

# **Redes Neurais e Deep Learning**

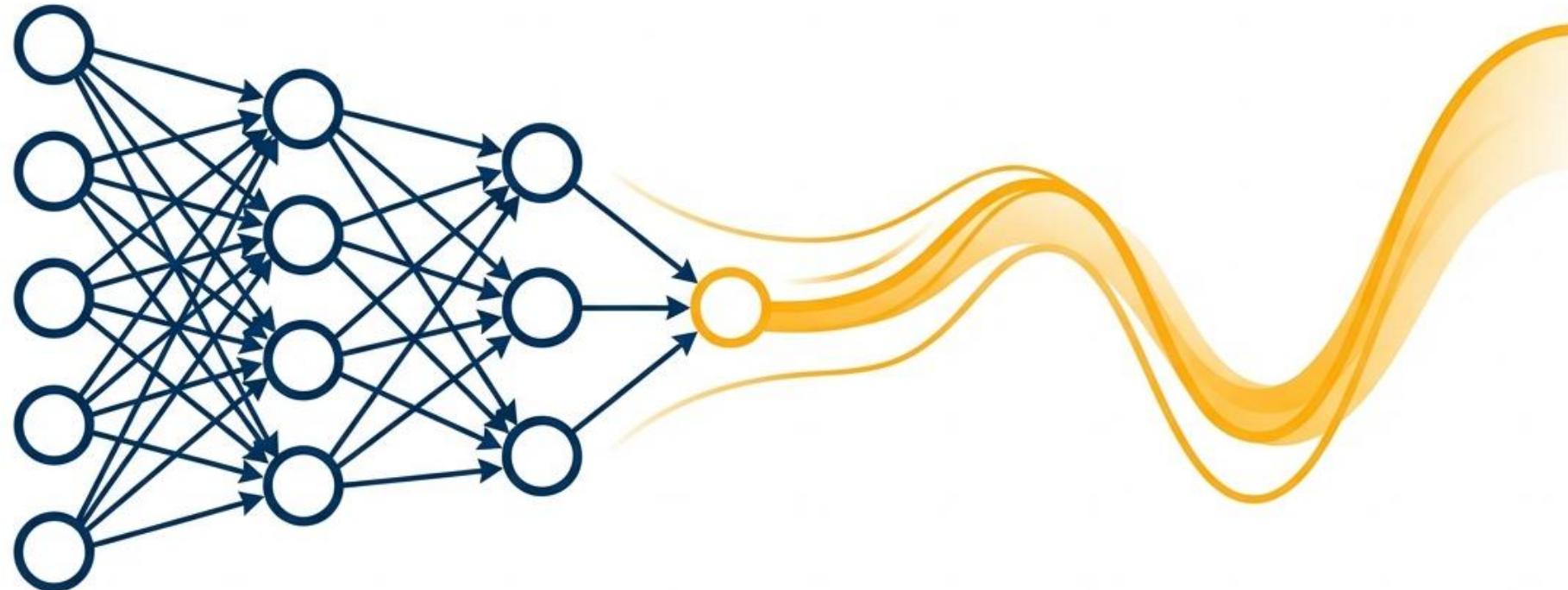
## **Prof. Domingos Napolitano**

Aula 1: Treinamento de Redes Neurais Densas e Redes Neurais  
Convolucionais CNN



# Treinamento Eficiente de Redes Neurais Profundas

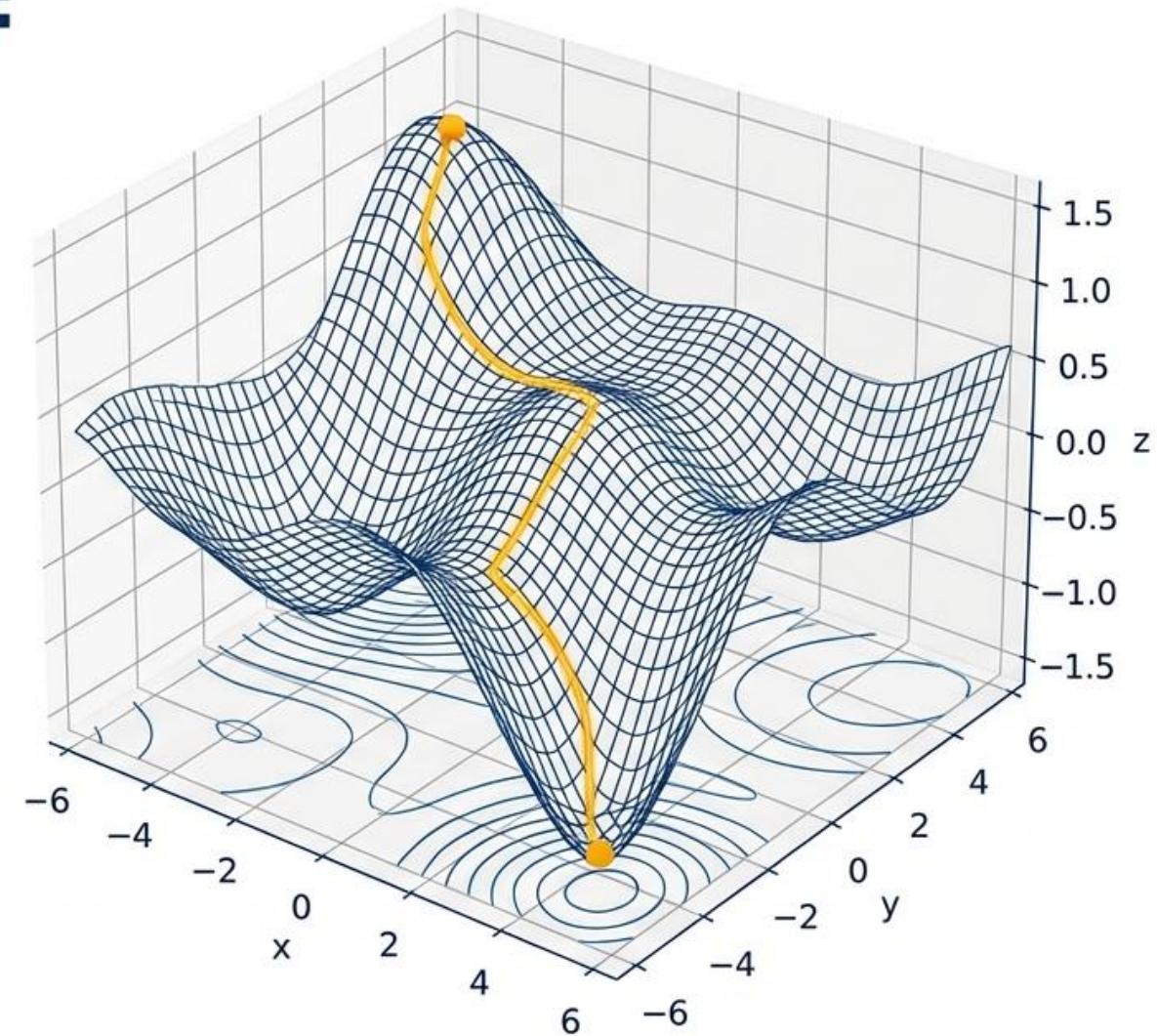
## Da Estabilidade dos Gradientes à Generalização Robusta



ESTRATÉGIAS DE OTIMIZAÇÃO E REGULARIZAÇÃO

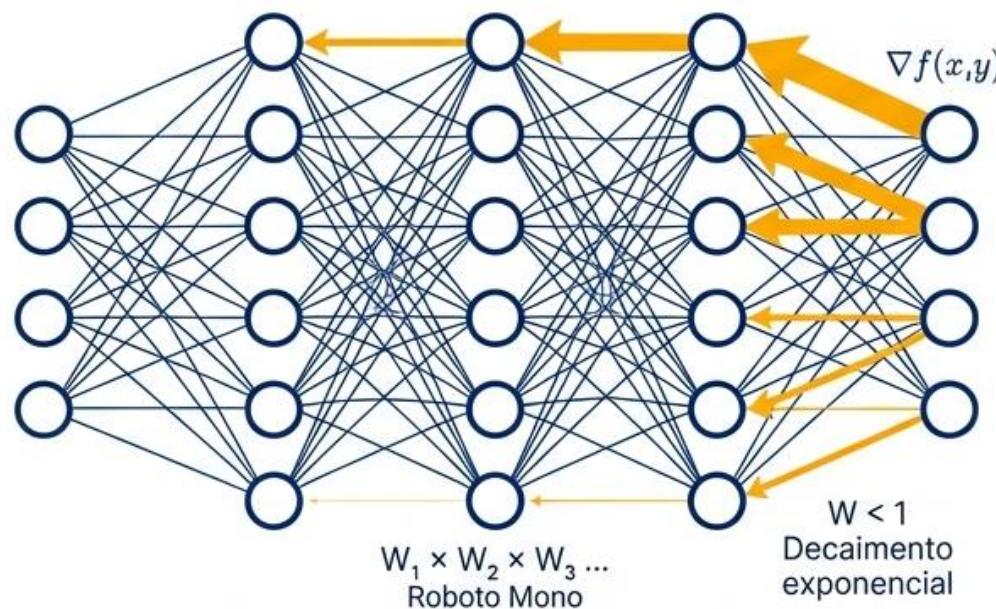
# O Motor do Aprendizado: Gradientes e Backpropagation

- **Conceito:** O treinamento é uma caminhada em busca do erro mínimo.
- **Intuição:** Descer uma montanha de olhos vendados, sentindo a inclinação.
- **Matemática:** O gradiente  $\nabla f(x, y)$  aponta a subida; caminhamos no sentido oposto.
- **Processo:** Forward Pass (Previsão) → Cálculo do Erro → Backward Pass (Ajuste).

