

# ALGORITMOS GENÉTICOS E REDES NEURAIS

## REDES NEURAIS CONVOLUCIONAIS - CNN

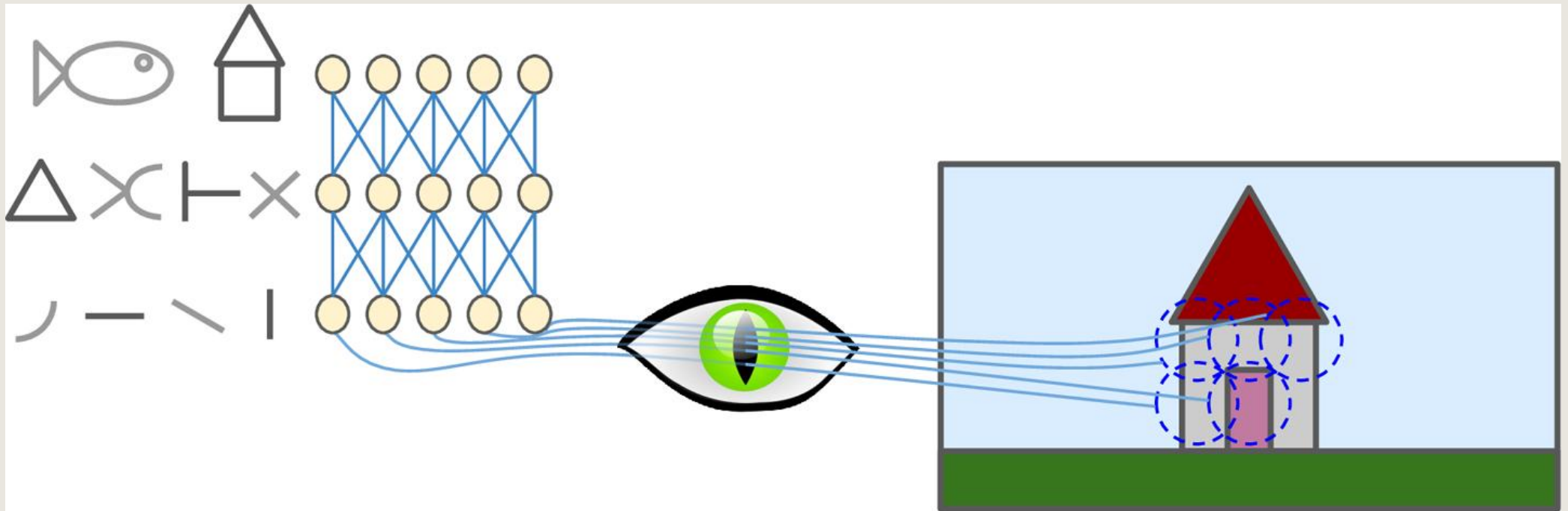
Prof. Flávio Belizário da Silva Mota  
Universidade do Vale do Sapucaí – UNIVAS  
Sistemas de Informação

# A ARQUITETURA DO CÓRTEX VISUAL

- Entre 1958 e 1959, David H. Hubel e Torsten Wiesel estudaram como a visão funcionava, fornecendo informações cruciais sobre a estrutura do córtex visual.
- Eles mostraram que muitos neurônios no córtex visual têm um **pequeno campo receptivo local**, ou seja, reagem apenas a estímulos visuais localizados em uma **região limitada do campo visual**. Os campos receptivos de diferentes neurônios podem se sobrepor e, juntos, preenchem todo o campo visual.
- Alguns neurônios reagem apenas a imagens **de linhas horizontais**, enquanto outros reagem apenas a linhas com diferentes orientações (dois neurônios podem ter o mesmo campo receptivo, mas reagir a diferentes orientações de linha).

