Inteligência Artificial

Raoni F. S. Teixeira

Aula 12 - Redes Convolucionais

1 Introdução

Redes Neurais Convolucionais (CNNs) possuem a mesma estrutura das redes neurais tradicionais. Elas usam neurônios com pesos ajustáveis, combinam operações lineares com ativações não lineares e seu treinamento depende de uma função de custo diferenciável.

A diferença é que as CNNs são construídas para processar sinais — dados estruturados *espacialmente* (e.g. imagem ou matriz 2D) ou *no tempo* (e.g. aúdio ou vetor). Como trabalham com matrizes ou vetores, as CNNs processam os sinais, explorando três princípios para maior eficiência:

- **Conectividade local**: Cada neurônio responde apenas a uma pequena região da entrada, reduzindo a complexidade.
- **Compartilhamento de pesos**: Os mesmos filtros são aplicados em diferentes posições, detectando padrões de forma consistente.
- **Invariância translacional**: Padrões são reconhecidos independentemente de sua posição no sinal.

2 Filtros

Um **filtro** é uma função que modifica um sinal, preservando componentes específicas (como certas frequências) e atenuando outras. O resultado é um sinal simplificado e mais limpo.

Convolução 1D

A Figura 1 ilustra a ação do filtro $F = (-1, +1)$ em um sinal binário unidimensional. O filtro (ou **núcleo convolucional**) desliza pelo sinal com um **passo (stride)** de 1, calculando o produto escalar em cada posição. O *stride* define o deslocamento do filtro a cada operação.

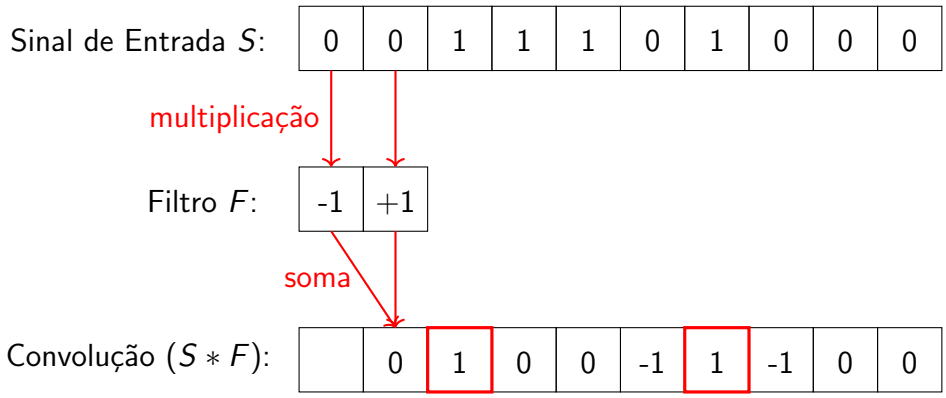

Figura 1: Convolução 1D para detectar as bordas esquerdas (células em destaque) de um sinal binário.

Para um sinal discreto S e um filtro F de tamanho M e stride s, a convolução unidimensional é definida por:

$$(S * F)[n] = \sum_{m=1}^{M} S[n \cdot s - m]F[m] \tag{1}$$

Convolução 2D

A Figura 2 mostra um filtro 3×3 aplicado a um sinal binário 2D S em que as células pretas correspondem ao valor 1 e as brancas ao valor 0. O filtro compara cada região de S, procurando padrões horizontais de tamanho três. O *stride* igual à 1 faz com que o filtro deslize por toda imagem.

A fórmula para convolução 2D com filtro $K \times K$ e *stride* s é:

$$(S * F)[i, j] = \sum_{m=0}^{K-1} \sum_{n=0}^{K-1} S[i \cdot s + m, j \cdot s + n] \cdot F[m, n] \tag{2}$$

Aplicando essa fórmula, o processamento nas bordas é incompleto. Na posição $(0, 0)$, por exemplo, um filtro 3×3 só opera sobre 4 células, faltando 5 vizinhos para completar o cálculo.