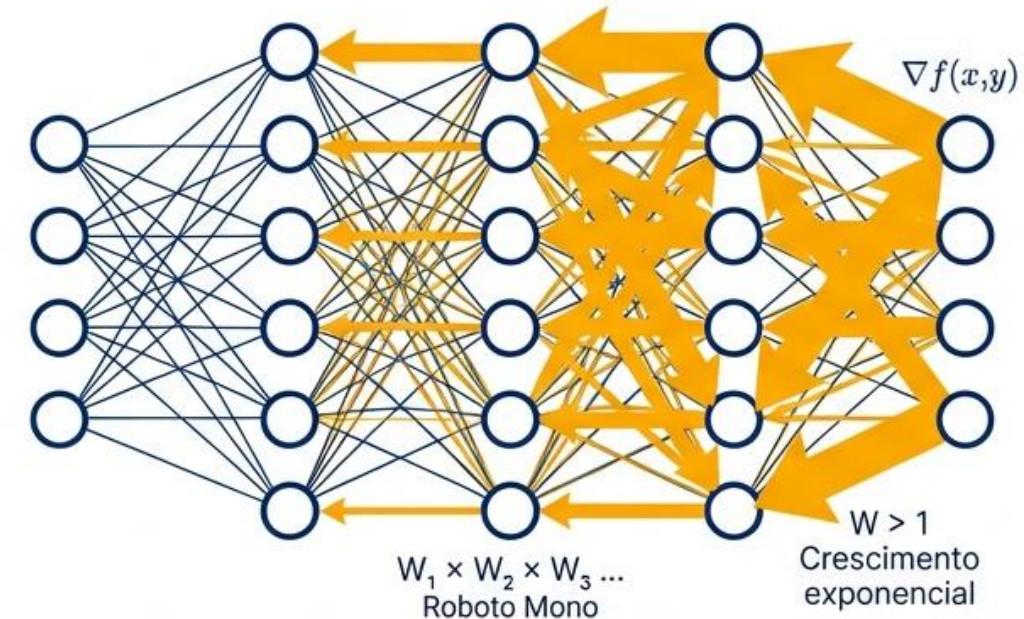


# O Problema dos Gradientes Instáveis

Vanishing (Desvanecente)



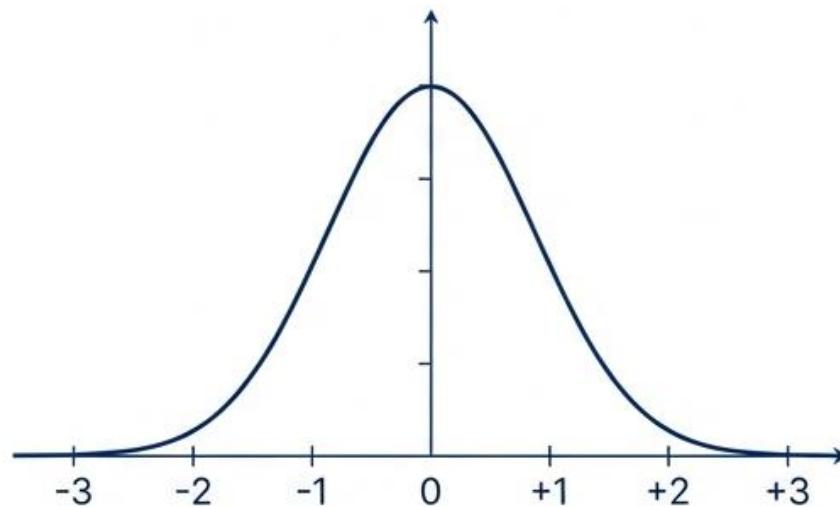
Exploding (Explosivo)



- **O Problema:** A “mensagem” de erro se corrompe em redes profundas.
- **Analogia (Telefone Sem Fio):** O sinal sussurado desaparece (Vanishing) ou o sinal gritado distorce (Exploding).
- **Causa:** Multiplicação sucessiva de pesos  $W$ . Se  $W < 1$  (Decaimento exponencial). Se  $W > 1$  (Crescimento exponencial).

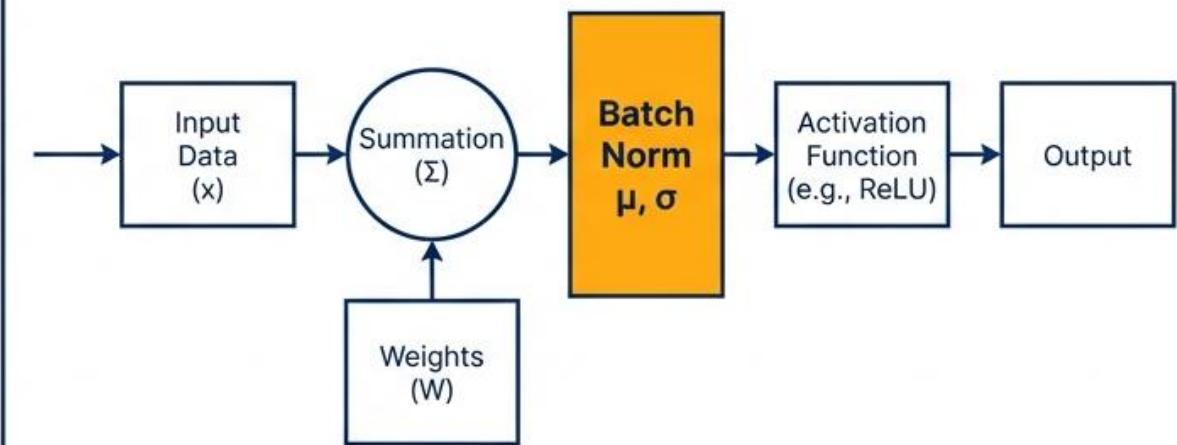
# A Solução Inicial: Inicialização e Normalização

## 1. Inicialização de He/Glorot



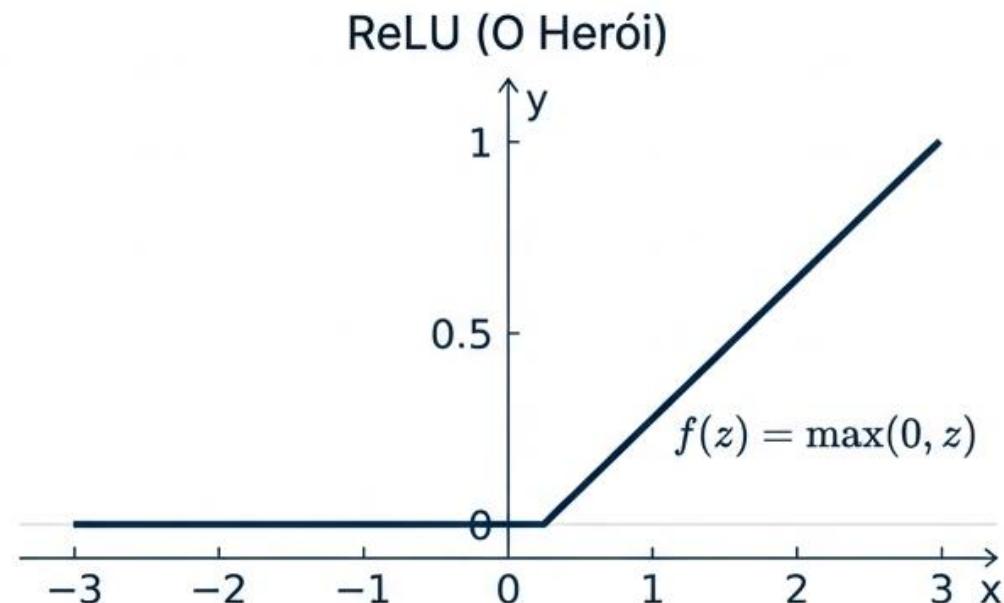
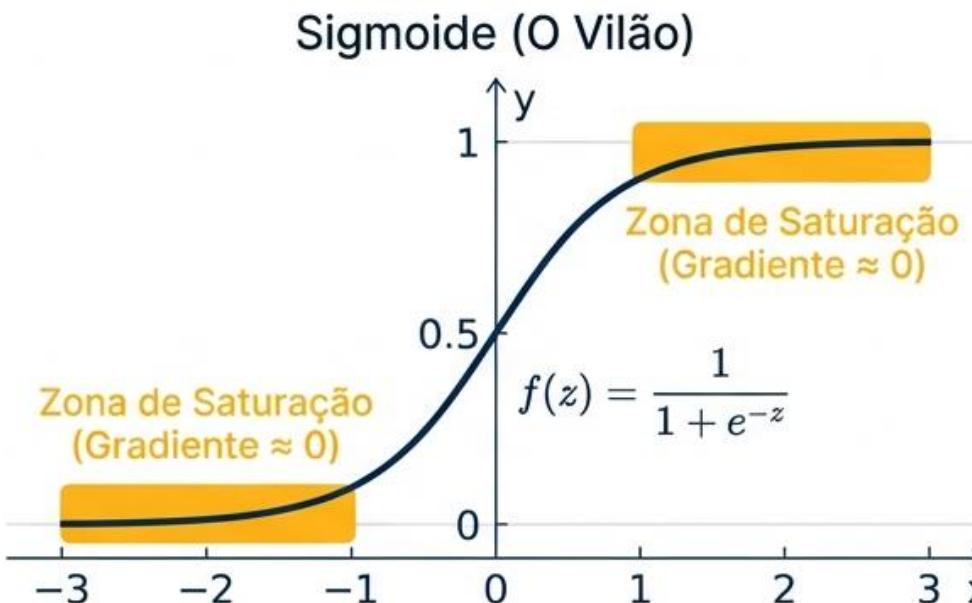
Objetivo: Variância da Saída = Variância da Entrada

## 2. Batch Normalization



Batch Norm re-centraliza e re-escala os dados a cada camada, permitindo taxas de aprendizado maiores.