# A ARQUITETURA DO CÓRTEX VISUAL

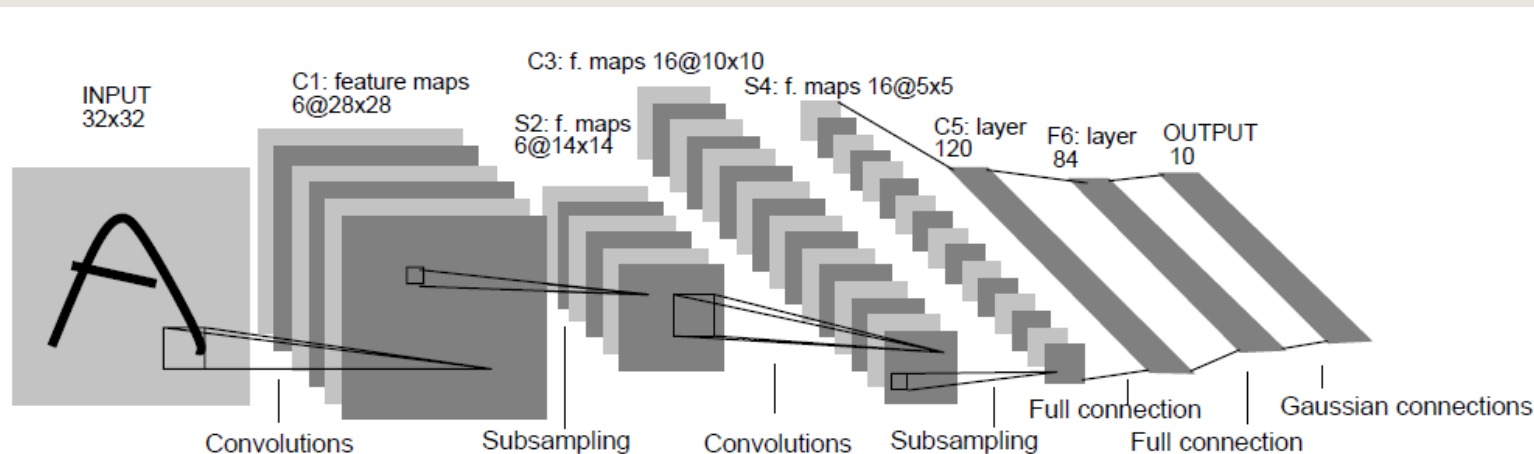
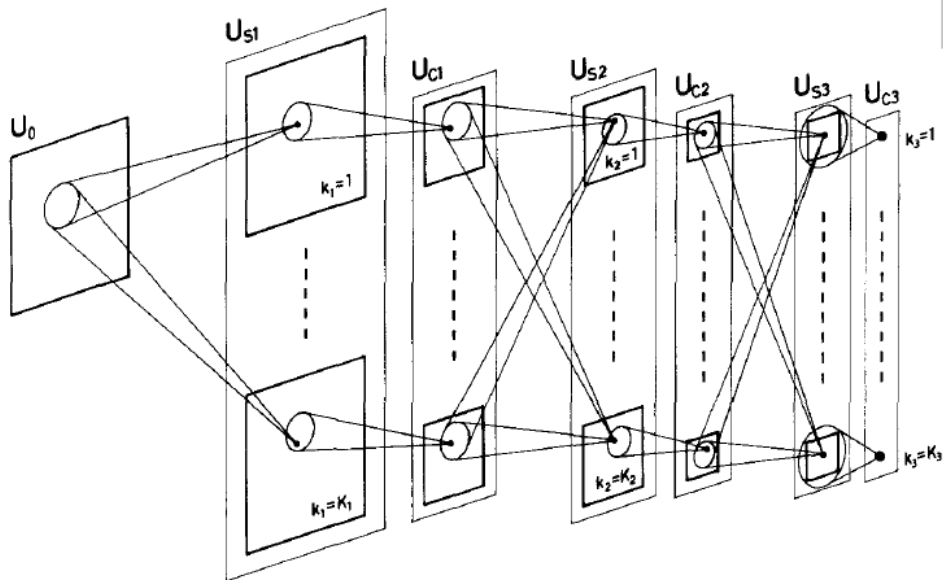
- Eles também notaram que alguns neurônios têm campos receptivos maiores e reagem a padrões mais complexos que são combinações de padrões de nível inferior. Essas observações levaram à ideia de que os **neurônios de nível superior são baseados nas saídas dos neurônios vizinhos de nível inferior.**



# REDES NEURAIS CONVOLUCIONAIS (CNN)

- Inspirados pela ideia de como o córtex visual funciona, em 1980 Kunihiro Fukushima cria o Neocognitron, a primeira rede neural convolucional, e em 1998 Yann LeCun introduz a LeNet-5.

