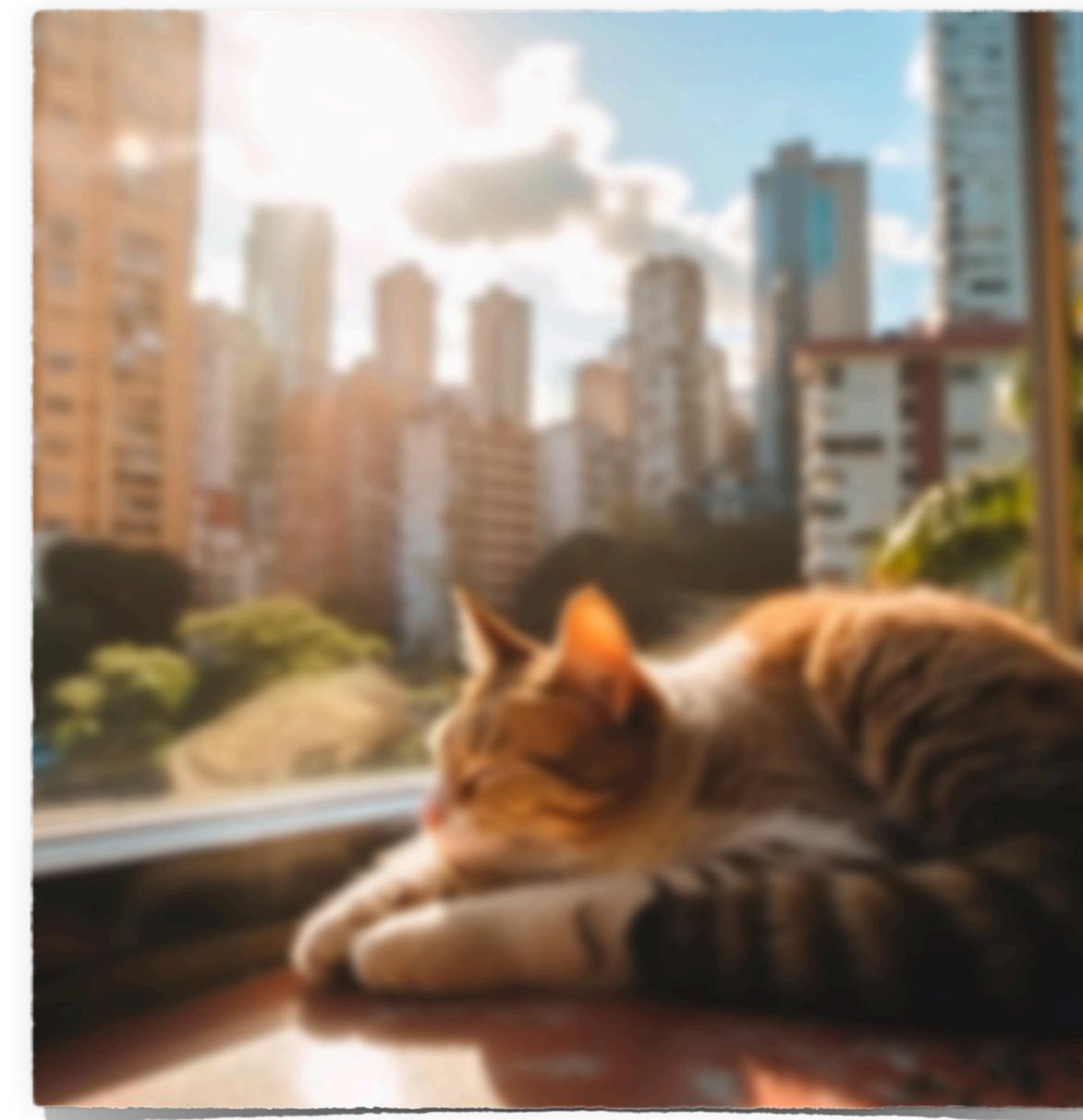
- ▶ Volume de dados para evitar sobreajuste
- ▶ Recursos computacionais (memória e processamento)

Convoluçãoes

Em processamento de imagens e visão computacional, convoluções são operações para aplicar **filtros** (i.e., transformações) em imagens.



Borrar
→
(blur)



Filtros

Um **filtro** (ou kernel) é uma pequena matriz (geralmente 3x3) de pesos que transforma um **pixel** com a soma ponderada dos pixels de sua vizinhança.

| | | | | |
|--|-----|-----|-----|--|
| | | | | |
| | 206 | 205 | 247 | |
| | 144 | 161 | 137 | |
| | 192 | 154 | 75 | |
| | | | | |

Pixel original (161)
e sua vizinhança

| | | |
|--------|--------|--------|
| 0,0625 | 0,125 | 0,0625 |
| 0,125 | 0,25 | 0,125 |
| 0,0625 | 0,0125 | 0,0625 |

*

Filtro (blur)

$$= \sum_{i=1}^3 \sum_{j=1}^3 = m_{i,j} * k_{i,j} =$$

$$\begin{aligned} & 206 * 0,0625 + 205 * 0,125 + 247 * 0,0625 + \\ & 144 * 0,125 + 161 * 0,25 + 137 * 0,125 + \\ & 192 * 0,0625 + 154 * 0,125 + 75 * 0,0625 = \\ & 178 \end{aligned}$$

| | | | | |
|--|-----|-----|-----|--|
| | | | | |
| | 206 | 205 | 247 | |
| | 144 | 178 | 137 | |
| | 192 | 154 | 75 | |
| | | | | |

Pixel transformado (178)
e sua vizinhança

Convoluçãoes

Uma **convolução** é uma operação entre uma imagem e um filtro que consiste em aplicar um filtro para cada pixel de uma imagem.

| | | | | |
|-----|-----|-----|-----|-----|
| 206 | 205 | 247 | 245 | 244 |
| 244 | 161 | 137 | 244 | 254 |
| 192 | 154 | 75 | 200 | 249 |
| 90 | 109 | 96 | 143 | 223 |
| 67 | 69 | 107 | 196 | 236 |

Imagen Original
 (5×5)

$$* \begin{matrix} 0,0625 & 0,125 & 0,0625 \\ 0,125 & 0,25 & 0,125 \\ 0,0625 & 0,0125 & 0,0625 \end{matrix}$$

Filtro (blur)

$$= \sum_{i=1}^3 \sum_{j=1}^3 = m_{i,j} * k_{i,j} =$$

| | | |
|--|--|--|
| | | |
| | | |
| | | |

Imagen Transformada
 (3×3)

Convoluçãoes

Uma **convolução** é uma operação entre uma imagem e um filtro que consiste em aplicar um filtro para cada pixel de uma imagem.

| | | | | |
|-----|-----|-----|-----|-----|
| 206 | 205 | 247 | 245 | 244 |
| 244 | 161 | 137 | 244 | 254 |
| 192 | 154 | 75 | 200 | 249 |
| 90 | 109 | 96 | 143 | 223 |
| 67 | 69 | 107 | 196 | 236 |

Imagen Original
 (5×5)

$$* \quad \begin{matrix} 0,0625 & 0,125 & 0,0625 \\ 0,125 & 0,25 & 0,125 \\ 0,0625 & 0,0125 & 0,0625 \end{matrix}$$

Filtro (blur)

$$= \sum_{i=1}^3 \sum_{j=1}^3 = m_{i,j} * k_{i,j} =$$

| | | |
|-----|--|--|
| 178 | | |
| | | |
| | | |

Imagen Transformada
 (3×3)

Convoluçãoes

Uma **convolução** é uma operação entre uma imagem e um filtro que consiste em aplicar um filtro para cada pixel de uma imagem.

| | | | | |
|-----|-----|-----|-----|-----|
| 206 | 205 | 247 | 245 | 244 |
| 244 | 161 | 137 | 244 | 254 |
| 192 | 154 | 75 | 200 | 249 |
| 90 | 109 | 96 | 143 | 223 |
| 67 | 69 | 107 | 196 | 236 |

Imagen Original
(5 × 5)

*

| | | |
|--------|--------|--------|
| 0,0625 | 0,125 | 0,0625 |
| 0,125 | 0,25 | 0,125 |
| 0,0625 | 0,0125 | 0,0625 |

Filtro (blur)

$$= \sum_{i=1}^3 \sum_{j=1}^3 = m_{i,j} * k_{i,j} =$$

| | | |
|-----|-----|--|
| 178 | 175 | |
| | | |
| | | |

Imagen Transformada
(3 × 3)

Convoluçãoes

Uma **convolução** é uma operação entre uma imagem e um filtro que consiste em aplicar um filtro para cada pixel de uma imagem.

| | | | | |
|-----|-----|-----|-----|-----|
| 206 | 205 | 247 | 245 | 244 |
| 244 | 161 | 137 | 244 | 254 |
| 192 | 154 | 75 | 200 | 249 |
| 90 | 109 | 96 | 143 | 223 |
| 67 | 69 | 107 | 196 | 236 |

Imagen Original
 (5×5)

*

| | | |
|--------|--------|--------|
| 0,0625 | 0,125 | 0,0625 |
| 0,125 | 0,25 | 0,125 |
| 0,0625 | 0,0125 | 0,0625 |

Filtro (blur)

$$= \sum_{i=1}^3 \sum_{j=1}^3 = m_{i,j} * k_{i,j} =$$

| | | |
|-----|-----|-----|
| 178 | 175 | 216 |
| | | |
| | | |

Imagen Transformada
 (3×3)

Convoluçãoes

Uma **convolução** é uma operação entre uma imagem e um filtro que consiste em aplicar um filtro para cada pixel de uma imagem.

| | | | | |
|-----|-----|-----|-----|-----|
| 206 | 205 | 247 | 245 | 244 |
| 244 | 161 | 137 | 244 | 254 |
| 192 | 154 | 75 | 200 | 249 |
| 90 | 109 | 96 | 143 | 223 |
| 67 | 69 | 107 | 196 | 236 |

Imagen Original
(5 × 5)

| | | |
|--------|--------|--------|
| 0,0625 | 0,125 | 0,0625 |
| 0,125 | 0,25 | 0,125 |
| 0,0625 | 0,0125 | 0,0625 |

*

Filtro (blur)

$$= \sum_{i=1}^3 \sum_{j=1}^3 = m_{i,j} * k_{i,j} =$$

| | | |
|-----|-----|-----|
| 178 | 175 | 216 |
| 141 | | |
| | | |

Imagen Transformada
(3 × 3)

Convoluçãoes

Uma **convolução** é uma operação entre uma imagem e um filtro que consiste em aplicar um filtro para cada pixel de uma imagem.

| | | | | |
|-----|-----|-----|-----|-----|
| 206 | 205 | 247 | 245 | 244 |
| 244 | 161 | 137 | 244 | 254 |
| 192 | 154 | 75 | 200 | 249 |
| 90 | 109 | 96 | 143 | 223 |
| 67 | 69 | 107 | 196 | 236 |

Imagen Original
 (5×5)

| | | |
|--------|--------|--------|
| 0,0625 | 0,125 | 0,0625 |
| 0,125 | 0,25 | 0,125 |
| 0,0625 | 0,0125 | 0,0625 |

*

Filtro (blur)

$$= \sum_{i=1}^3 \sum_{j=1}^3 = m_{i,j} * k_{i,j} =$$

| | | |
|-----|-----|-----|
| 178 | 175 | 216 |
| 141 | 133 | 183 |
| 106 | 117 | 167 |

Imagen Transformada
 (3×3)

Detecção de borda

Filtros podem ser utilizados para detecção de borda em imagens, o que é particularmente importante para extração de características.