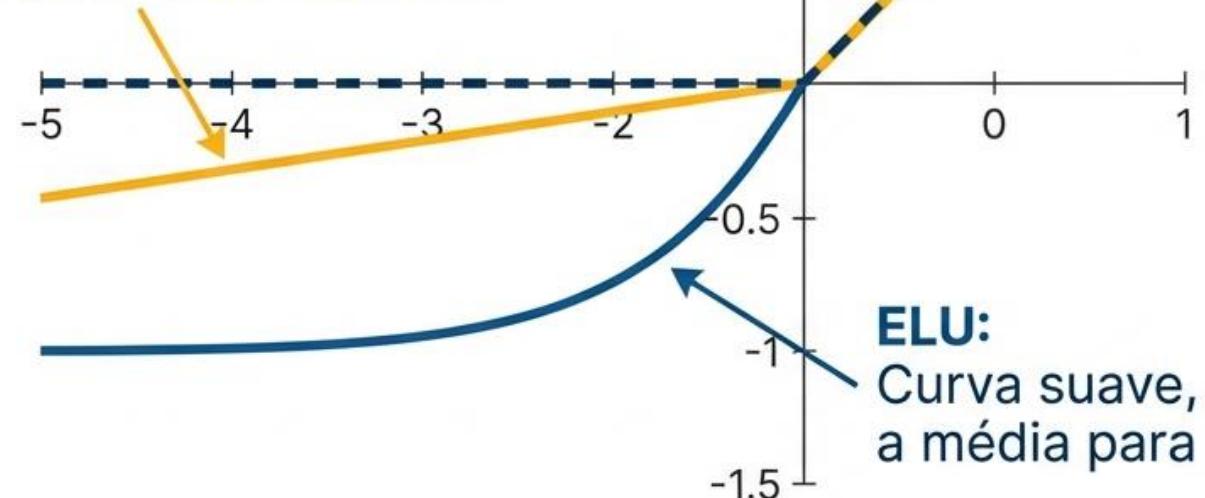
# Funções de Ativação: A Evolução



- **Sigmoide (O Vilão):** Satura nas extremidades. O gradiente desaparece, impedindo o aprendizado em redes profundas.
- **ReLU (O Herói):**  $f(z) = \max(0, z)$ . Não satura para valores positivos e é computacionalmente eficiente.
- **Dying ReLU:** O neurônio pode 'morrer' se a saída for sempre negativa (gradiente zero).

# Ativações Modernas: Revivendo os Neurônios

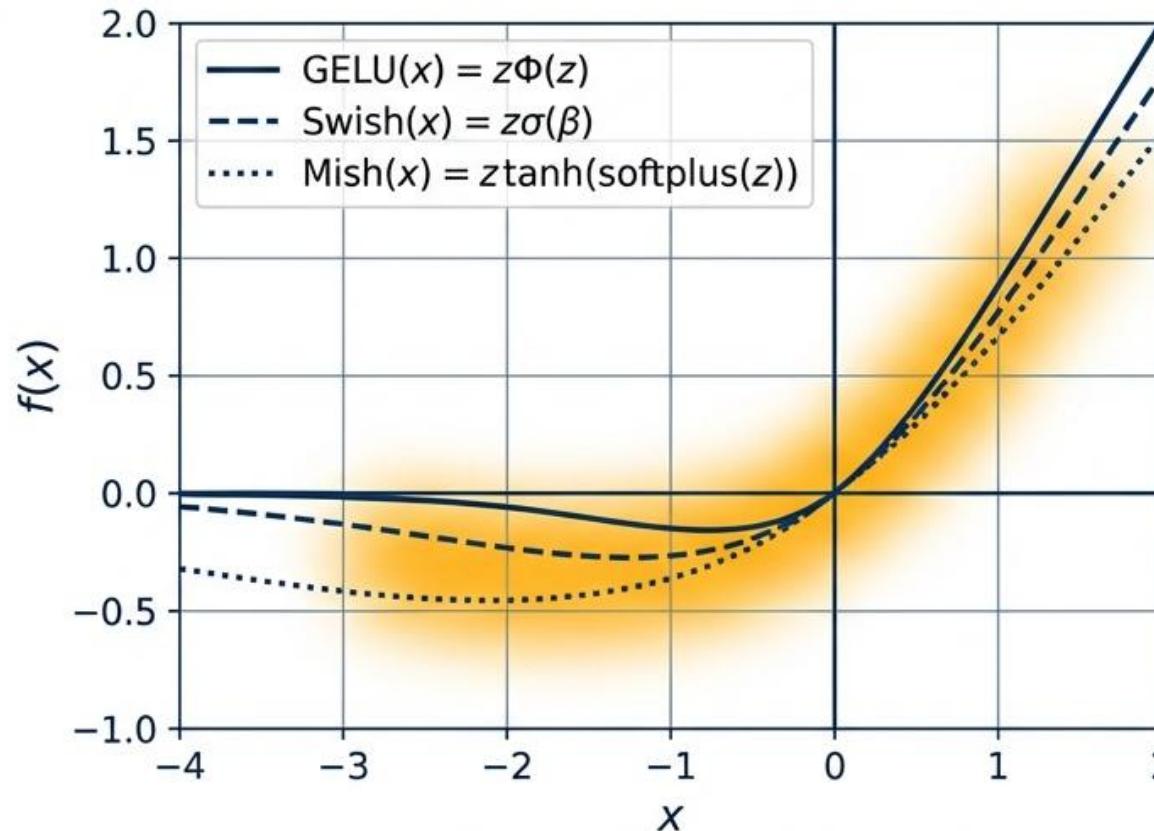
**Leaky ReLU:** Permite um pequeno gradiente negativo.



**ELU:**  
Curva suave, empurra a média para zero.

**SELU:** Propriedade de auto-normalização em redes densas.

# O Estado da Arte: Swish, GELU & Mish

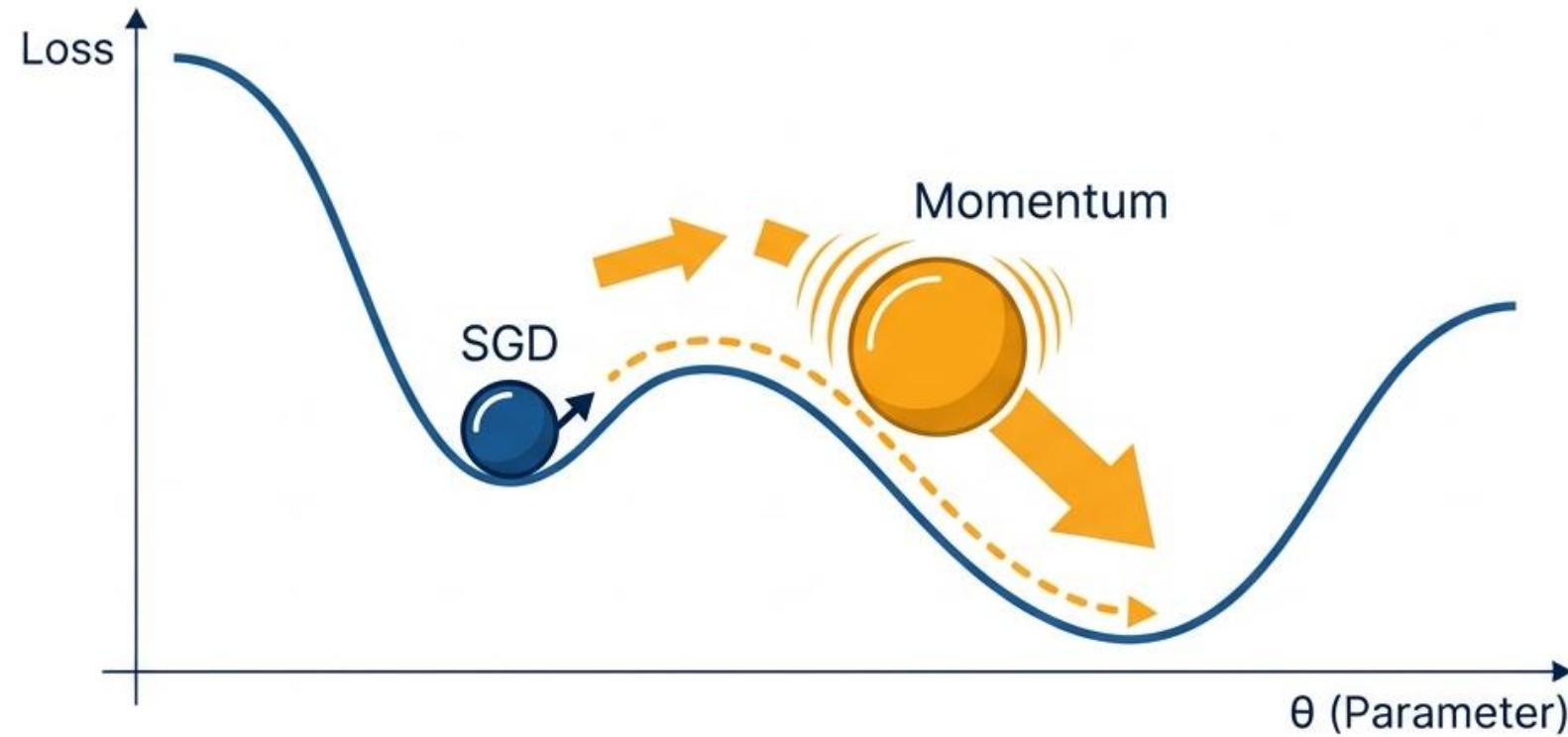


$$\text{Swish}(x) = x \cdot \sigma(\beta x)$$

- **Características:** Funções não-monotônicas e suaves (deriváveis em todo o domínio).
- **Uso:** Padrão em modelos de linguagem (Transformers, BERT, GPT).
- **Insight:** A suavidade da curva facilita a otimização em paisagens de perda complexas.