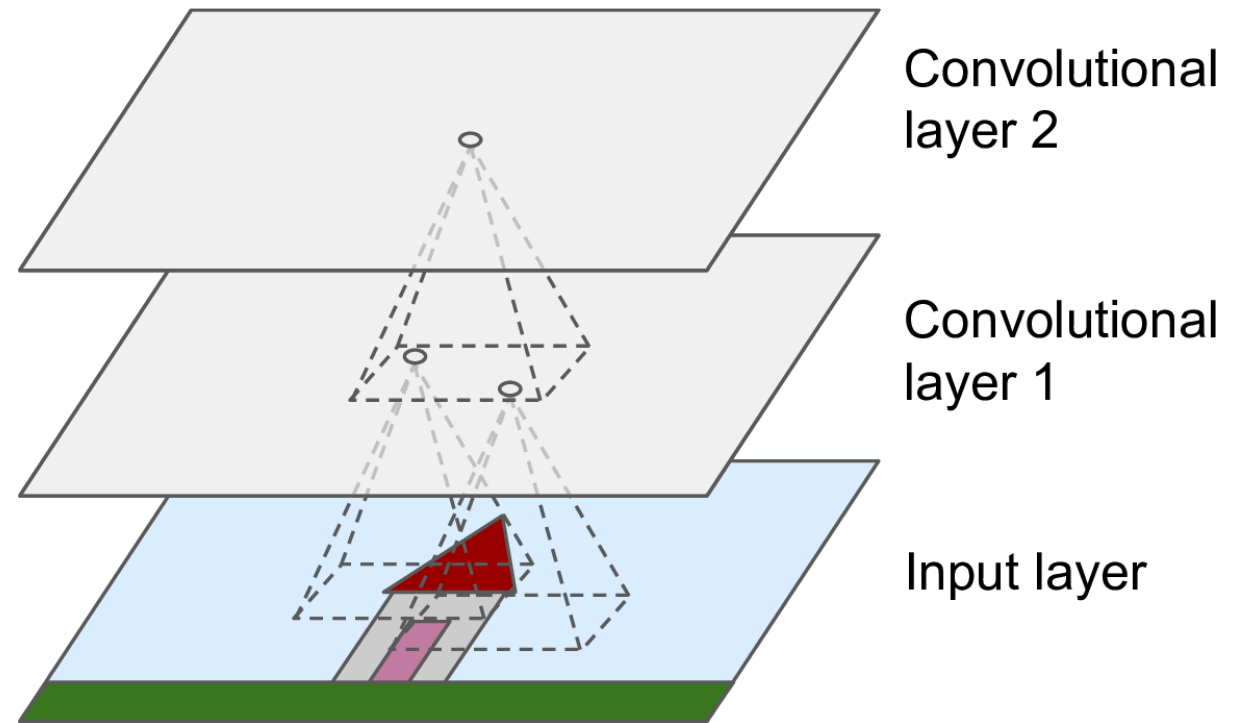
Neocognitron



LeNet-5

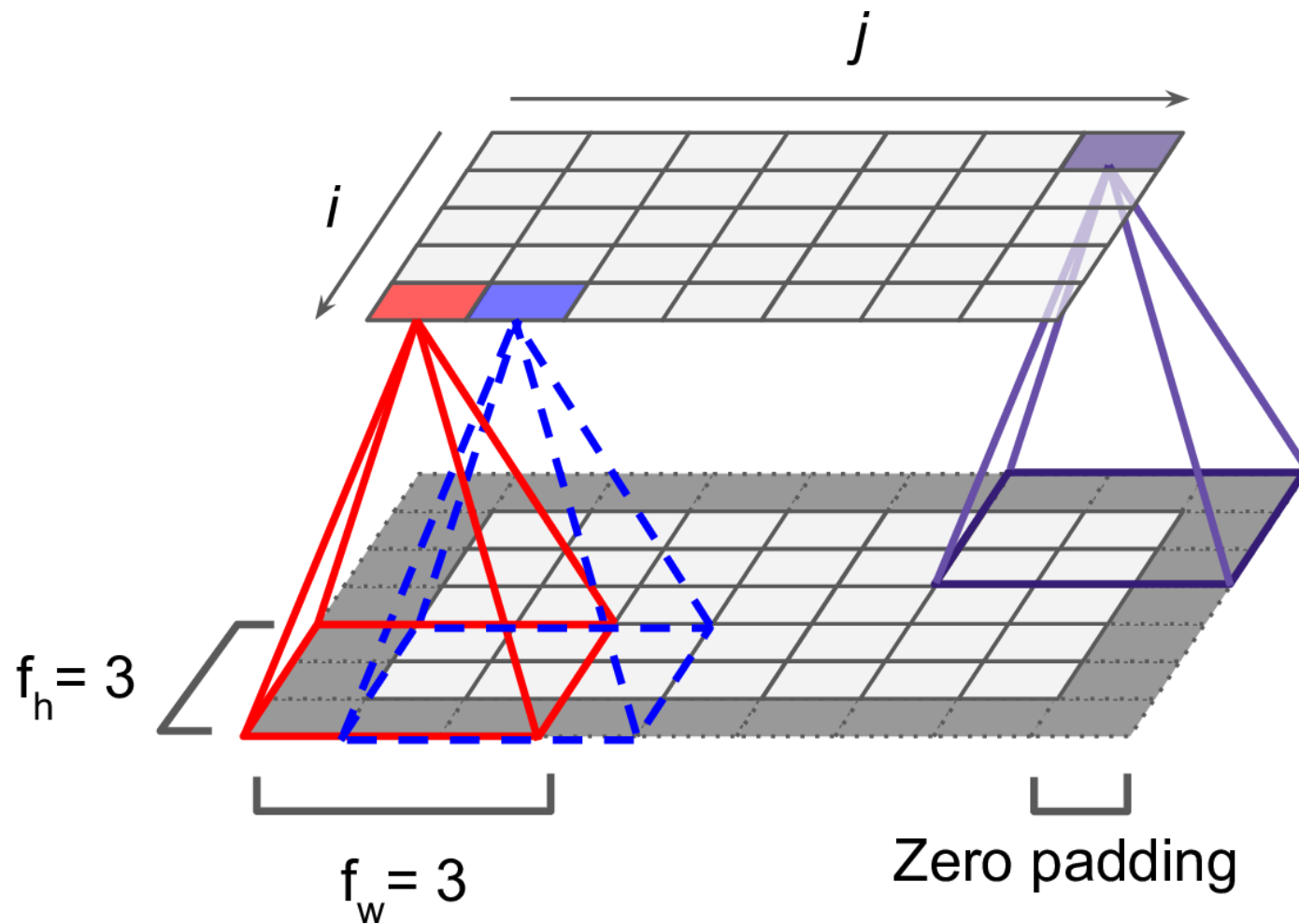
# CAMADAS CONVOLUCIONAIS

- O bloco básico de uma CNN é a camada convolucional.
- Os neurônios da primeira camada convolucional não estão conectados a todos os pixels da imagem de entrada, mas apenas aos pixels em seus campos receptivos.
- Por sua vez, cada neurônio da segunda camada convolucional está conectado apenas aos neurônios localizados dentro de um pequeno retângulo na primeira camada.
- Essa arquitetura permite que a rede se concentre em pequenas características de baixo nível na primeira camada oculta, para então agrupá-las em características maiores e de nível superior na camada oculta seguinte, e assim por diante.