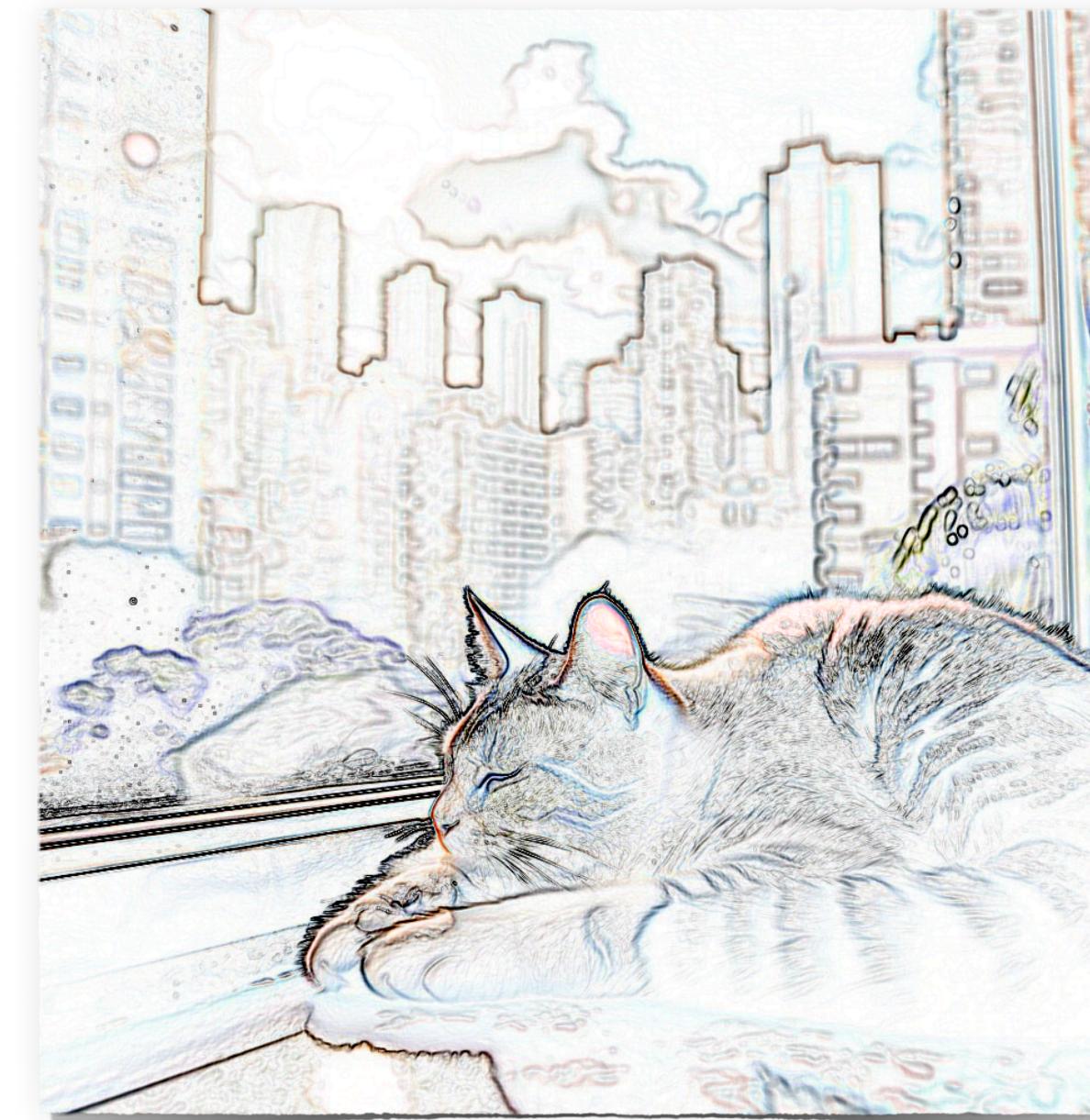


| | | |
|---|---|----|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

Vertical

| | | |
|----|----|----|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| -1 | -1 | -1 |

Horizontal



Desenvolvendo Filtros

Diferentes filtros de detecção de borda foram desenvolvidos científicamente pela comunidade de processamento de imagens.

| | | |
|---|---|----|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

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

Sobel

| | | |
|----|---|-----|
| 3 | 0 | -3 |
| 10 | 0 | -10 |
| 3 | 0 | -3 |

Scharr

Aprendendo Filtros

Redes Neurais Convolucionais (CNNs) **aprendem filtros** a partir de imagens e uma função de erro por meio do gradiente descendente.



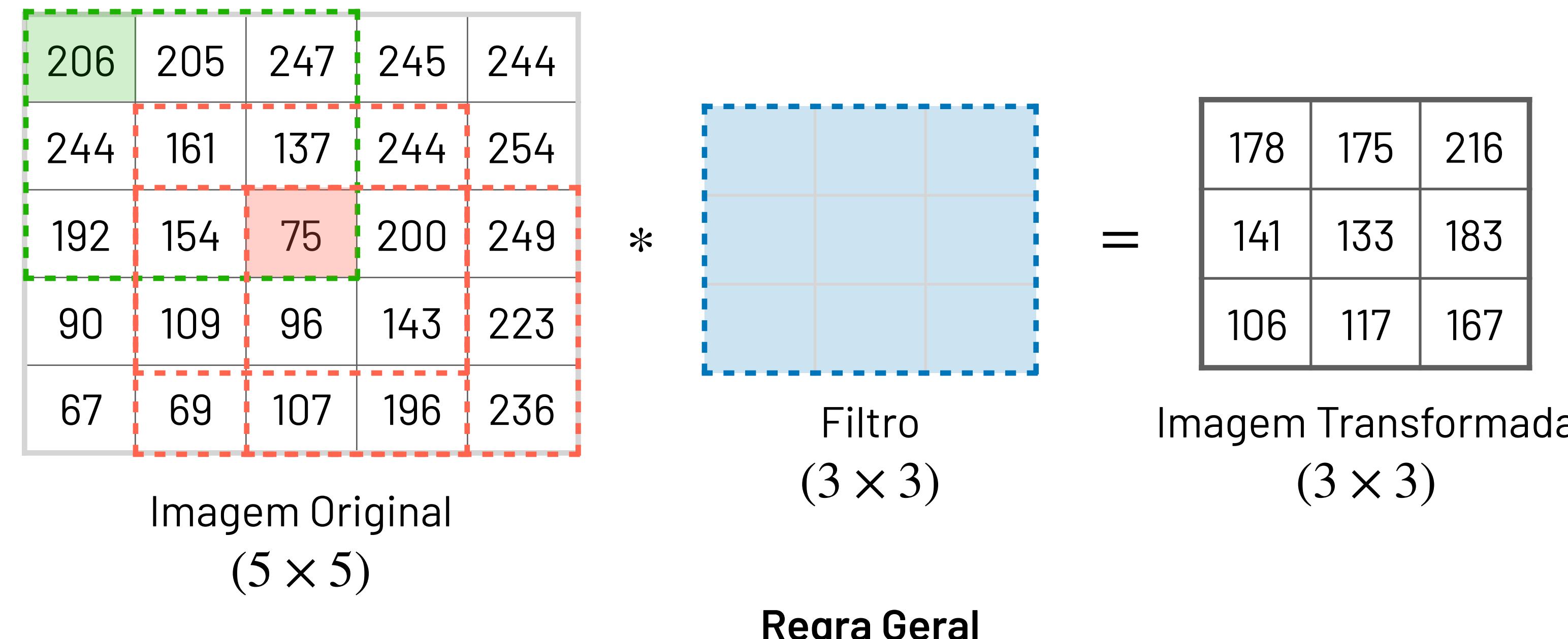
*

| | | |
|----|----|----|
| W1 | W2 | W3 |
| W4 | W5 | W6 |
| W7 | W8 | W9 |

Os pesos da uma CNN são organizados em filtros de convolução

Convoluçãoes reduzem o tamanho da imagem

- ▶ Aplicações consecutivas de convoluções podem tornar a imagem muito pequena (e.g., 1x1)
- ▶ Pixels dos cantos são menos compartilhados que pixels do meio



Regra Geral

$$(n \times n) * (f \times f) = (n - f + 1 \times n - f + 1)$$

Preenchimento (Padding)

Adicionar uma **borda** com p pixels na imagem original.

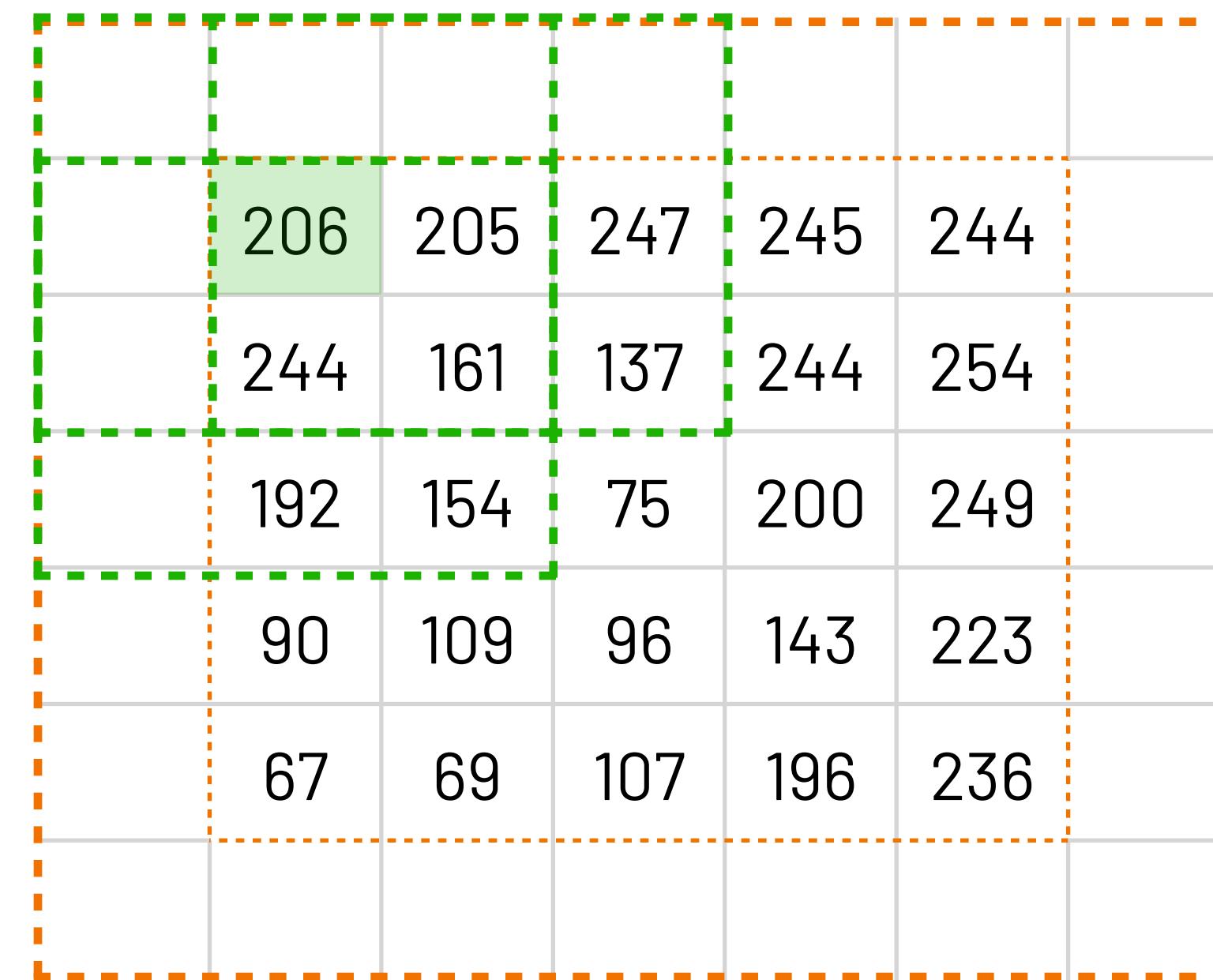


Imagen Original
(5 x 5)

The diagram illustrates the convolution process. On the left, the padded input image (5x5) is multiplied by a filter (3x3). The result is shown on the right, labeled as the transformed image (5x5).