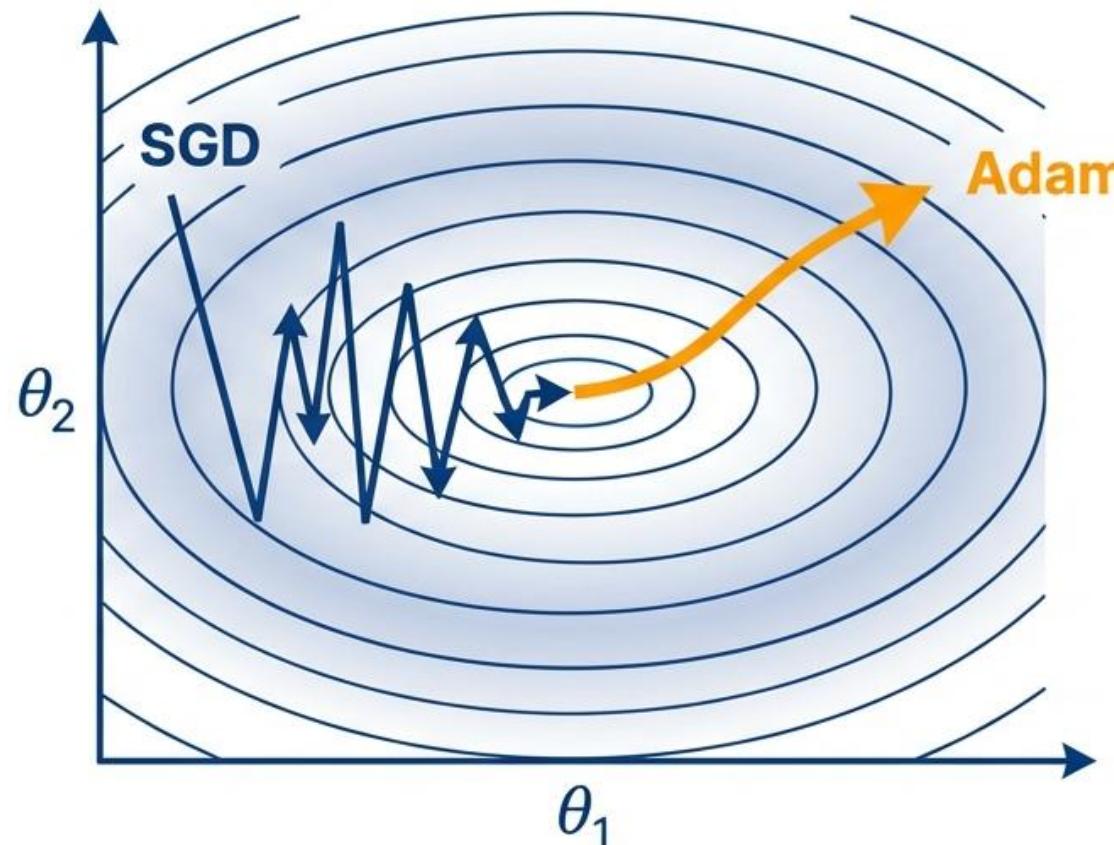
# Acelerando a Descida: Otimizadores (Momentum)

- **SGD:** Passos baseados apenas na inclinação atual. Lento em platôs.
- **Momentum:** Acumula velocidade das iterações anteriores (Inércia).
- **Benefício:** Atravessa mínimos locais e acelera em direções consistentes.



# Otimizadores Adaptativos: A Hegemonia do Adam

$$\text{Adam} = \text{Momentum} + \text{RMSProp}$$



- **Conceito:** Taxas de aprendizado individuais para cada parâmetro.
- **RMSProp:** Divide o gradiente por uma média móvel de sua magnitude recente.
- **Adam (Adaptive Moment Estimation):** Combina inércia (Momentum) com escala adaptativa (RMSProp). É o padrão de ouro atual.

# O Inimigo: Overfitting (Sobreajuste)

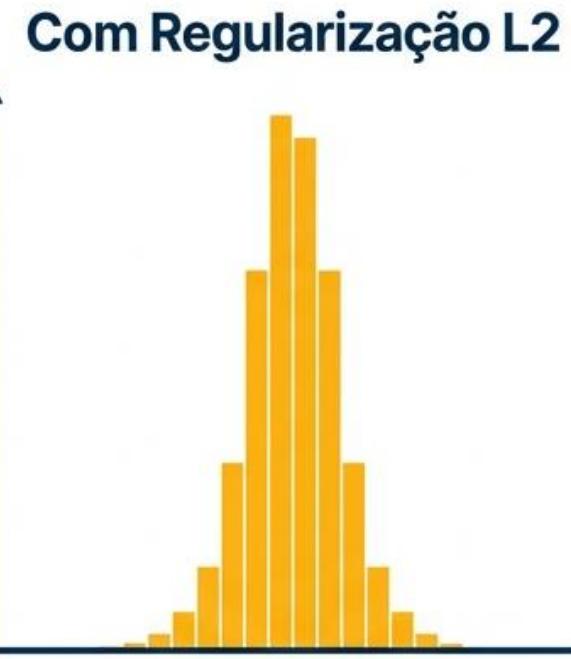
“Com quatro parâmetros posso ajustar um elefante...” — Von Neumann

O modelo ‘memoriza’ o ruído dos dados de treino e perde a capacidade de generalizar para dados novos.



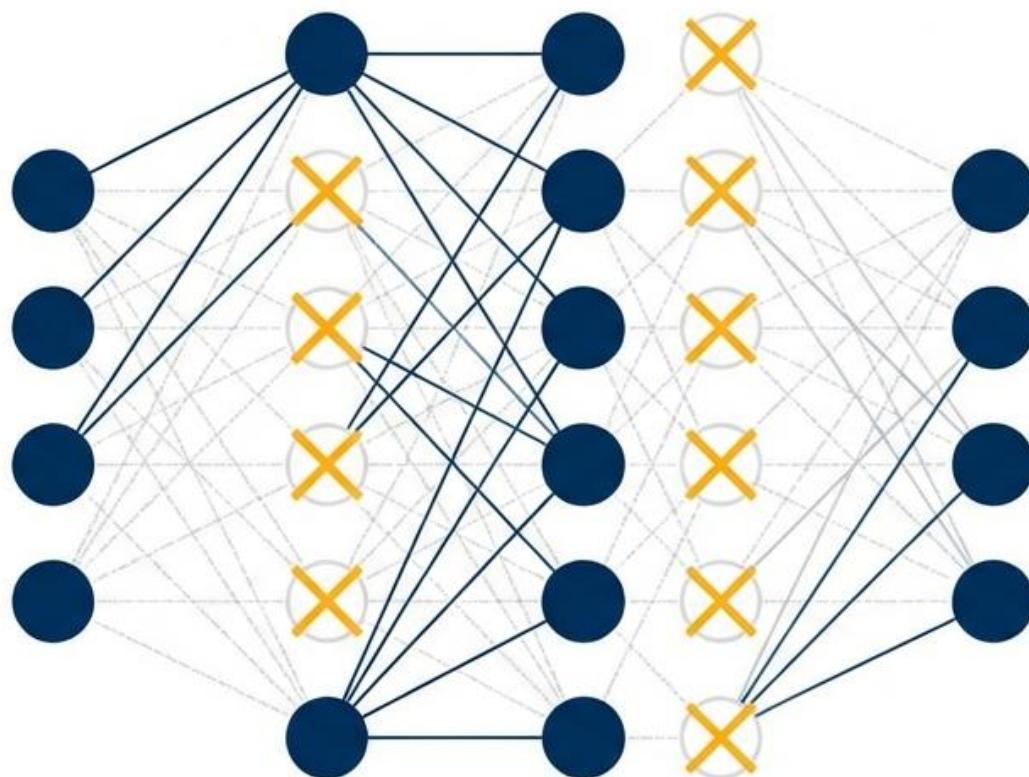
# Regularização L1, L2 e Max-Norm

$$J(\theta) + \lambda ||w||^2$$



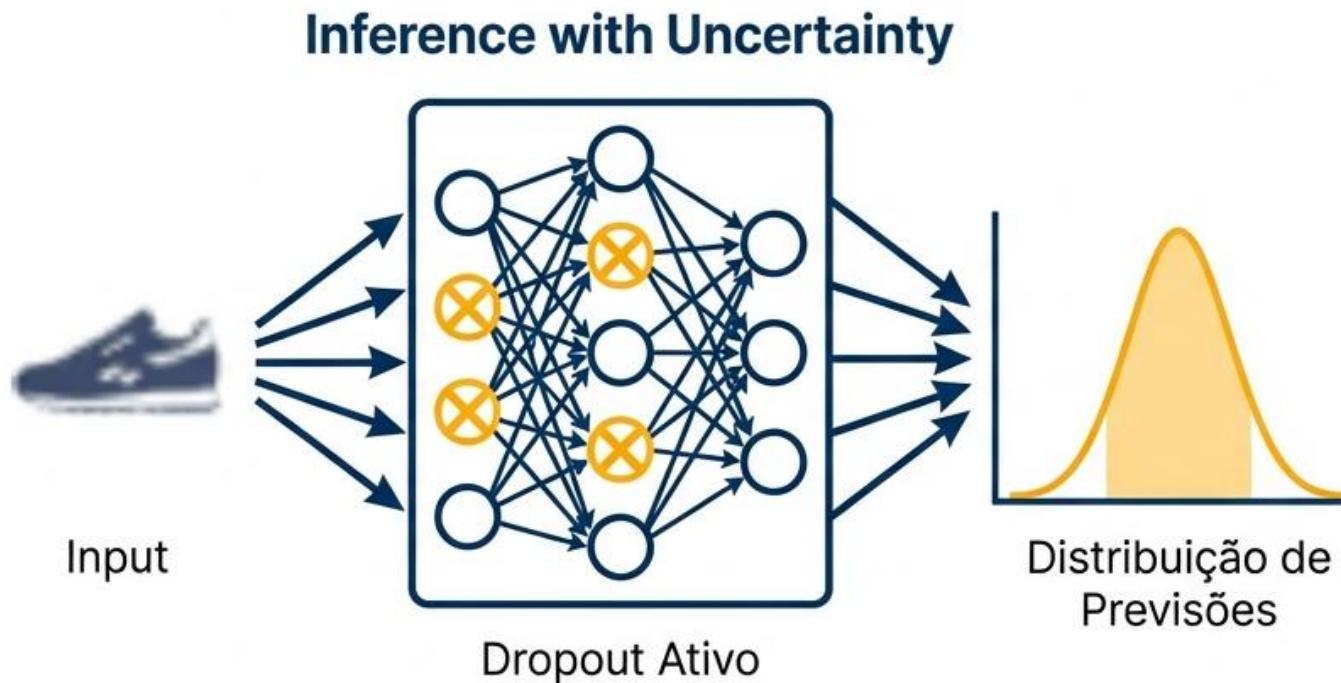
- **L2 (Weight Decay)**: Penaliza pesos grandes. Força uma distribuição difusa e pequena. (**Ideal para AdamW**).
- **L1 (Lasso)**: Força pesos irrelevantes a zero. Cria modelos esparsos (**Feature Selection**).
- **Max-Norm**: Corta o vetor de pesos se ele exceder um limite máximo " $r$ ".