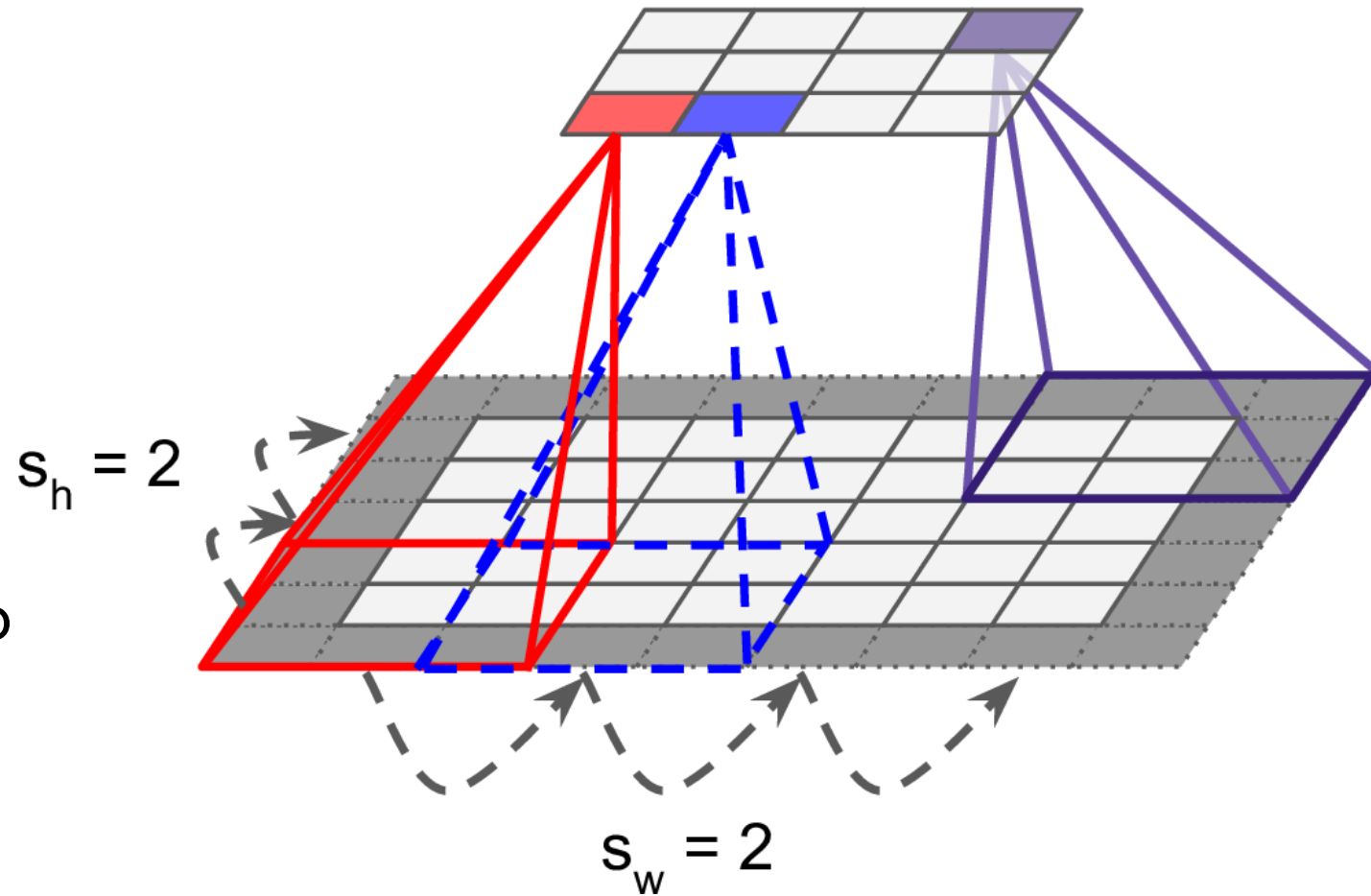
# CAMADAS CONVOLUCIONAIS

- Um neurônio localizado na linha  $i$ , coluna  $j$  de uma determinada camada está conectado às saídas dos neurônios da camada anterior, localizados nas linhas  $i$  a  $i + f_h - 1$ , colunas  $j$  a  $j + f_w - 1$ , onde  $f_h$  e  $f_w$  são a **altura** e a **largura do campo receptivo**.
- Para que uma camada tenha a mesma altura e largura que a camada anterior, é comum adicionar zeros em torno das entradas, como mostrado no diagrama. Isso é chamado de preenchimento com zeros (**zero padding**).