*

Filtro
(3 x 3)

=

Imagen Transformada
(5 x 5)

Regra Geral

$$(n \times n) * (f \times f) = (n + 2p - f + 1 \times n + 2p - f + 1)$$

Preenchimento (Padding)

Para encontrar o valor de p que mantém o tamanho de uma imagem $n \times n$ após a convolução com um filtro de tamanho f (ímpar), basta resolver a seguinte equação:

$$n + 2p - f + 1 = n$$

$$2p - f + 1 = 0$$

$$2p = f - 1$$

$$p = \frac{f - 1}{2}$$

Convoluçãoes Passadas (*Strided Convolutions*)

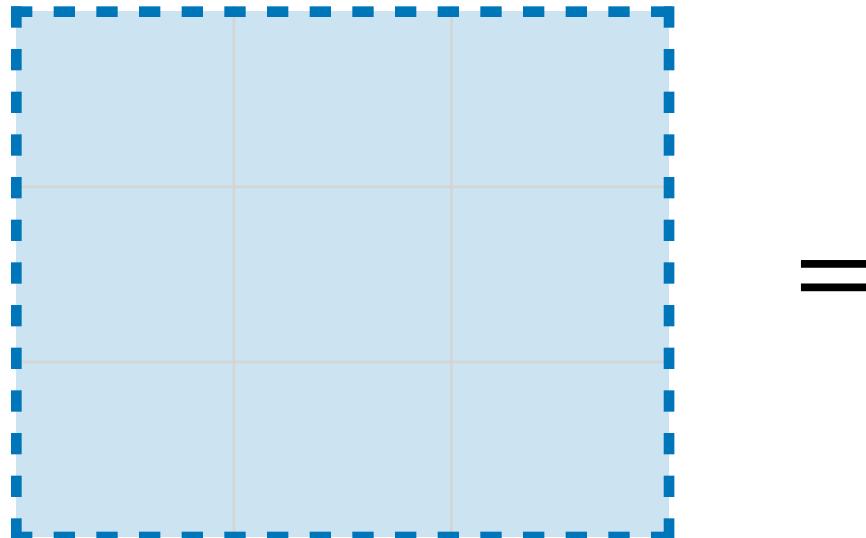
Convoluçãoes podem ser executadas com passos (strides) maiores do que 1.

stride = 2

| | | | | |
|-----|-----|-----|-----|-----|
| 206 | 205 | 247 | 245 | 244 |
| 244 | 161 | 137 | 244 | 254 |
| 192 | 154 | 75 | 200 | 249 |
| 90 | 109 | 96 | 143 | 223 |
| 67 | 69 | 107 | 196 | 236 |

Imagen Original
 (5×5)

*



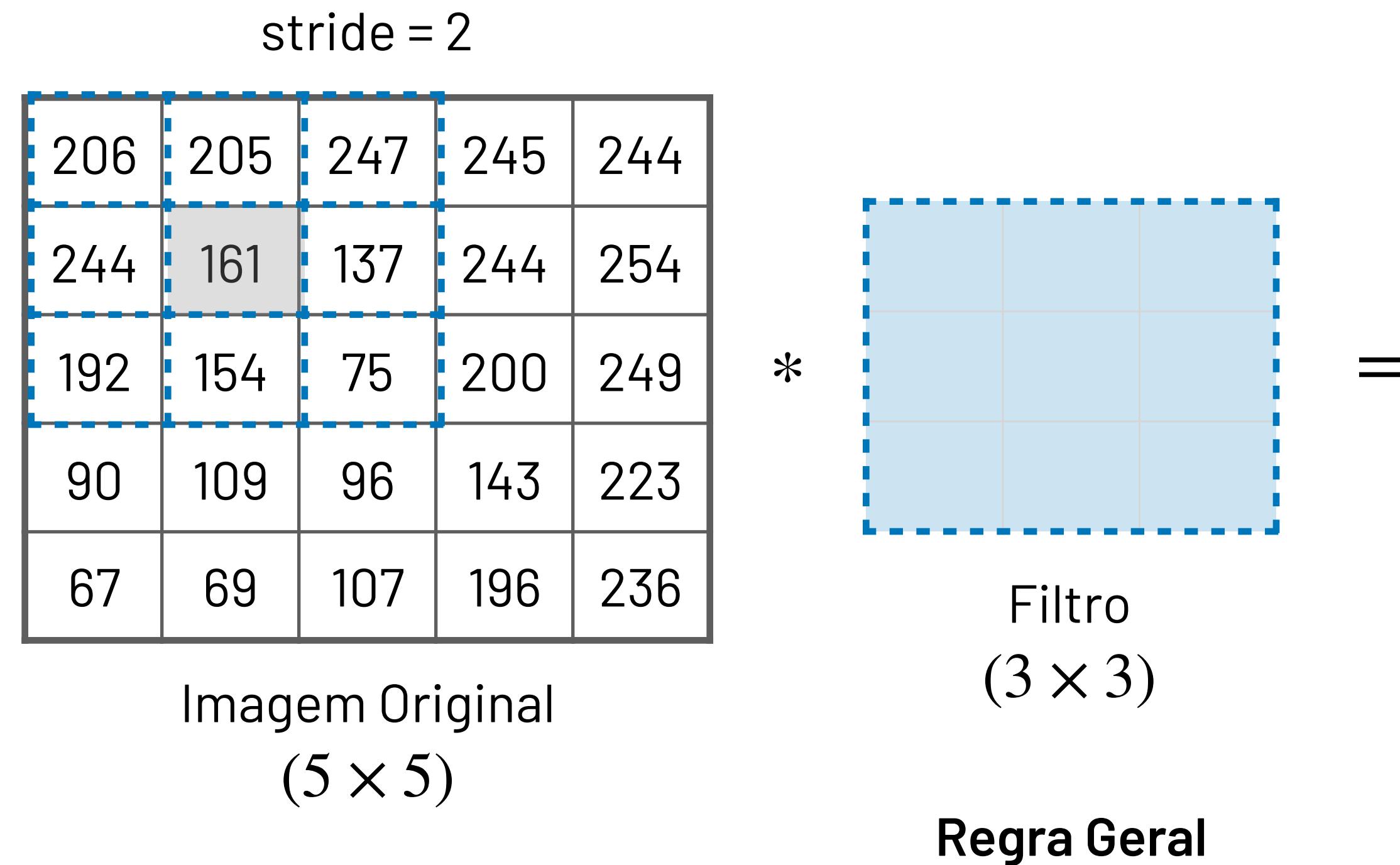
Filtro
 (3×3)

Regra Geral

$$(n \times n) * (f \times f) = \left(\frac{n + 2p - f}{s} + 1 \times \frac{n + 2p - f}{s} + 1 \right)$$

Convoluçãoes Passadas (*Strided Convolutions*)

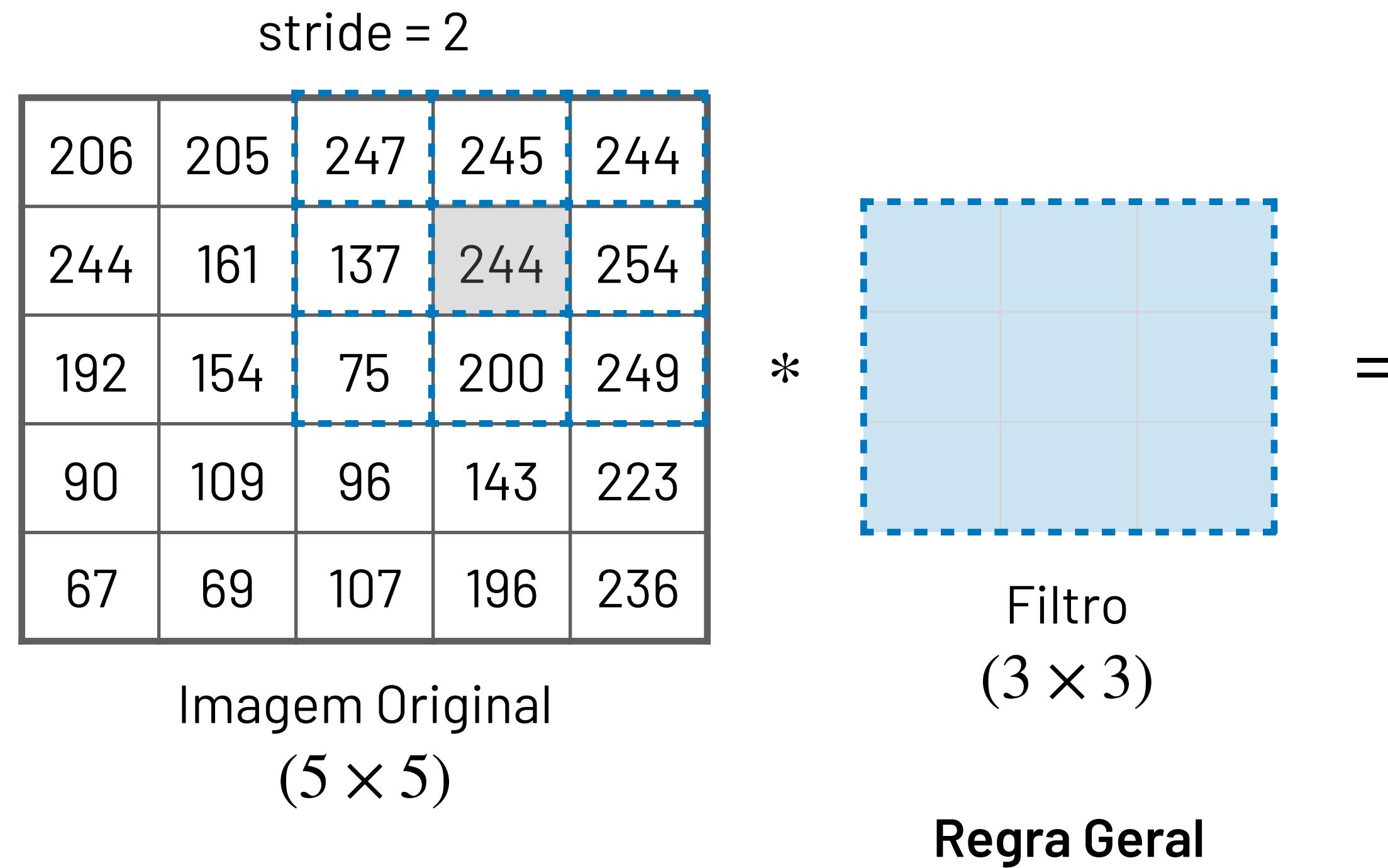
Convoluçãoes podem ser executadas com passos (strides) maiores do que 1.



$$(n \times n) * (f \times f) = \left(\frac{n + 2p - f}{s} + 1 \right) \times \left(\frac{n + 2p - f}{s} + 1 \right)$$