# Dropout: A Arte do Esquecimento Estratégico



- **Mecânica:** Desligar aleatoriamente neurônios (ex: 50%) a cada passo de treino.
- **Analogia:** Uma empresa onde os funcionários mudam todo dia. Ninguém pode ser insubstituível; todos devem aprender a tarefa.
- **Resultado:** Quebra a co-adaptação e cria redundância robusta.

# Monte Carlo (MC) Dropout



- **Inovação:** Manter o Dropout ligado durante a inferência (teste).
- **Método:** Passar o mesmo dado N vezes pela rede.
- **Insight:** Se as respostas variam muito, o modelo está "confuso". Permite medir incerteza sem retreinar.

# Resumo Prático: Configurações Recomendadas

## Configuração Padrão

- Ativação: **ReLU**
- Init: **He Initialization**
- Otimizador: **Adam**
- Reg: **Early Stopping**

## Rede Profunda / Complexa

- Ativação: **Swish** ou **GELU**
- Extra: **Batch Normalization**

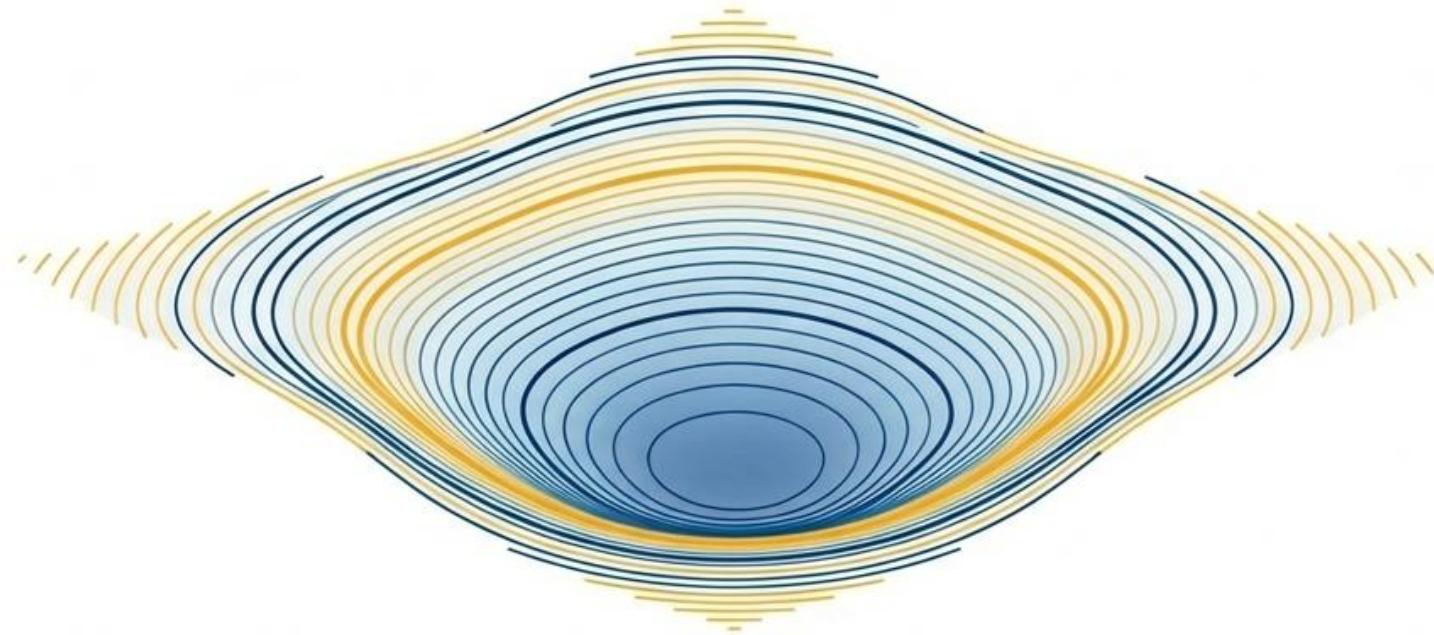
## Poucos Dados (Overfitting)

- Regularização: **Dropout + L2 (Weight Decay)**
- Estratégia: **Transfer Learning**

## Rede Apenas Densa (MLP)

- Ativação: **SELU (Auto-normalização)**
- Init: **LeCun Normal**

# Conclusão

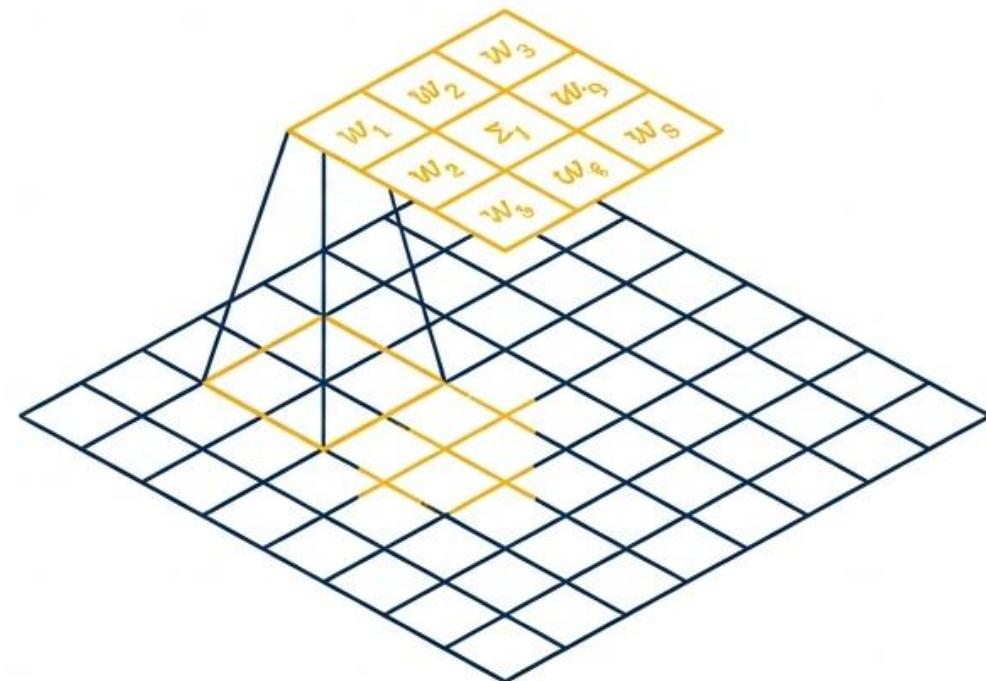


O sucesso no Deep Learning é o equilíbrio entre  
**Estabilidade** (Gradientes/BN), **Velocidade** (Otimizadores)  
e **Generalização** (Regularização).

Referências: Geron, A. 'Hands-On Machine Learning'; Glorot & Bengio (2010); He et al. (2015); Hinton et al. (2012).