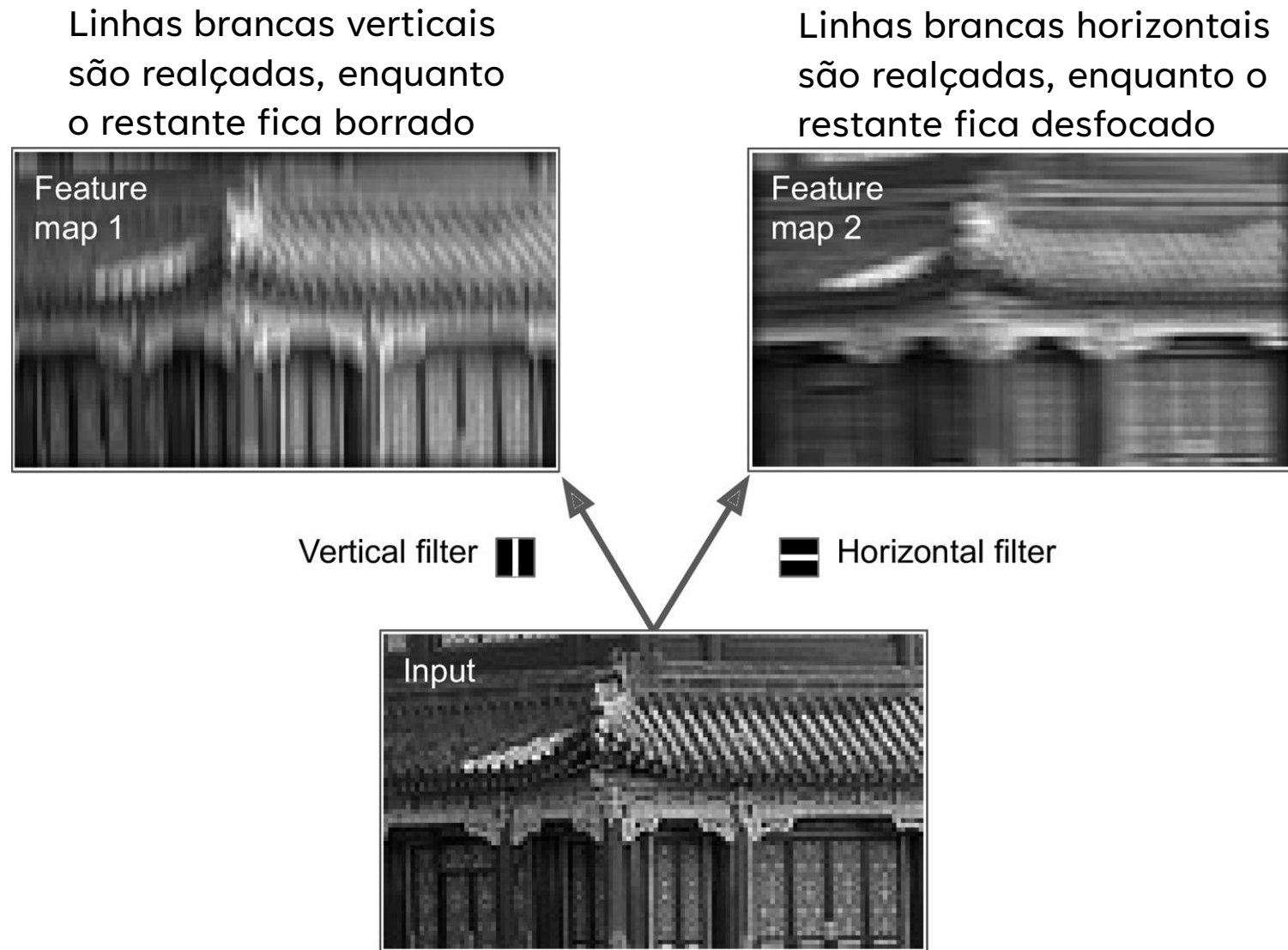
# CAMADAS CONVOLUCIONAIS

- Também é possível conectar uma camada de entrada maior a uma camada muito menor **espaçando os campos receptivos**.
- Isso reduz drasticamente a complexidade computacional do modelo. A mudança de um campo receptivo para o próximo é chamada de passo (**stride**).



# CAMADAS CONVOLUCIONAIS

- Os pesos dos neurônios nas camadas convolucionais são **matrizes do tamanho do campo receptivo**. Esses pesos são chamados de **filtros ou kernels de convolução**.
- Os filtros (pesos) são aplicados nos pixels da imagem, produzindo uma nova imagem na camada seguinte que vai ter uma dimensão menor.
- A ideia é extrair características da imagem, como bordas, formatos, sombras, etc.





# CAMADAS CONVOLUCIONAIS

- Na figura anterior, os dois filtros aplicados são como as matrizes abaixo.
- No **filtro vertical**, os neurônios que usam esses pesos **ignorarão tudo em seu campo receptivo, exceto a linha vertical central** (já que todas as entradas serão multiplicadas por zero, exceto as localizadas na linha vertical central).
- No filtro horizontal, os neurônios que usam esses pesos ignorarão tudo em seu campo receptivo, **exceto a linha horizontal central**.

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

Filtro vertical

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

Filtro horizontal

# CAMADAS CONVOLUCIONAIS

- A aplicação dos pesos geram **mapas de características**, que destacam áreas da imagem que mais ativam o filtro.
- Obviamente, **não é necessário definir os filtros manualmente**: durante o treinamento, a **camada convolucional aprenderá automaticamente os filtros mais úteis para sua tarefa**, e as camadas superiores aprenderão a combiná-los em padrões mais complexos.
- Matematicamente, **os pesos são multiplicados pelos pixels e o resultado de cada multiplicação é somado para produzir o valor da próxima camada**.
- Isso é feito sucessivamente entre camadas.