Convoluçãoes Passadas (*Strided Convolutions*)

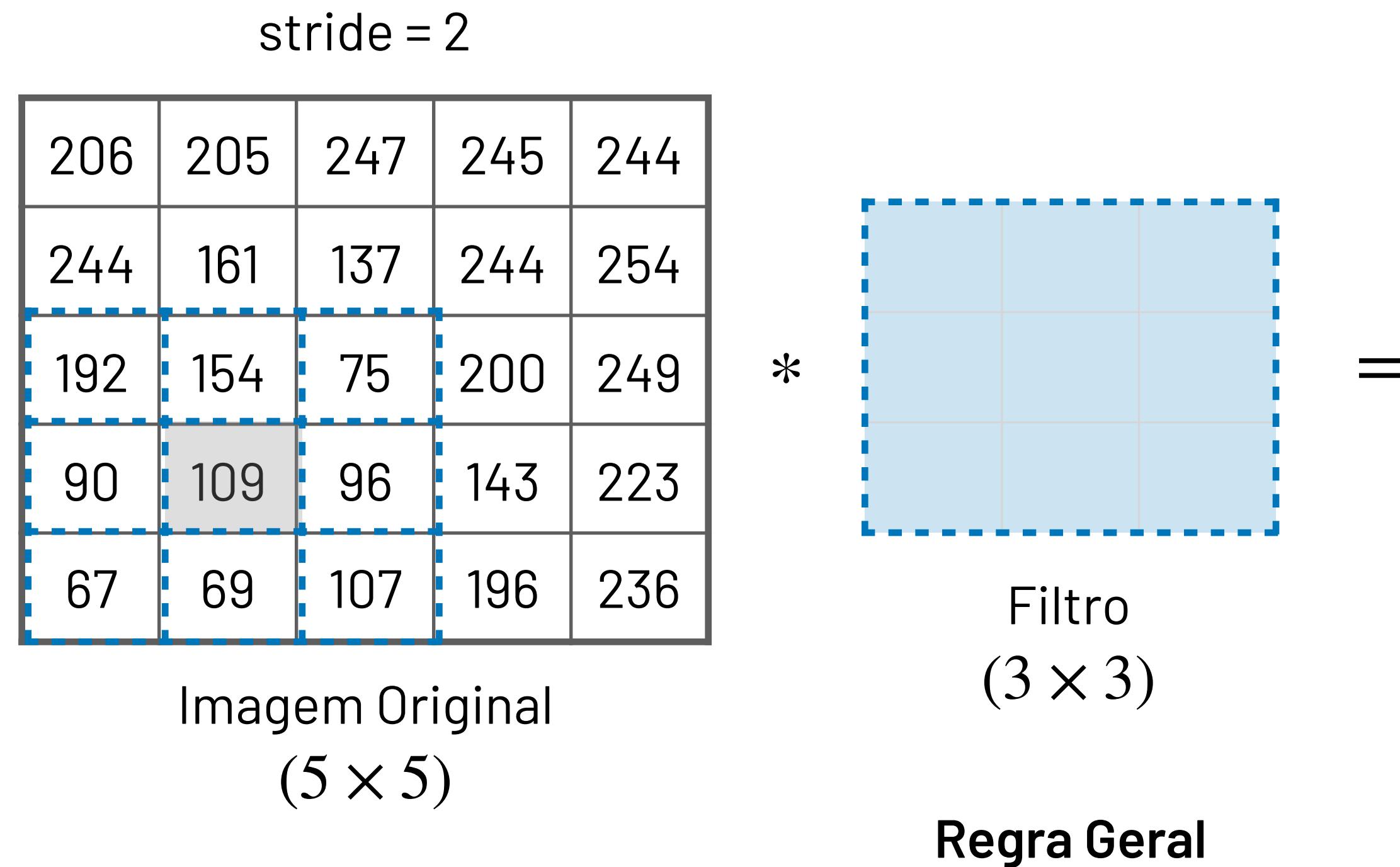
Convoluçãoes podem ser executadas com passos (strides) maiores do que 1.



$$(n \times n) * (f \times f) = \left(\frac{n + 2p - f}{s} + 1 \times \frac{n + 2p - f}{s} + 1 \right)$$

Convoluçãoes Passadas (*Strided Convolutions*)

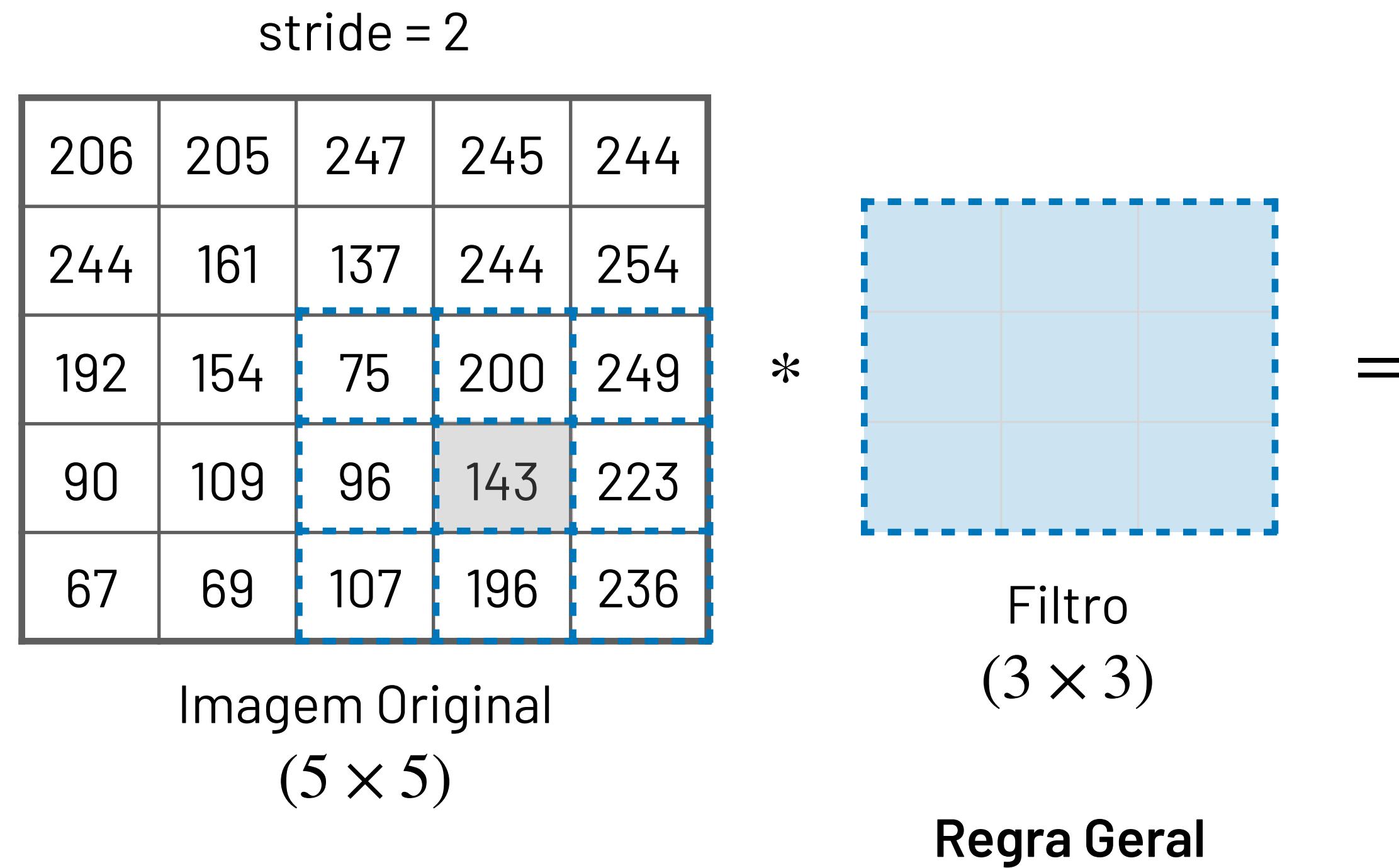
Convoluçãoes podem ser executadas com passos (strides) maiores do que 1.



$$(n \times n) * (f \times f) = \left(\frac{n + 2p - f}{s} + 1 \times \frac{n + 2p - f}{s} + 1 \right)$$

Convoluçãoes Passadas (*Strided Convolutions*)

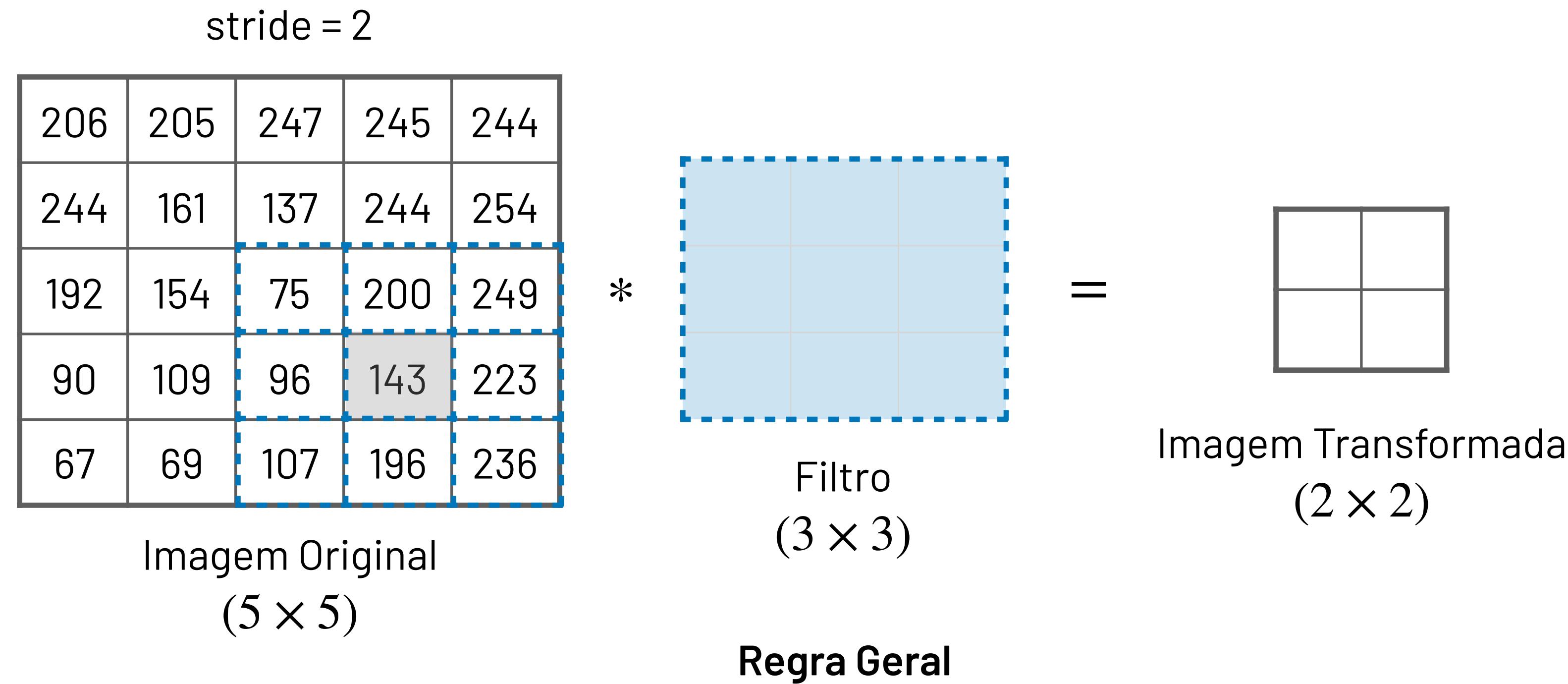
Convoluçãoes podem ser executadas com passos (strides) maiores do que 1.