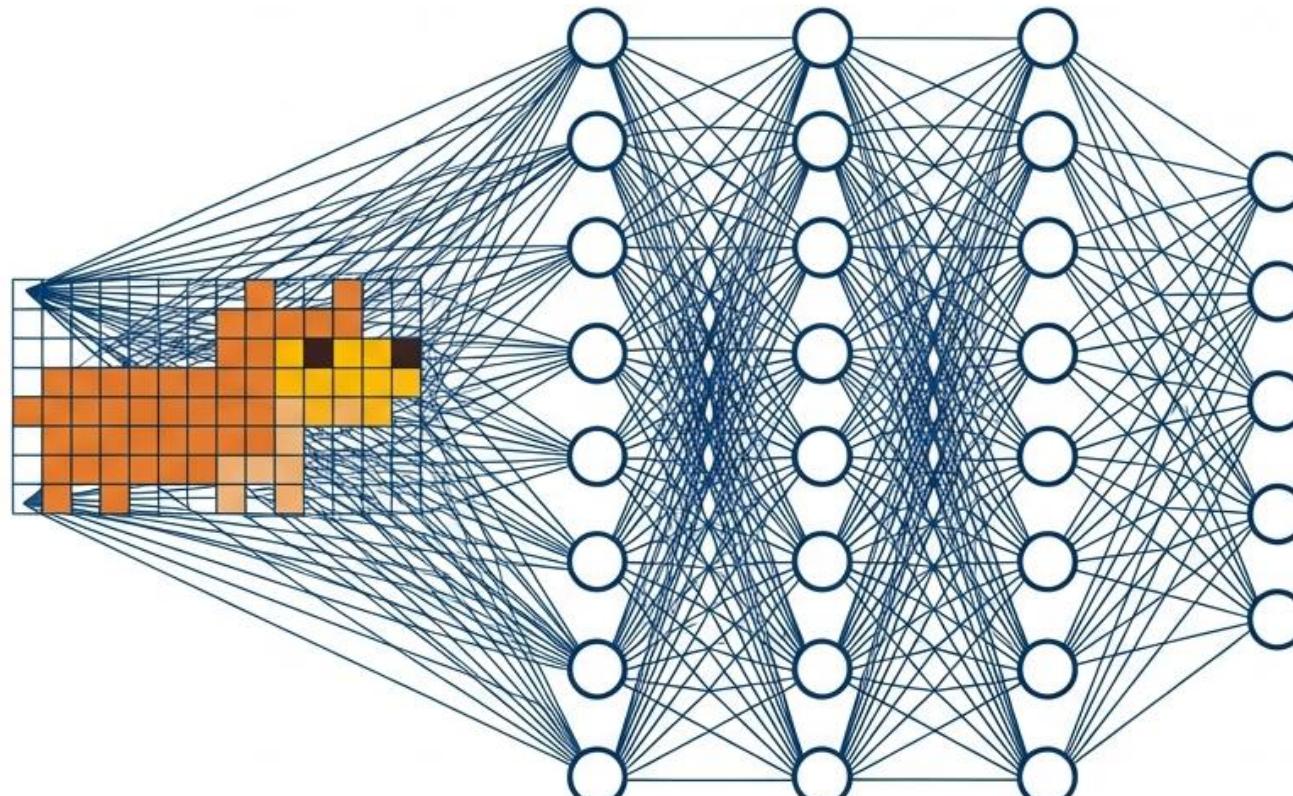
# Redes Neurais Convolucionais: Arquitetura, Operações e Eficiência

Uma jornada da teoria matemática às aplicações modernas de Visão Computacional



TREINAMENTO EFICIENTE DE REDES NEURAIS

# O Problema da Escala e Invariância



Rede Densa (MLP)

## The Analysis

### Explosão de Parâmetros

Uma imagem de 100x100 pixels gera 10 milhões de conexões na primeira camada. O custo computacional torna-se inviável.

### Falta de Invariância

Se o objeto move-se do canto superior para o inferior, uma rede densa o interpreta como uma entrada totalmente nova.

# Inspiração Biológica: O Córtex Visual



Células Simples  
(Bordas)

Células Complexas  
(Formas)

Objeto  
(Abstração)

\*\*Campos Receptivos (Hubel & Wiesel, 1962):\*\* O cérebro processa informações em **camadas hierárquicas**. Neurônios reagem apenas a regiões específicas do campo visual, combinando bordas simples para formar objetos complexos.

# O Input: O Computador Vê Números

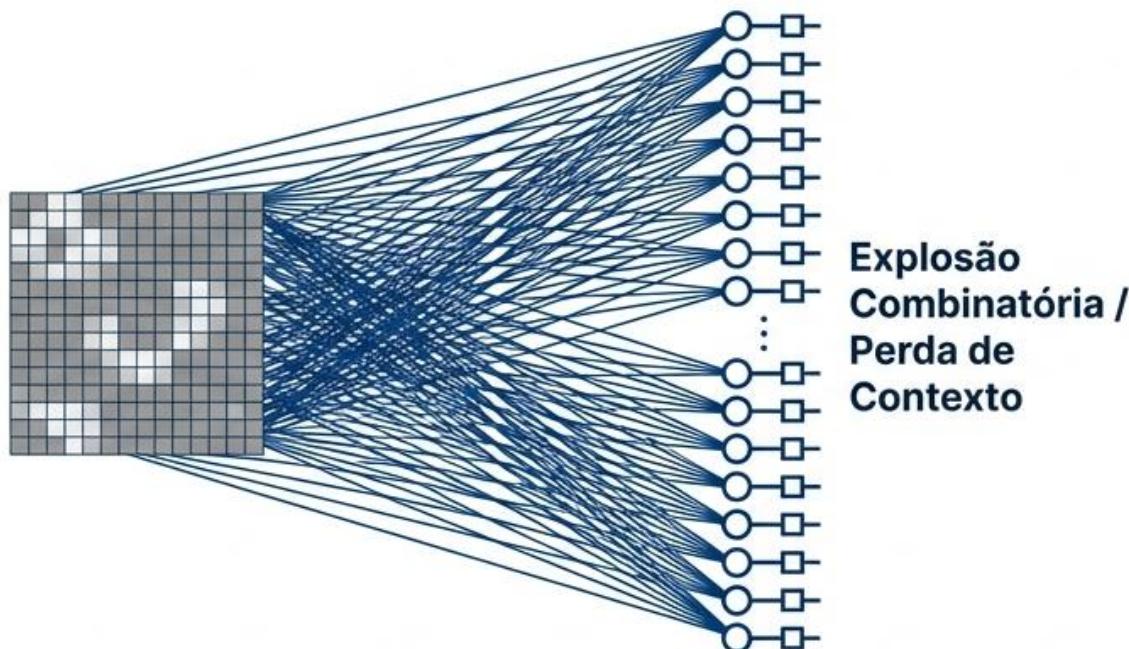


- Imagem 32x32 pixels = 1.024 entradas
- Canais RGB (x3) = 3.072 valores

Para o computador, não existem formas ou sorrisos, apenas matrizes de intensidade.

# O Problema das Redes Densas Tradicionais

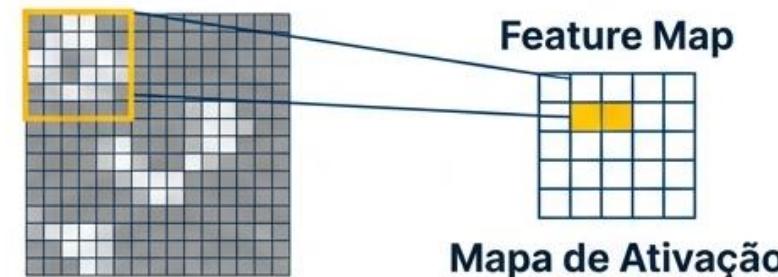
**Rede Densa (ANN)**



A complexidade cresce exponencialmente com o tamanho da imagem, perdendo a estrutura espacial.

Redes densas ignoram que pixels vizinhos formam estruturas (como um olho ou uma borda).

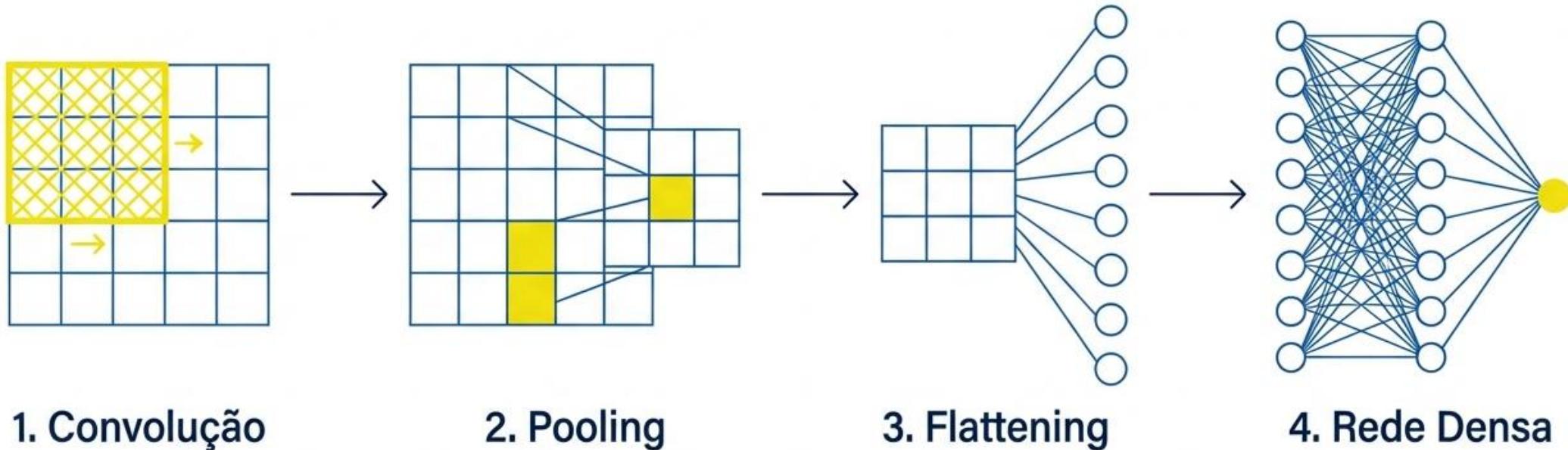
**Rede Convolucional (CNN)**



O kernel desliza pela imagem, capturando padrões locais e mantendo as relações espaciais.