A solução é adicionar uma borda de zeros ao redor da imagem original — técnica conhecida como **padding**. Esse ajuste garante três benefícios. Primeiro, equaliza o processamento de todos os pixels, incluindo os das bordas. Segundo, permite controlar o tamanho da saída, mantendo as dimensões originais quando necessário. Terceiro, preserva informações espaciais.

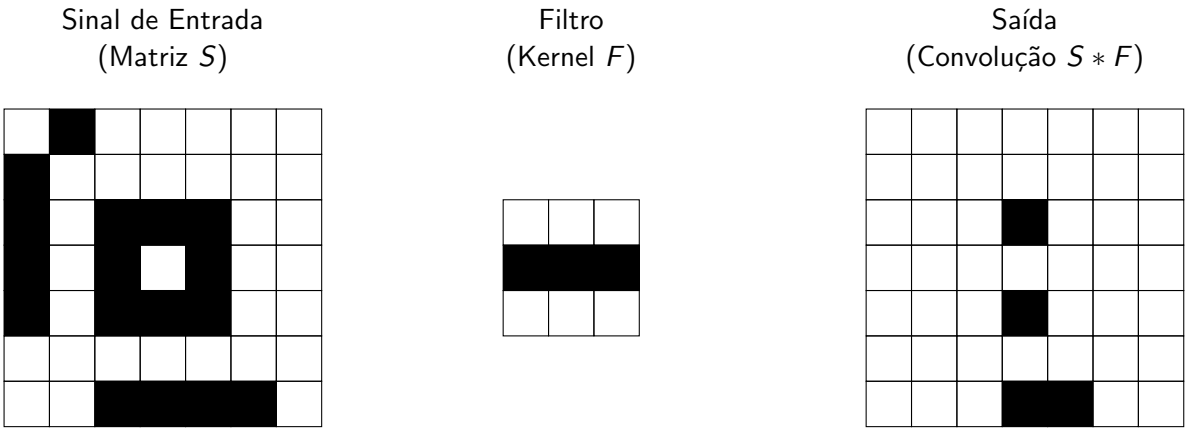

Figura 2: Convolução 2D com filtro 3×3 e *stride* 1 que detecta padrões horizontais no sinal S.

3 Agrupamento (Pooling)

O agrupamento (pooling) é um filtro sem pesos que reduz a dimensionalidade preservando características do sinal. Um parâmetro é novamente o stride que indica os deslocamentos na aplicação do filtro. Quando stride é igual a dois, por exemplo, o filtro pula uma célula em cada dimensão.

A Figura 3 ilustra os dois tipos mais comuns de agrupamento a seguir para uma matriz de entrada e filtros 2×2 com stride 2:

1. **Agrupamento Máximo (Max Pooling)**: que devolve o valor máximo em cada janela do filtro e cujo principal benefício é aumentar a invariância a pequenas translações (o padrão é reconhecido independentemente da posição exata no sinal).

2. **Agrupamento Médio (Average Pooling)**: que calcula a média dos valores na janela e é usado para suavizar o sinal.

4 Arquitetura de uma CNN

A Figura 4 mostra a arquitetura típica de uma Rede Neural Convolucional (CNN) para classificação de imagens, organizada em dois estágios principais:

1. **Extração hierárquica de características**:

    - Sequência de Blocos repetidos de:

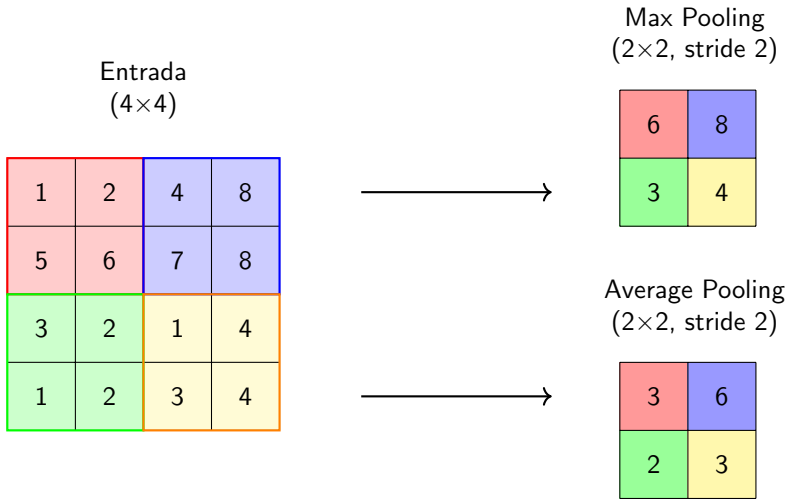

Figura 3: Operações de pooling usando filtro 2×2. As cores indicam janelas distintas. O Max Pooling extrai os valores máximos (6, 8, 3, 4) e Average Pooling as médias (3, 6, 2, 3) em cada janela.

  - Camada convolucional + ReLU (retificação)
  - Camada de *max pooling* (opcional)

2. **Classificação**:
   - Camadas totalmente conectadas no final
   - Função de ativação final (ex: Softmax para classificação)

As camadas convolucionais processam volumes 3D de entrada ($L_{\text{in}} \times A_{\text{in}} \times D_{\text{in}}$) através de bancos de filtros, gerando saídas com dimensões calculadas por:

$$\begin{aligned} L_{\text{out}} &= \left\lfloor \frac{L_{\text{in}} - F + 2P}{S} \right\rfloor + 1, \\ A_{\text{out}} &= \left\lfloor \frac{A_{\text{in}} - F + 2P}{S} \right\rfloor + 1, \\ D_{\text{out}} &= K \quad \text{(número de filtros)} \end{aligned} \tag{3}$$

em que F é o tamanho do filtro (ex.: 3×3), K o número de filtros, S o *stride*, e P o *padding*.

Na primeira camada, K filtros são aplicados à imagem original, cada um produzindo um **mapa de características** distinto. Esses mapas são empilhados em um volume 3D de saída (Figura 5), cujas dimensões espaciais ($L_{\text{out}} \times A_{\text{out}}$) preservam a estrutura da imagem, enquanto a profundidade ($D_{\text{out}} = K$) codifica as características detectadas pelos filtros.

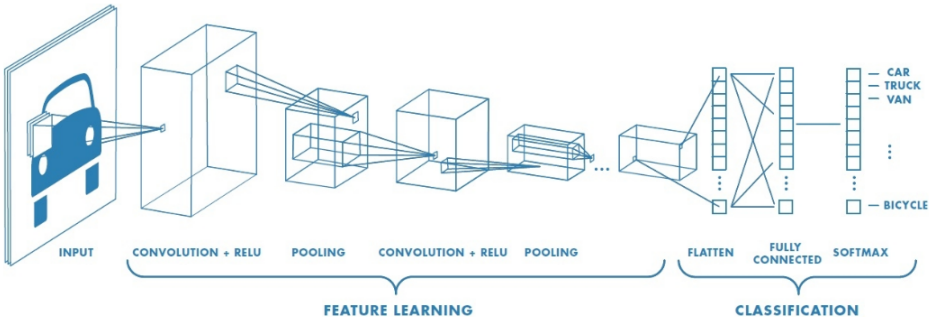

Figura 4: Arquitetura típica de CNN. As primeiras camadas detectam características locais (bordas, texturas), enquanto as profundas integram informações para reconhecimento de padrões complexos. Note o **efeito funil**: as dimensões espaciais (altura/largura) reduzem-se progressivamente, enquanto a profundidade (canais) aumenta, concentrando a informação relevante.