$$(n \times n) * (f \times f) = \left(\frac{n + 2p - f}{s} + 1 \times \frac{n + 2p - f}{s} + 1 \right)$$

Convoluçãoes Passadas (*Strided Convolutions*)

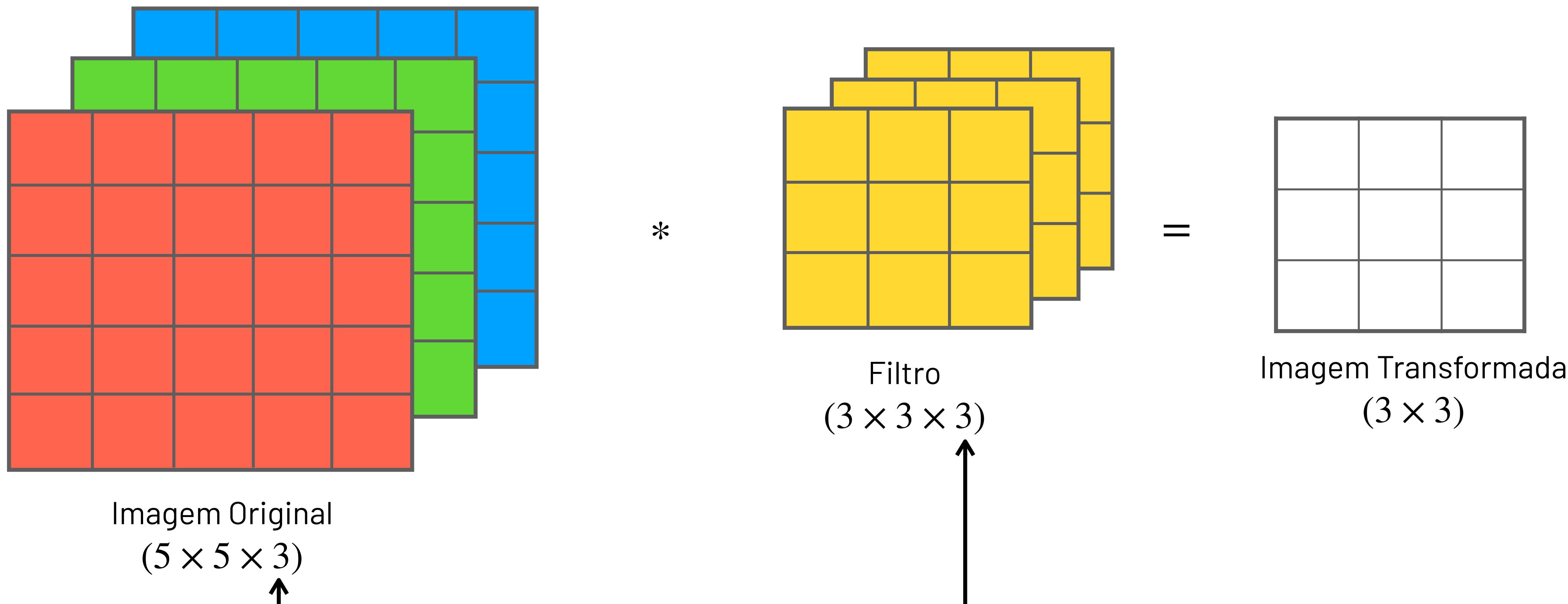
Convoluçãoes podem ser executadas com passos (strides) maiores do que 1.



$$(n \times n) * (f \times f) = \left(\frac{n + 2p - f}{s} + 1 \right) \times \left(\frac{n + 2p - f}{s} + 1 \right)$$

Convoluçãoes em Volumes

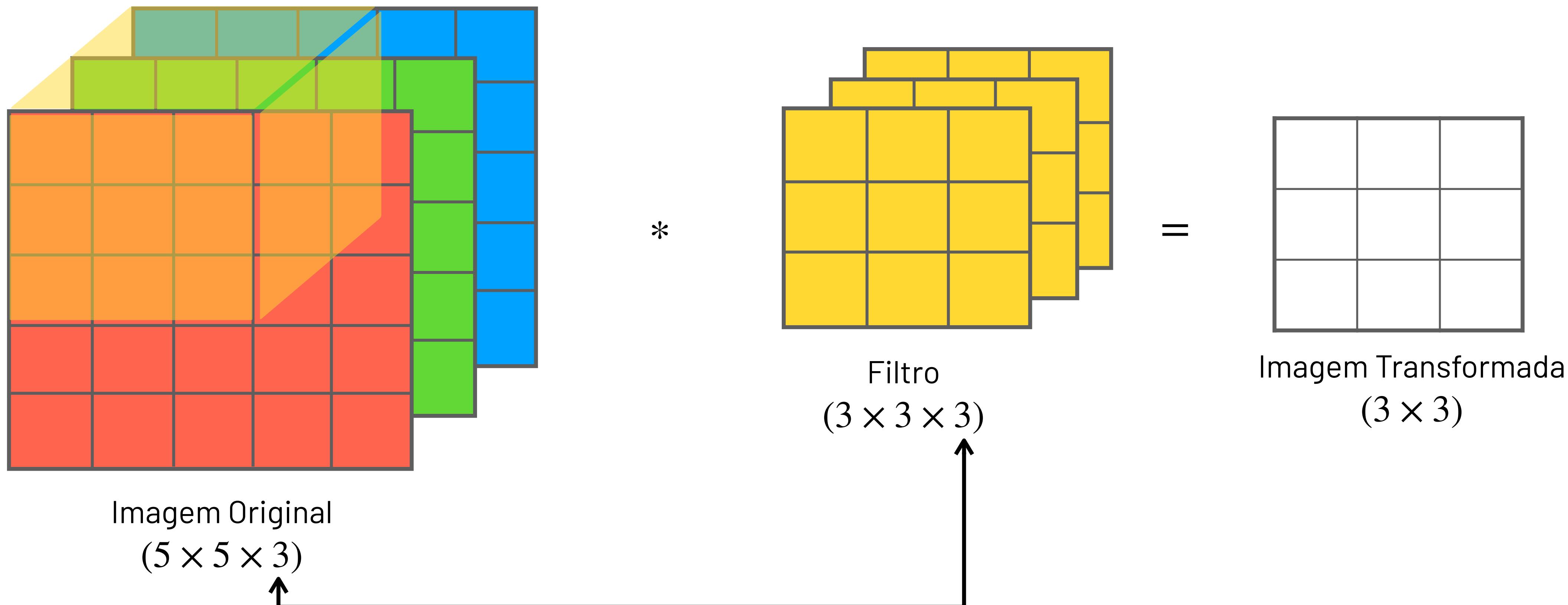
Convoluçãoes em imagens coloridas (R,G,B) necessitam filtros com 3 canais



O número de canais deve ser o mesmo na imagem e no filtro!

Convoluçãoes em Volumes

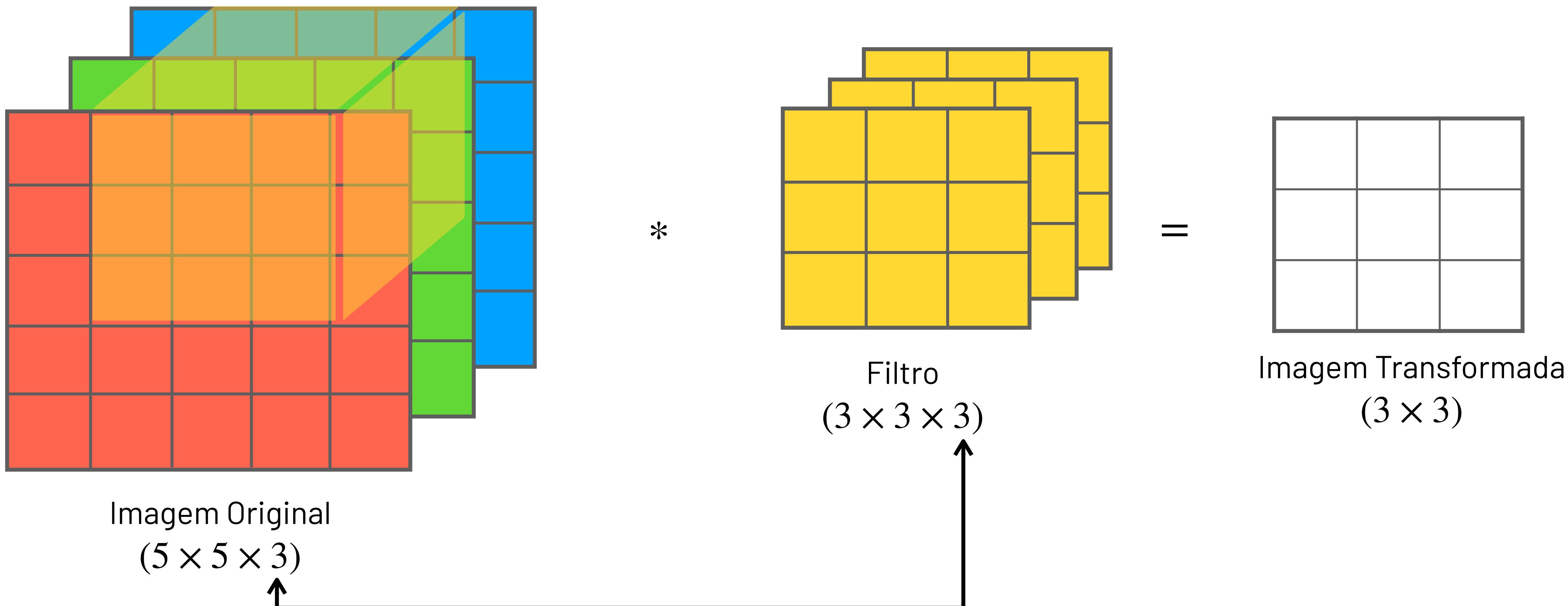
Convoluçãoes em imagens coloridas (R,G,B) necessitam filtros com 3 canais



O número de canais deve ser o mesmo na imagem e no filtro!