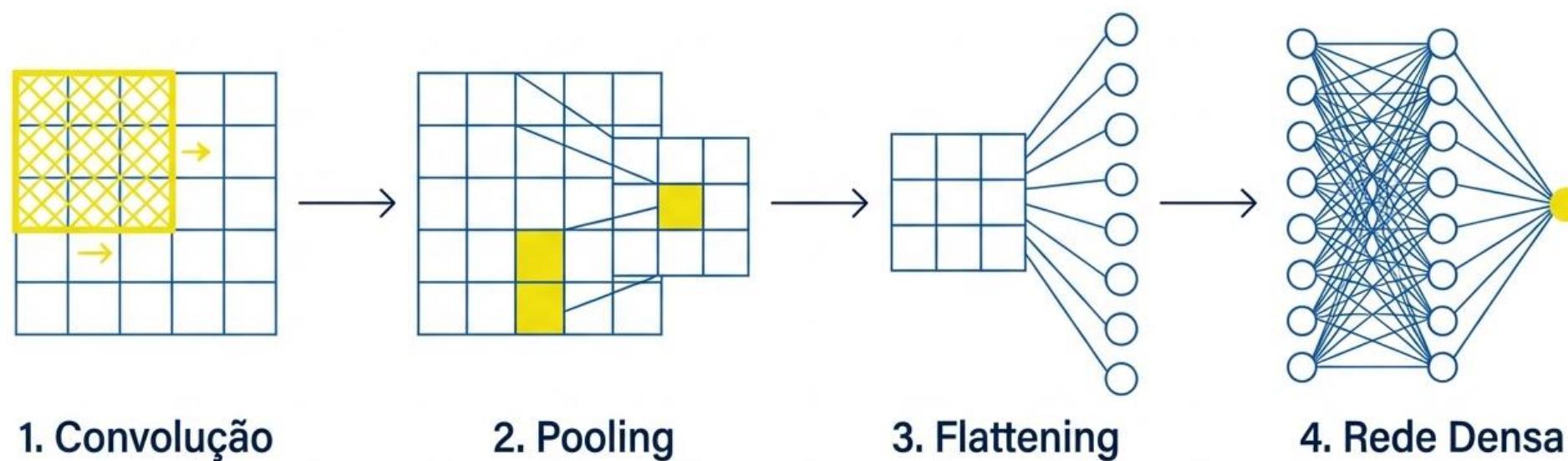


# A Arquitetura em 4 Etapas



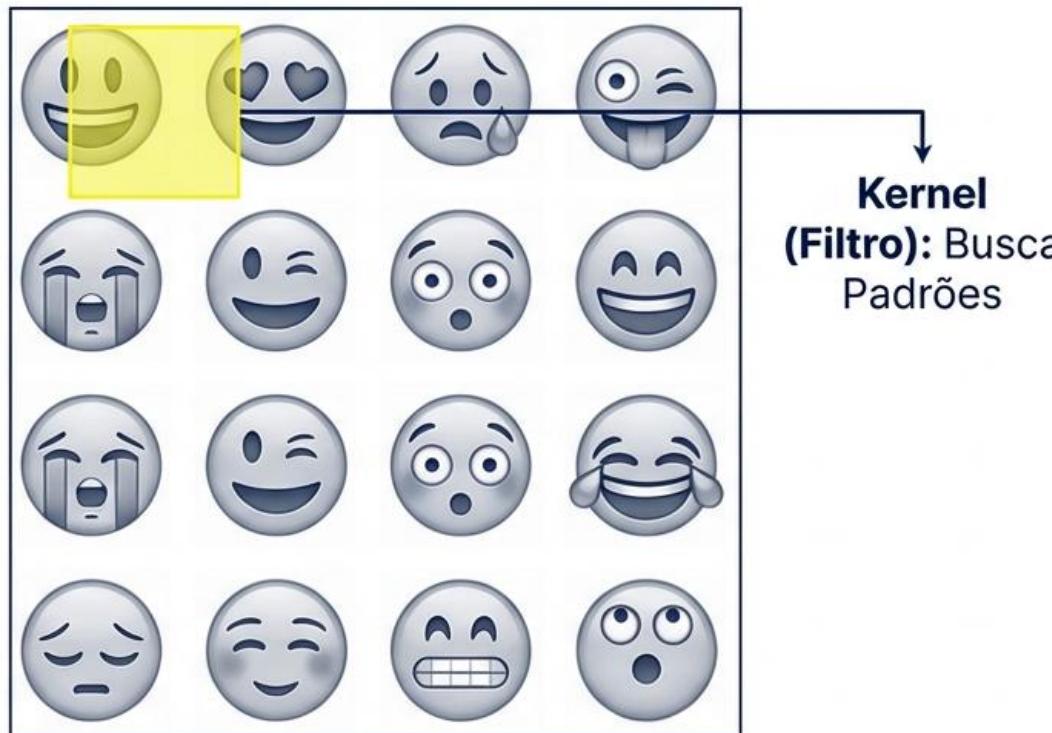
A CNN funciona como uma linha de montagem: extrai características, resume as informações e, por fim, toma uma decisão.

# A Arquitetura em 4 Etapas



A CNN funciona como uma linha de montagem: extrai características, resume as informações e, por fim, toma uma decisão.

# Etapa 1: Convolução e Extração de Características



- O '**Kernel**' funciona como uma lanterna que varre a imagem.
- **Objetivo:** Encontrar **características** específicas (bordas, curvas, texturas).
- **Exemplos:** Distância entre olhos, formato do nariz.
- **Importante:** A rede aprende estes filtros sozinha.

# A Matemática do Operador de Convolução

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] g[n - m]$$

Convolução = Produto Escalar entre a Imagem (f) e o Filtro (g).

# Calculando o Mapa de Características

1	0	1
0	1	0
1	0	1

×

1	0	1
0	1	0
1	0	1

↑  
Imagen (Seção)

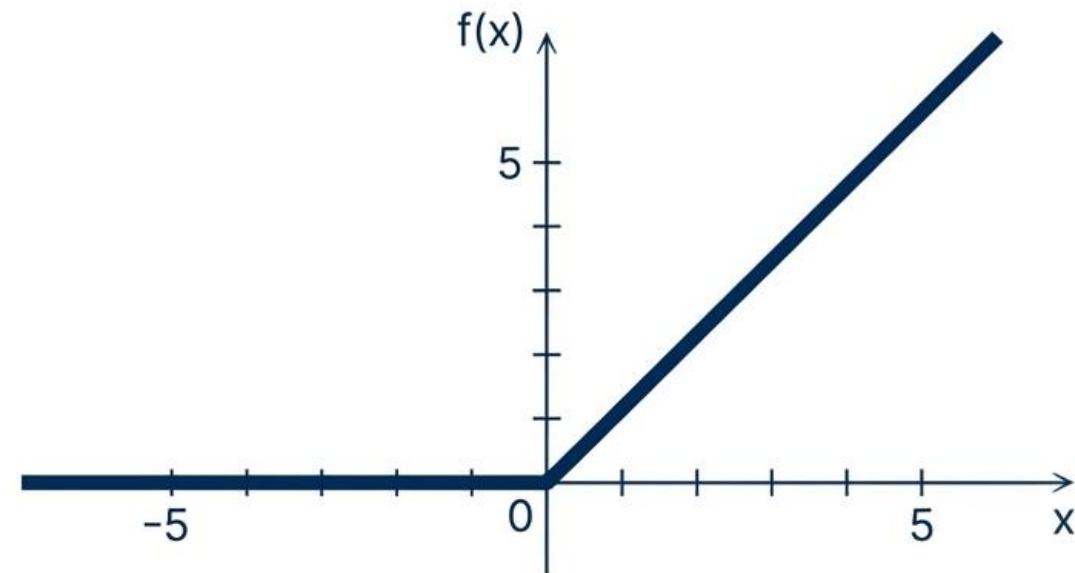
↑  
Kernel

→  
Soma dos  
Produtos = 5


Feature Map  
(Resultado)

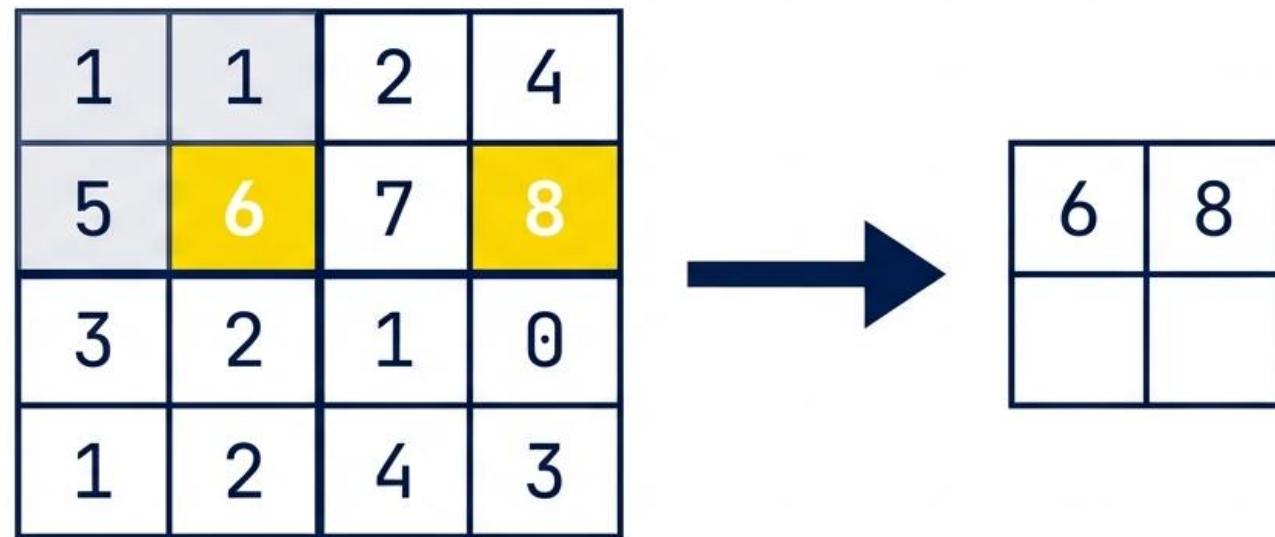
Números altos indicam que o padrão procurado foi encontrado.