**X**

0	0	0	0	0
0	2	1	2	0
0	5	0	1	0
0	1	7	3	0
0	0	0	0	0

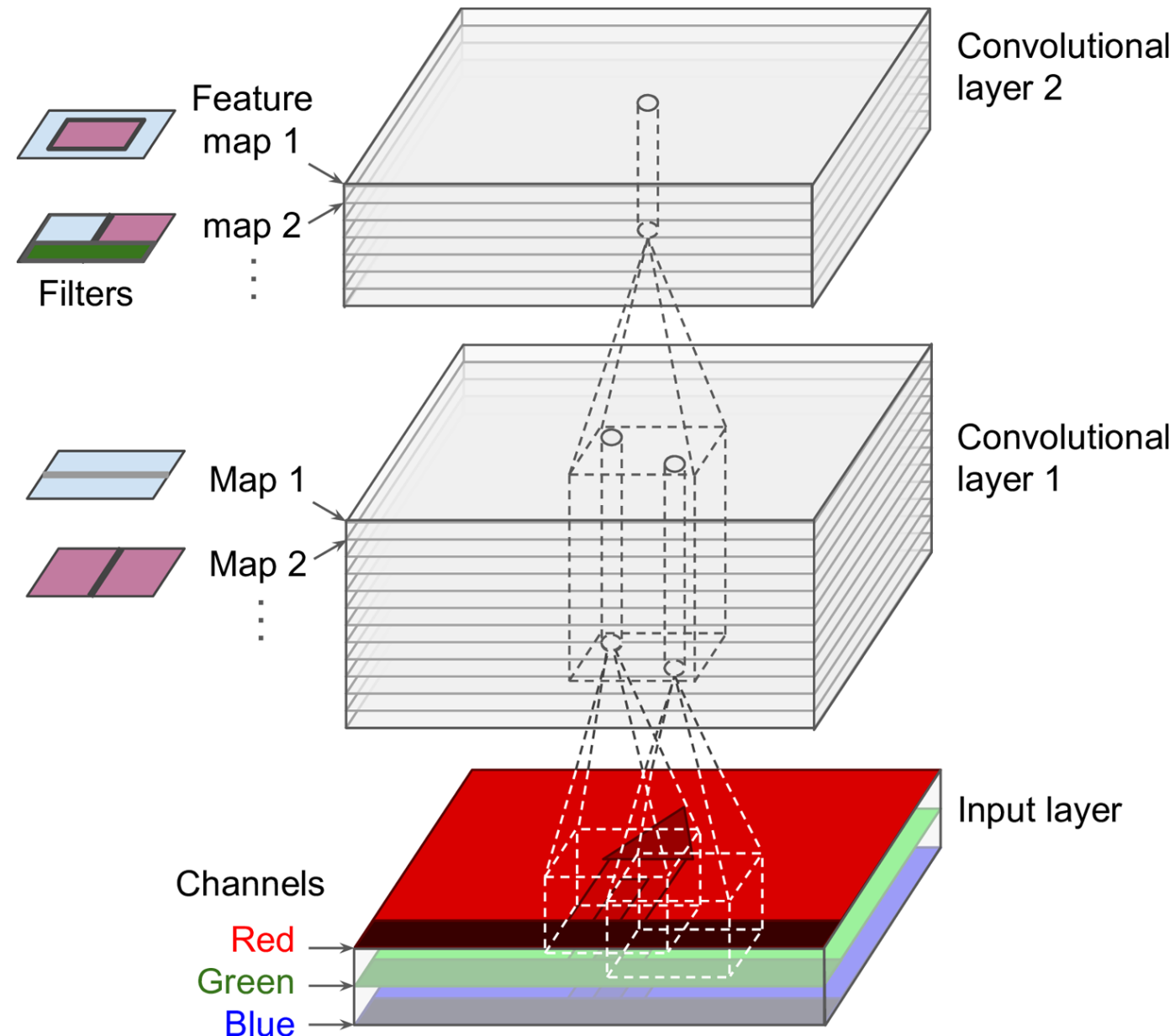
**W**

0.5	0.7	0.4
0.3	0.4	0.1
0.5	1	0.5

\*

# CAMADAS CONVOLUCIONAIS

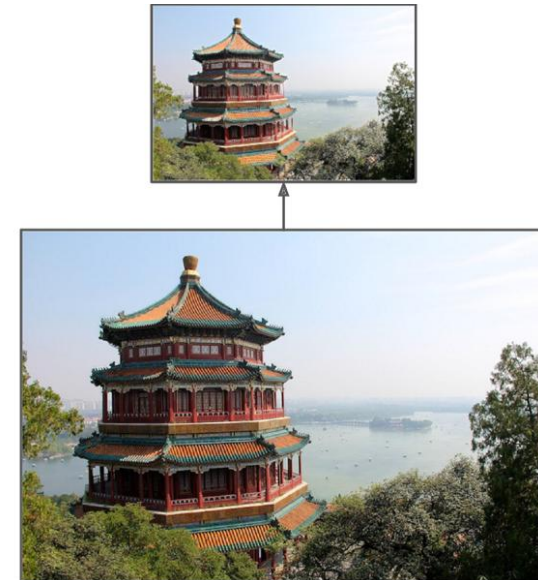
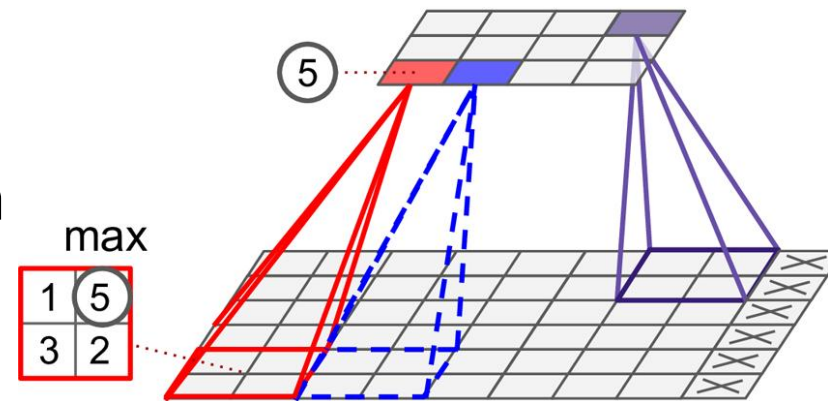
- Uma camada convolucional possui múltiplos filtros (você decide quantos) e gera um mapa de características por filtro, sendo mais precisamente representada em 3D.
- Todos os neurônios dentro de um determinado mapa de características compartilham os pesos. Neurônios em diferentes mapas de características usam pesos diferentes.
- Em resumo, uma camada convolucional aplica simultaneamente múltiplos filtros treináveis às suas entradas, tornando-a capaz de detectar múltiplas características em qualquer ponto de suas entradas.