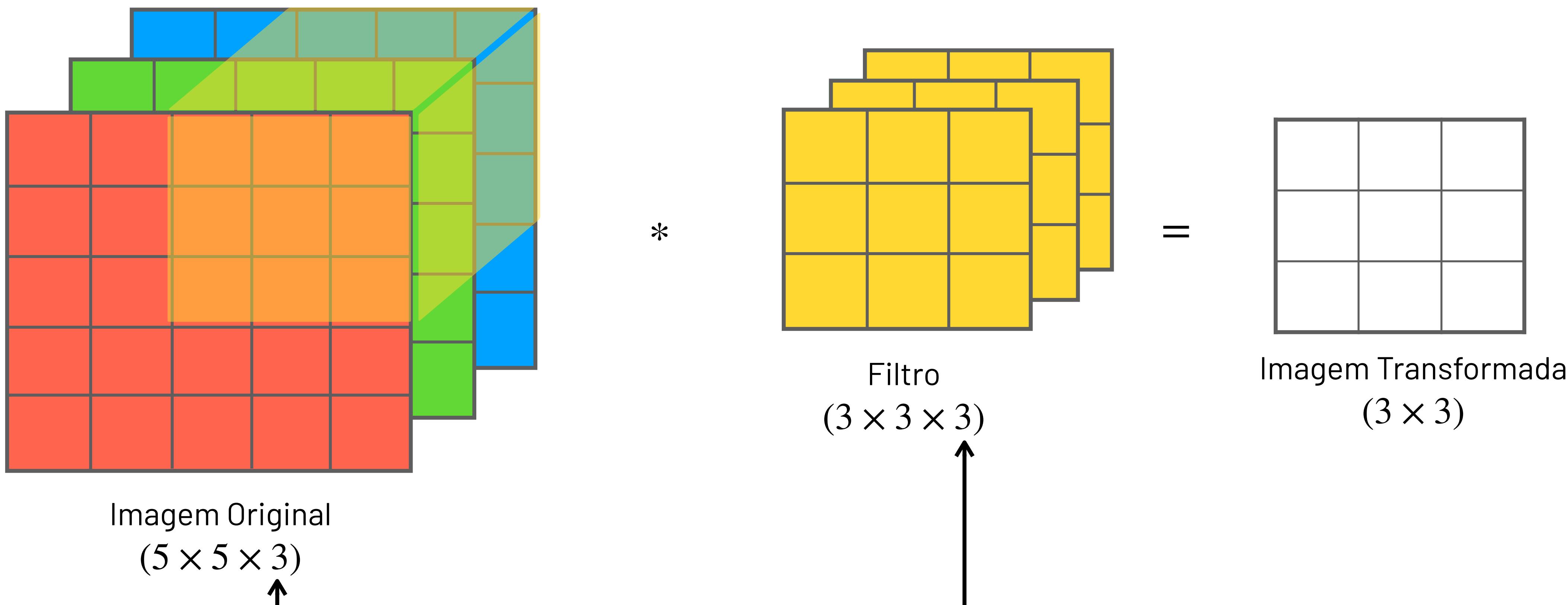
Convoluçãoes em Volumes

Convoluçãoes em imagens coloridas (R,G,B) necessitam filtros com 3 canais



Convoluçãoes em Volumes

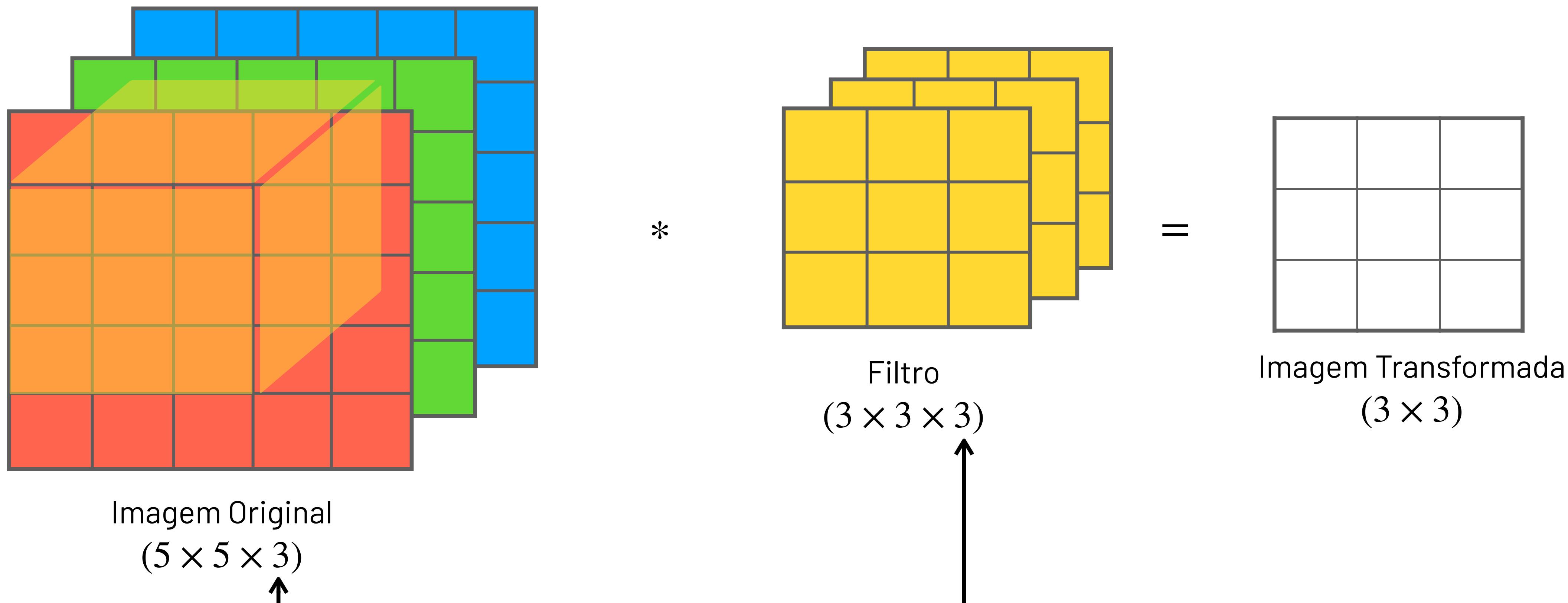
Convoluçãoes em imagens coloridas (R,G,B) necessitam filtros com 3 canais



O número de canais deve ser o mesmo na imagem e no filtro!

Convoluçãoes em Volumes

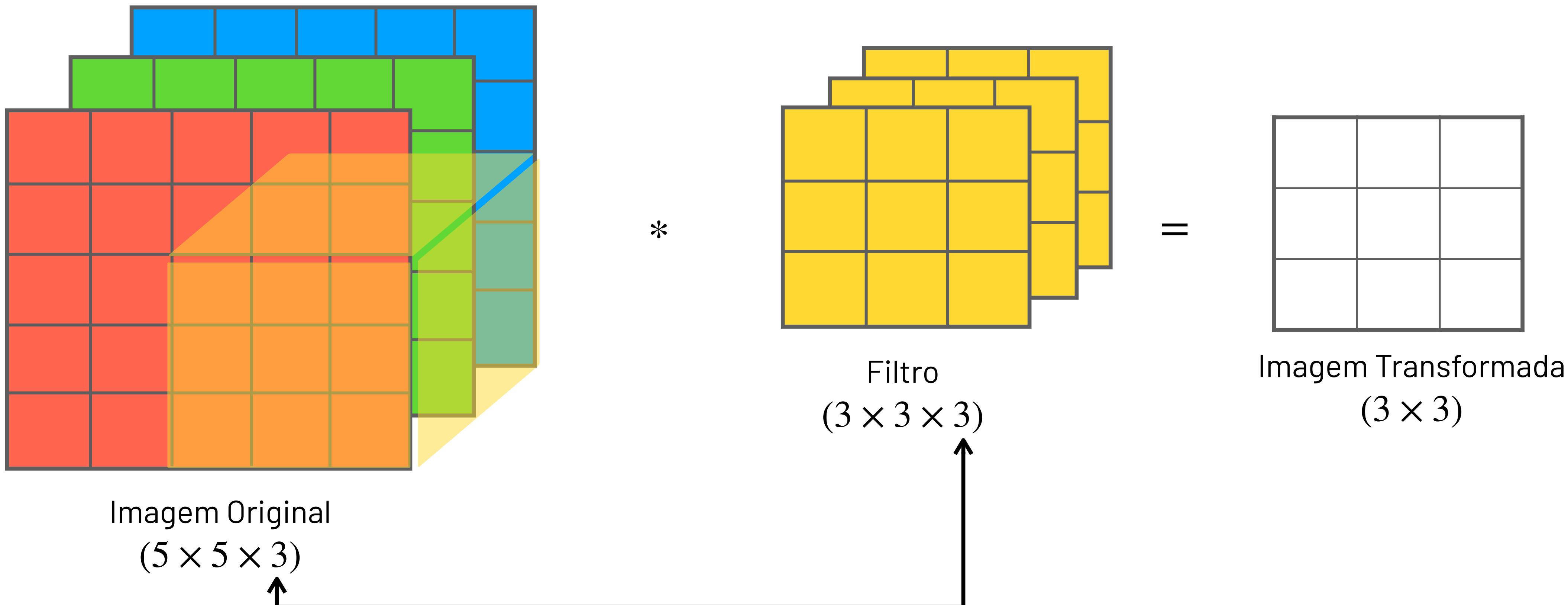
Convoluçãoes em imagens coloridas (R,G,B) necessitam filtros com 3 canais



O número de canais deve ser o mesmo na imagem e no filtro!

Convoluçãoes em Volumes

Convoluçãoes em imagens coloridas (R,G,B) necessitam filtros com 3 canais



O número de canais deve ser o mesmo na imagem e no filtro!