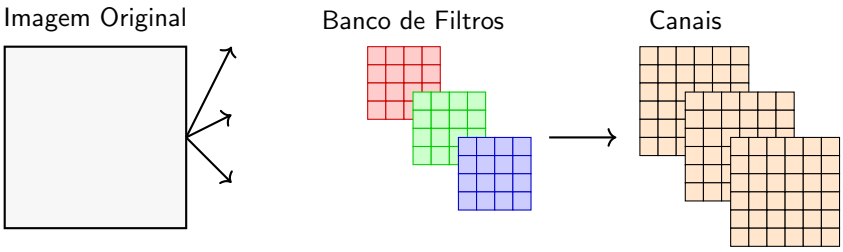

Figura 5: Processo de aplicação de múltiplos filtros: (1) Cada filtro gera um mapa de ativação distinto; (2) Os mapas são empilhados formando um volume 3D de características.

Conforme os dados fluem através das camadas da rede, sua estrutura sofre uma transformação. Nas primeiras etapas, operações de *pooling* e convoluções com *stride* maior que 1 ($S > 1$) começam a reduzir as dimensões espaciais. Uma imagem de entrada de 224×224 pixels, por exemplo, pode ser gradualmente reduzida a apenas 7×7 na última camada convolucional - como uma fotografia que, vista de longe, perde detalhes mas revela a essência da cena.

Paralelamente, ocorre um fenômeno oposto nos canais de profundidade. Onde inicialmente tínhamos apenas 3 canais (RGB), a dimensão D se expande para 256 ou mais. Cada novo canal atua como um especialista, aprendendo a detectar padrões específicos - desde bordas simples até formas complexas e texturas.

O resultado final é um **efeito funil** computacional: a informação espacial bruta, inicialmente dispersa em milhares de pixels, condensa-se em representações cada vez mais compactas e semanticamente ricas - a essência do aprendizado hierárquico em redes convolucionais.

Tabela 1: Complexidade de parâmetros por tipo de camada

Camada	**CONV**	**POOL**	**FC**
Entrada	$I \times I \times C$	$I \times I \times C$	N_{in}
Saída	$O \times O \times K$	$O \times O \times C$	N_{out}
Parâmetros	$(F^2 \cdot C + 1) \cdot K$	0	$(N_{\text{in}} + 1)N_{\text{out}}$

4.1 Complexidade do Modelo

A Tabela 1 resume os parâmetros por tipo de camada:

5 Transferência de Aprendizado

O treinamento de redes neurais convolucionais a partir do zero é incomum na prática, principalmente devido à exigência de grandes conjuntos de dados. A solução amplamente adotada é a transferência de aprendizado, implementada de duas formas:

- **Ajuste Fino (Fine-Tuning)**: Modifica pesos específicos de uma rede pré-treinada enquanto mantém sua arquitetura original. A técnica padrão congela as camadas iniciais – responsáveis por detectar padrões visuais básicos – e ajusta apenas as camadas finais, que contêm características específicas do domínio. Esse método exige atenção redobrada para prevenir *overfitting*, particularmente com conjuntos de dados limitados.
- **CNN como Extrator de Características**: Transforma redes pré-treinadas em extratores de features poderosos ao remover sua camada de classificação final. As ativações das camadas intermediárias, como os vetores 4096-D da penúltima camada da AlexNet, servem como entrada para novos classificadores (SVMs ou modelos lineares). Essa abordagem preserva o conhecimento hierárquico aprendido enquanto se adapta a novas tarefas.

Essas estratégias aproveitam arquiteturas consolidadas e pesos otimizados disponíveis em repositórios como Hugging Face ou Pytorch Model Zoo. Além de poupar semanas de treinamento, essa prática oferece acesso imediato a modelos de alta performance como VGG, ResNet e EfficientNet com ajustes mínimos para novas aplicações.

6 Exercícios

1. Projete uma arquitetura CNN para o dataset MNIST com 2 camadas convolucionais e uma densa.

2. Compare o número de parâmetros de sua rede com a rede neural tradicional das aulas 9 e 10.