# CAMADAS DE POOLING

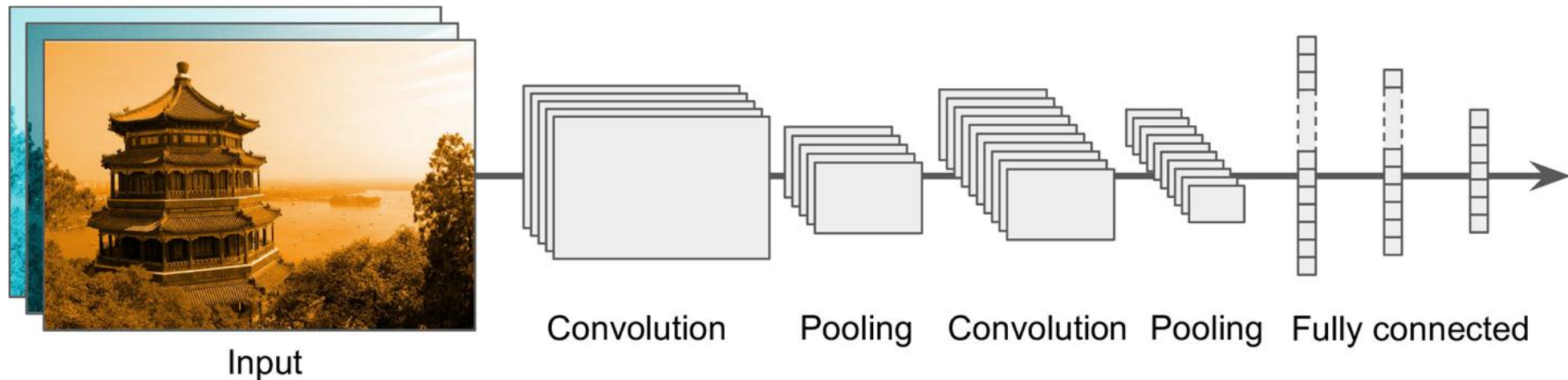
- Essas camadas tem como objetivo reduzir a imagem de entrada para diminuir a carga computacional, o uso de memória e o número de parâmetros, limitando, assim, o risco de sobreajuste.
- São configuradas exatamente como as camadas convolucionais, mas ela não tem pesos ajustáveis, apenas uma função de agragação, como máximo ou média.

Uma camada de max pooling, que é o tipo mais comum de camada de pooling, com kernel 2x2. Os valores de entrada são 1, 5, 3 e 2, portanto, apenas o valor máximo, 5, é propagado para a próxima camada.



# ARQUITETURA DA CNN

- As arquiteturas típicas de CNN **empilham algumas camadas convolucionais**, cada uma geralmente seguida por uma camada ReLU. Depois uma **camada de pooling**, depois mais algumas camadas convolucionais (+ReLU), depois outra camada de pooling e assim por diante.
- A **imagem fica cada vez menor à medida que avança pela rede**, mas também normalmente fica cada vez mais profunda (ou seja, com **mais mapas de características**), graças às camadas convolucionais.
- No **fim da rede**, adiciona-se uma **rede neural MLP**, composta por algumas camadas totalmente conectadas (+ReLUs). Essa camada final gera a previsão (por exemplo, uma camada softmax que gera probabilidades estimadas de classe).



# CNN EXPLAINER

- Antes de implementar uma CNN no TensorFlow, vamos explorar um pouco mais seu funcionamento através do simulador online CNN Explainer.
- Essa página é fruto do artigo “**CNN Explainer: Learning Convolutional Neural Networks with Interactive Visualization**” e tem como objetivo demonstrar, camada por camada, o que acontece dentro de uma rede CNN.
- <https://poloclub.github.io/cnn-explainer/>