Múltiplos Filtros

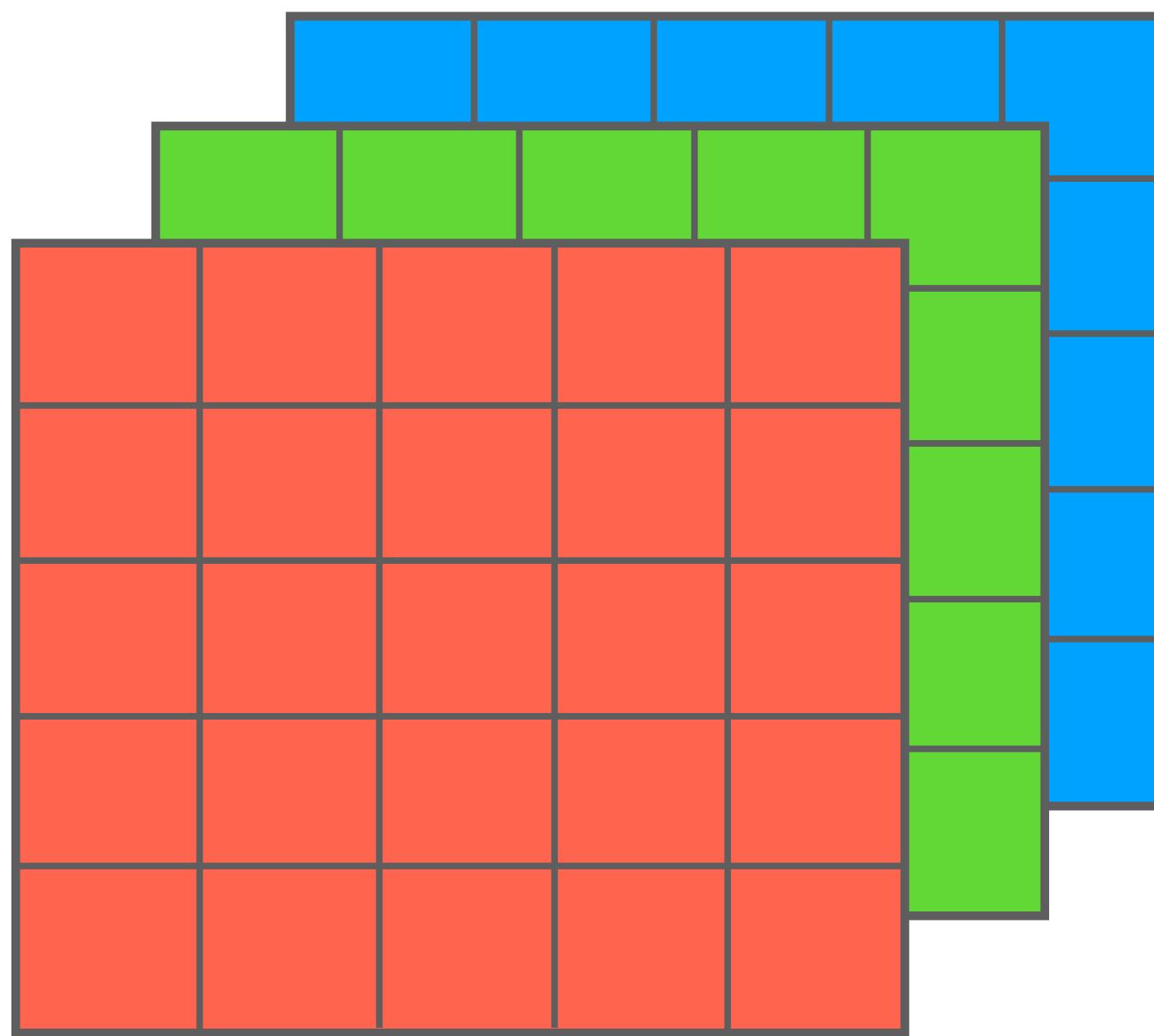
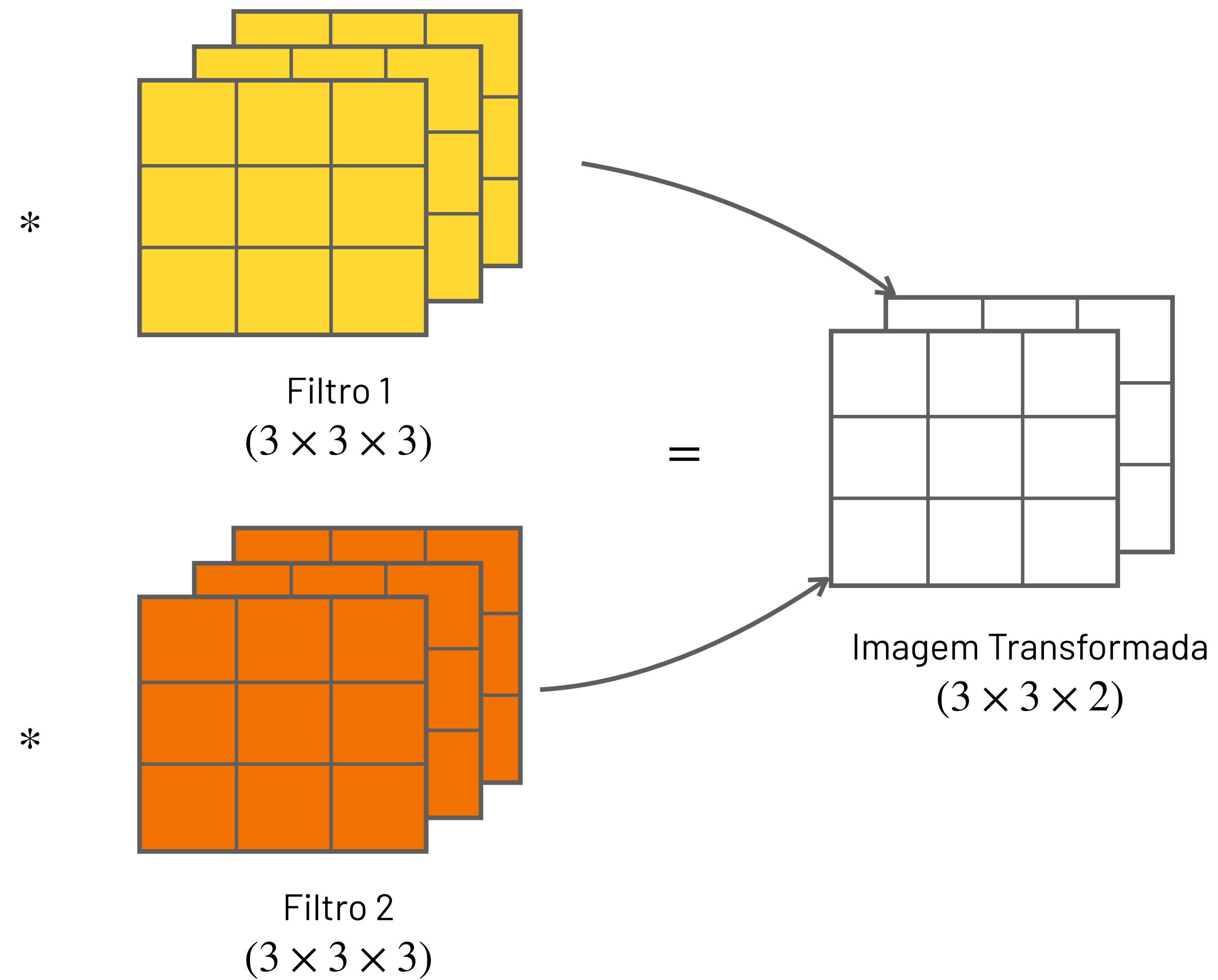
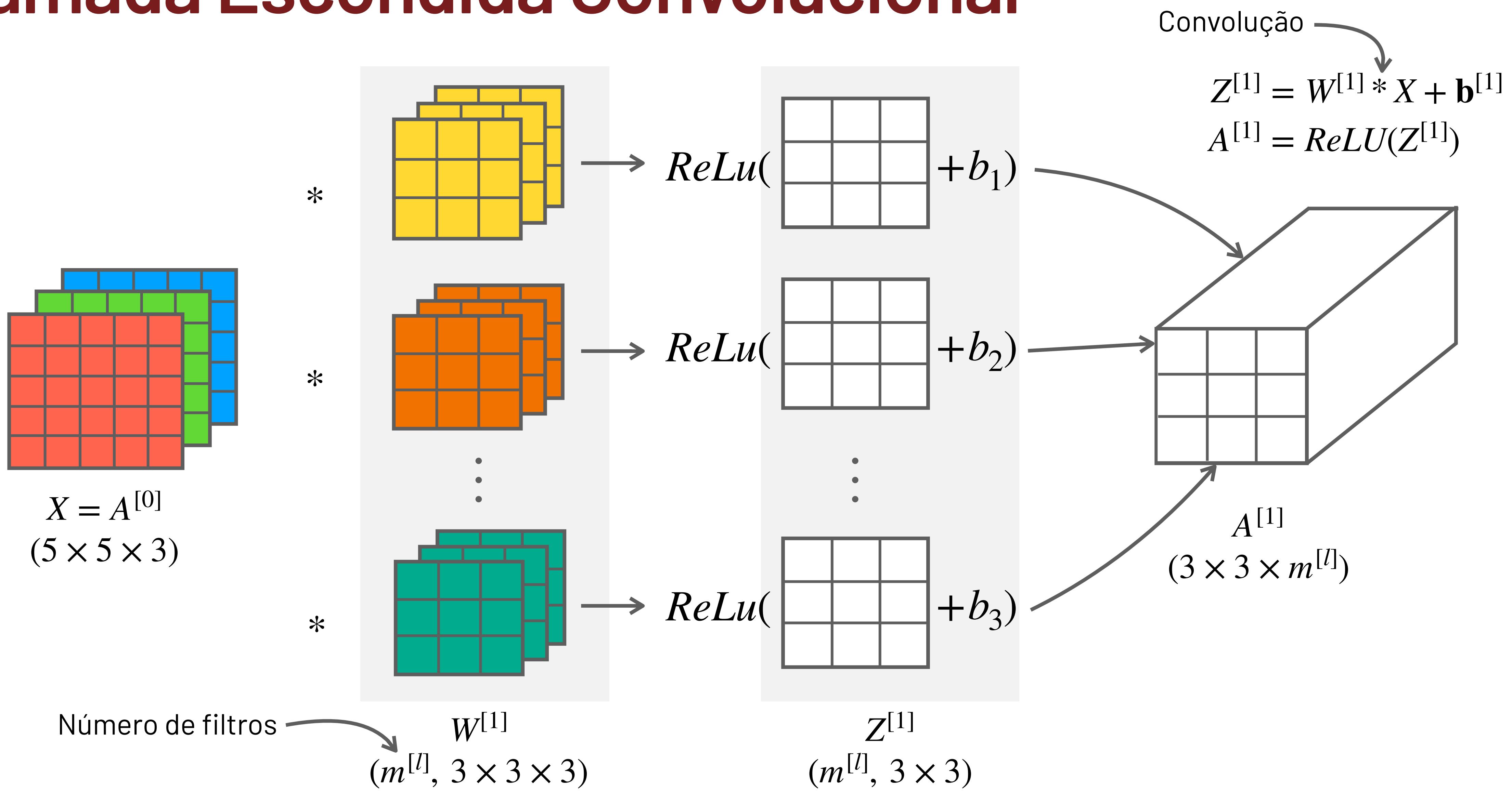


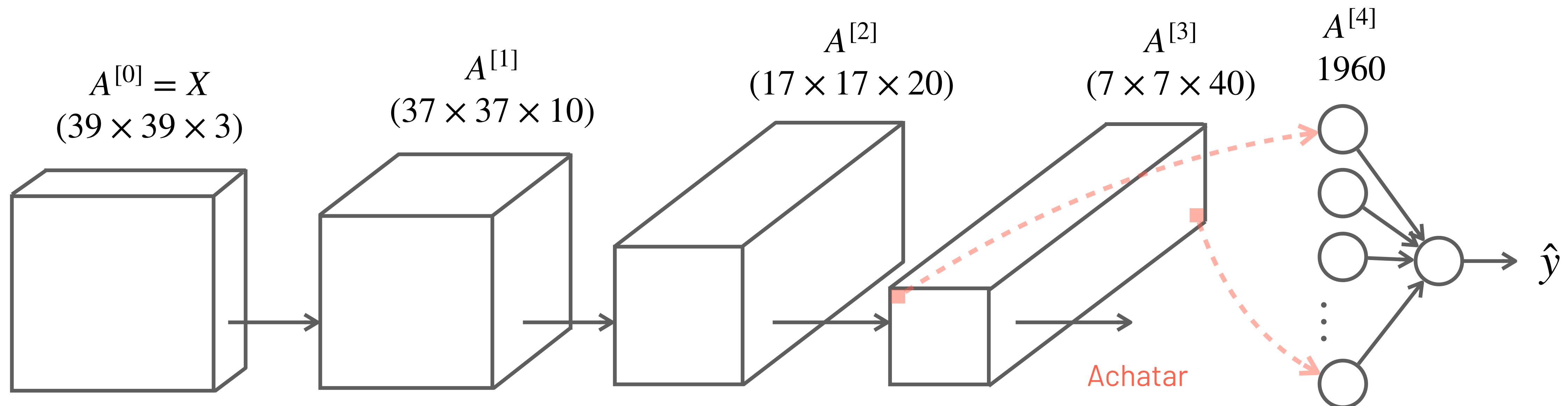
Imagen Original $(5 \times 5 \times 3)$



Camada Escondida Convolucional



Rede Neural Convolucional (CNN) de Classificação



$$\begin{array}{ll} n^{[0]} = 39 & f^{[1]} = 3 \\ s^{[1]} = 1 & \\ p^{[1]} = 0 & \\ m^{[1]} = 10 & \end{array}$$

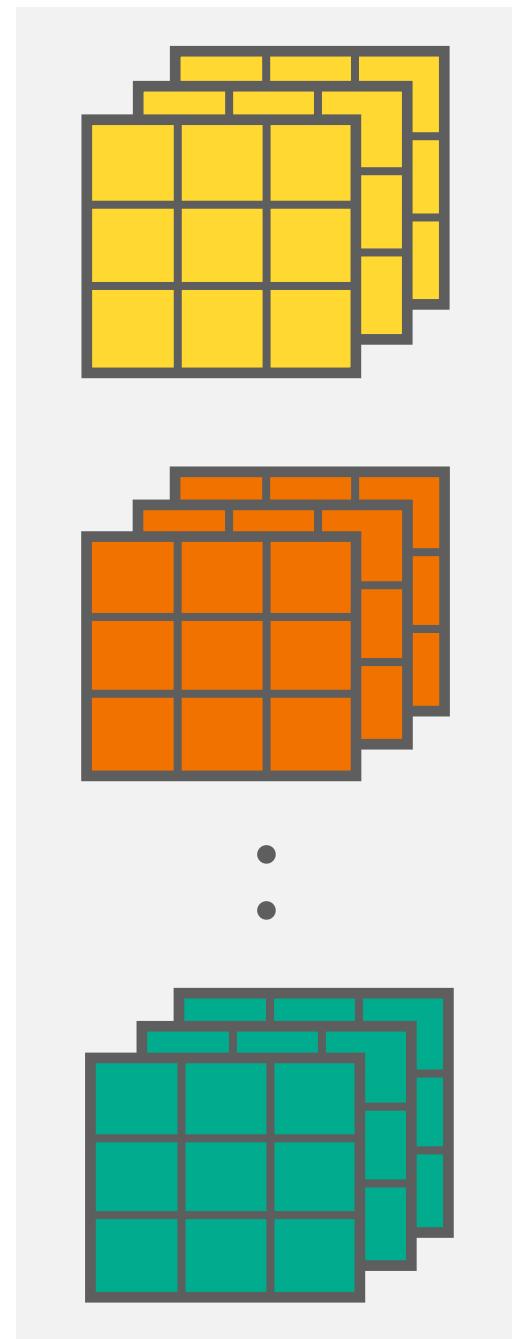
$$\begin{array}{ll} f^{[2]} = 5 & f^{[3]} = 5 \\ s^{[2]} = 2 & s^{[3]} = 2 \\ p^{[2]} = 0 & p^{[3]} = 0 \\ m^{[2]} = 20 & m^{[3]} = 40 \end{array}$$

Notação:

- $f^{[l]}$ tamanho dos filtros da camada l
- $s^{[l]}$ tamanho do stride da camada l
- $p^{[l]}$ tamanho do padding da camada l
- $m^{[l]}$ número de filtros na camada l

Número de parâmetros

Quantos parâmetros uma camada com 10 filtros ($3 \times 3 \times 3$) tem?



$W^{[1]}$
 $(10, 3 \times 3 \times 3)$

$$\begin{aligned} 3 \times 3 \times 3 &= 27 \\ &\quad + 1 \\ &= 28 \\ &\quad \times 10 \\ &= \underline{\underline{280}} \text{ parâmetros} \end{aligned}$$

Camadas de Pooling

Além das camadas convolucionais, CNNs tipicamente também utilizam **camadas de pooling** para extrair características de imagens:

- ▶ Max Pooling
- ▶ Average Pooling

Essas camadas realizam computações fixas e por isso não possuem pesos para aprender!

Max pooling

Filtro para extrair o elemento máximo da vizinhança.

$$\begin{matrix} & \begin{matrix} 1 & 3 \\ 2 & 9 \\ 1 & 3 \\ 5 & 6 \end{matrix} & \begin{matrix} 2 & 1 \\ 1 & 1 \\ 2 & 3 \\ 1 & 2 \end{matrix} \end{matrix} * \begin{matrix} \text{Max pooling} \\ f=2 \\ s=2 \end{matrix} = \begin{matrix} 9 & 2 \\ 6 & 3 \end{matrix} \quad (2 \times 2)$$

(4 × 4)

Sem pesos para aprender!

Average pooling

Filtro para extrair a média da vizinhança.

| | | | |
|---|---|---|---|
| 1 | 3 | 2 | 1 |
| 2 | 9 | 1 | 1 |
| 1 | 3 | 2 | 3 |
| 5 | 6 | 1 | 2 |

(4×4)

$$\begin{array}{c} * \\ \text{Average pooling} \\ f = 2 \\ s = 2 \end{array} = \begin{array}{|c|c|} \hline 3.75 & 1.25 \\ \hline 3.75 & 2 \\ \hline \end{array} \quad (2 \times 2)$$

Sem pesos para aprender!

Exercício

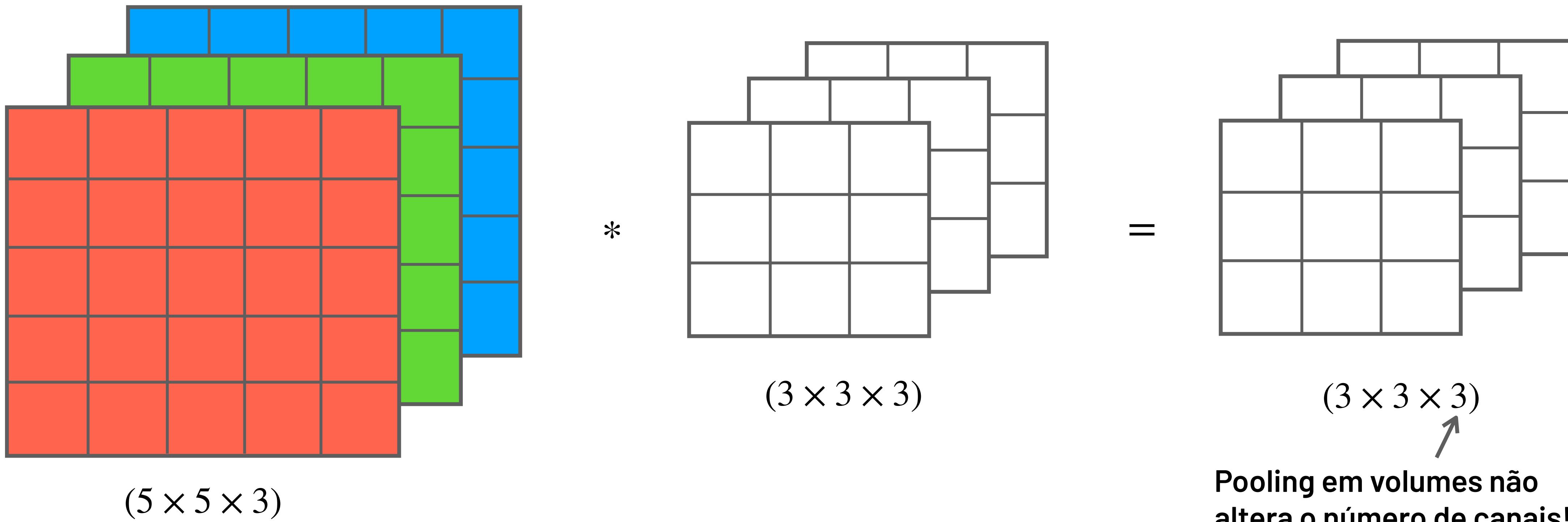
Calcule a matriz resultante da aplicação do filtro max pool com $f = 3$ e $s = 1$

| | | | | |
|---|---|---|---|---|
| 1 | 3 | 2 | 1 | 3 |
| 2 | 9 | 1 | 1 | 5 |
| 1 | 3 | 2 | 3 | 2 |
| 8 | 3 | 5 | 1 | 0 |
| 5 | 6 | 1 | 2 | 9 |

(5×5)

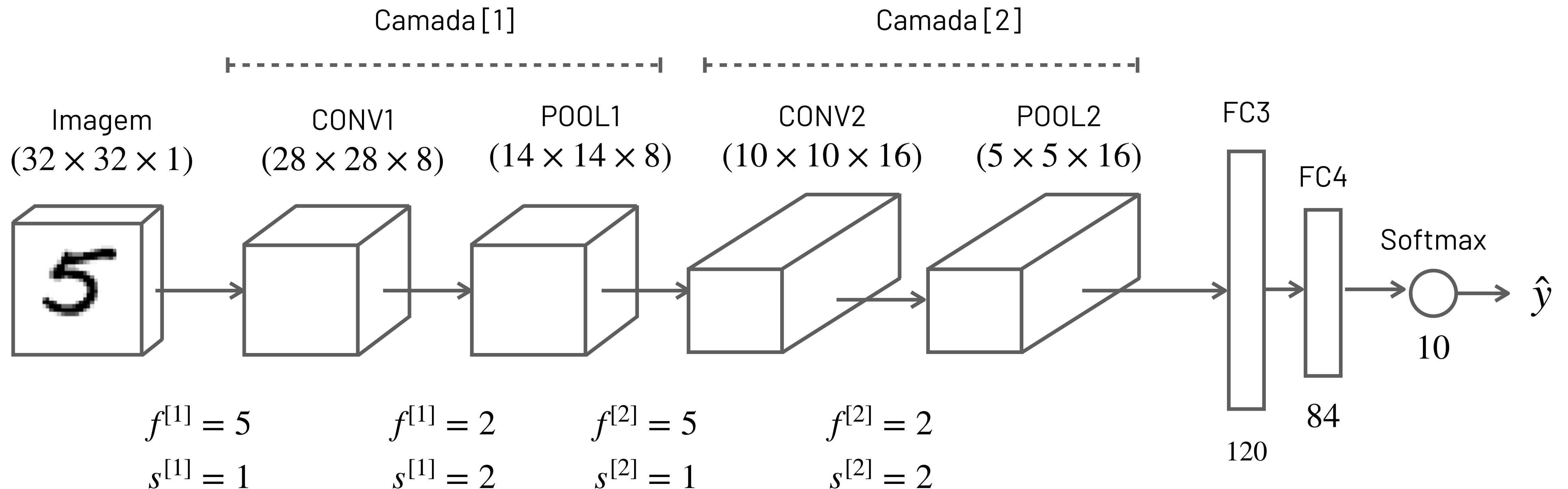
Camadas de Pooling em Volumes

Os filtros de pooling são aplicados de forma independente para cada canal



CNN com Camadas de Pooling

LeNet-5



CNN com Camadas de Pooling

| | Dimensões da Ativação | Tamanho da Ativação | Número de Parâmetros |
|------------------------|-----------------------|---------------------|----------------------|
| | (32, 32, 3) | 3072 | 0 |
| CONV1(f=5, s=1) | (28, 28, 8) | 6272 | 608 |
| POOL1 | (14, 14, 8) | 1568 | 0 |
| CONV2(f=5, s=1) | (10, 10, 16) | 1600 | 3216 |
| POOL2 | (5, 5, 16) | 400 | 0 |
| FC3 | (120, 1) | 120 | 48120 |
| FC4 | (84, 1) | 84 | 10164 |
| Softmax | (10, 1) | 10 | 850 |

Porque Convoluçãoes?

Redução do número de parâmetros

| | | | | | |
|----|----|----|---|---|---|
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |

$$\begin{matrix} & * & \\ \begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix} & = & \begin{matrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{matrix} \end{matrix}$$

Detecção de
Borda Vertical

| | | | |
|---|----|----|---|
| 0 | 30 | 30 | 0 |
| 0 | 30 | 30 | 0 |
| 0 | 30 | 30 | 0 |
| 0 | 30 | 30 | 0 |

Compartilhamento de parâmetros

Um filtro para detecção de características (e.g., detector de bordas) que funciona em uma parte da imagem, provavelmente funciona em outra parte da imagem.

Conexões Esparsas

Em cada camada, cada saída depende de apenas um número pequenos de entradas

Próxima aula

A12: Estudo de Casos de CNNs

Resnet, Inception Network, MobileNet e Efficient Net.