# Não-Linearidade: A Camada ReLU



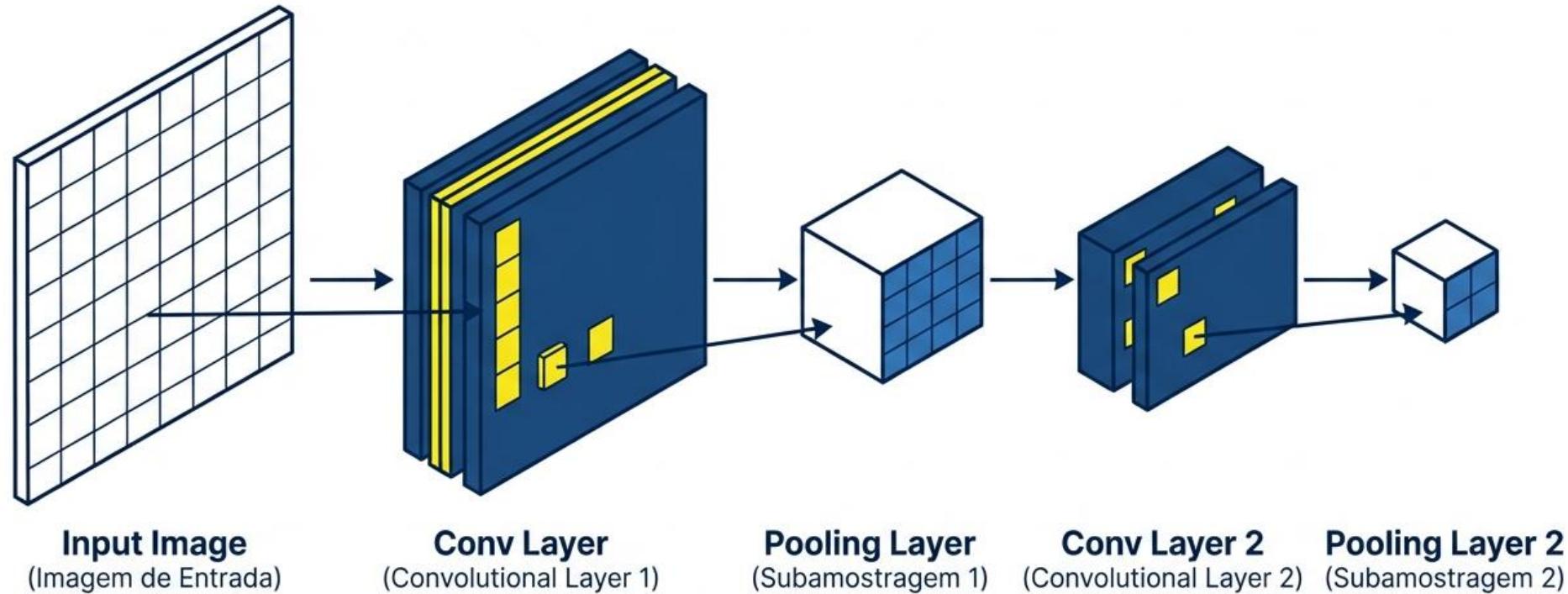
- ReLU (Rectified Linear Unit)
- Remove valores negativos (pixels pretos/ausência de sinal).
- Quebra a linearidade, permitindo que a rede aprenda formas complexas.

## Etapa 2: Pooling (Subamostragem)



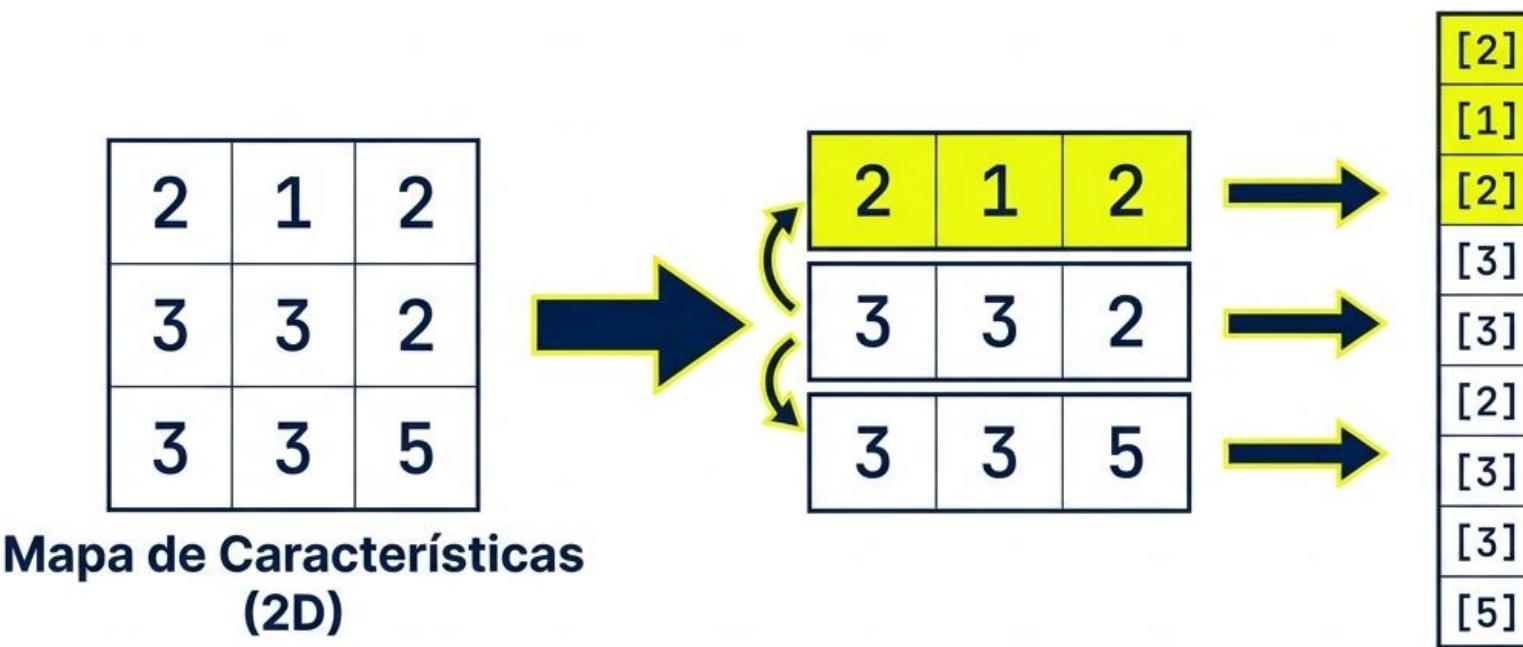
- Reduz a dimensionalidade da imagem.
- Preserva apenas a característica mais forte (Max Pooling).
- Evita Overfitting e reduz custo computacional.

# A Pilha de Extração (Deep Learning)



A imagem diminui em tamanho (pixels), mas aumenta em profundidade (abstração).

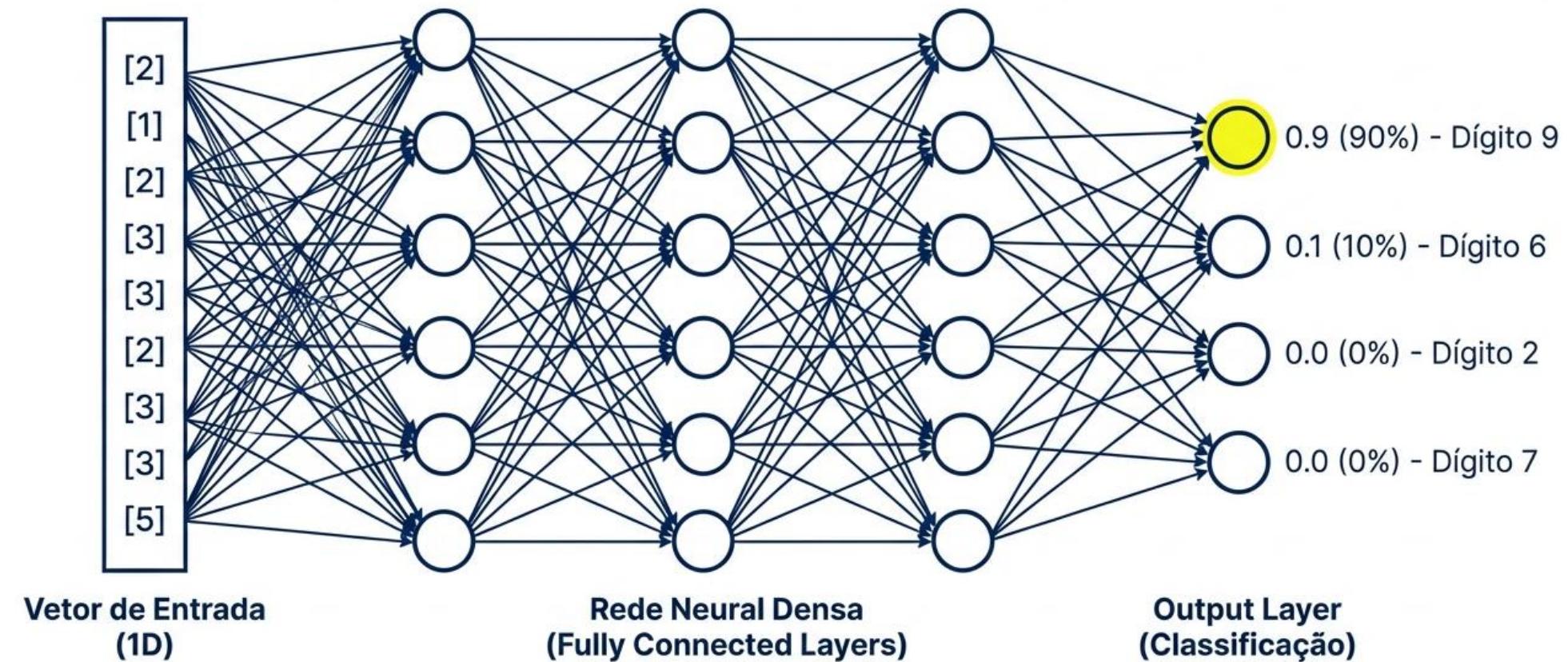
## Etapa 3: Flattening (Achatamento)



Transformando matrizes em uma lista para a Rede Neural Densa.

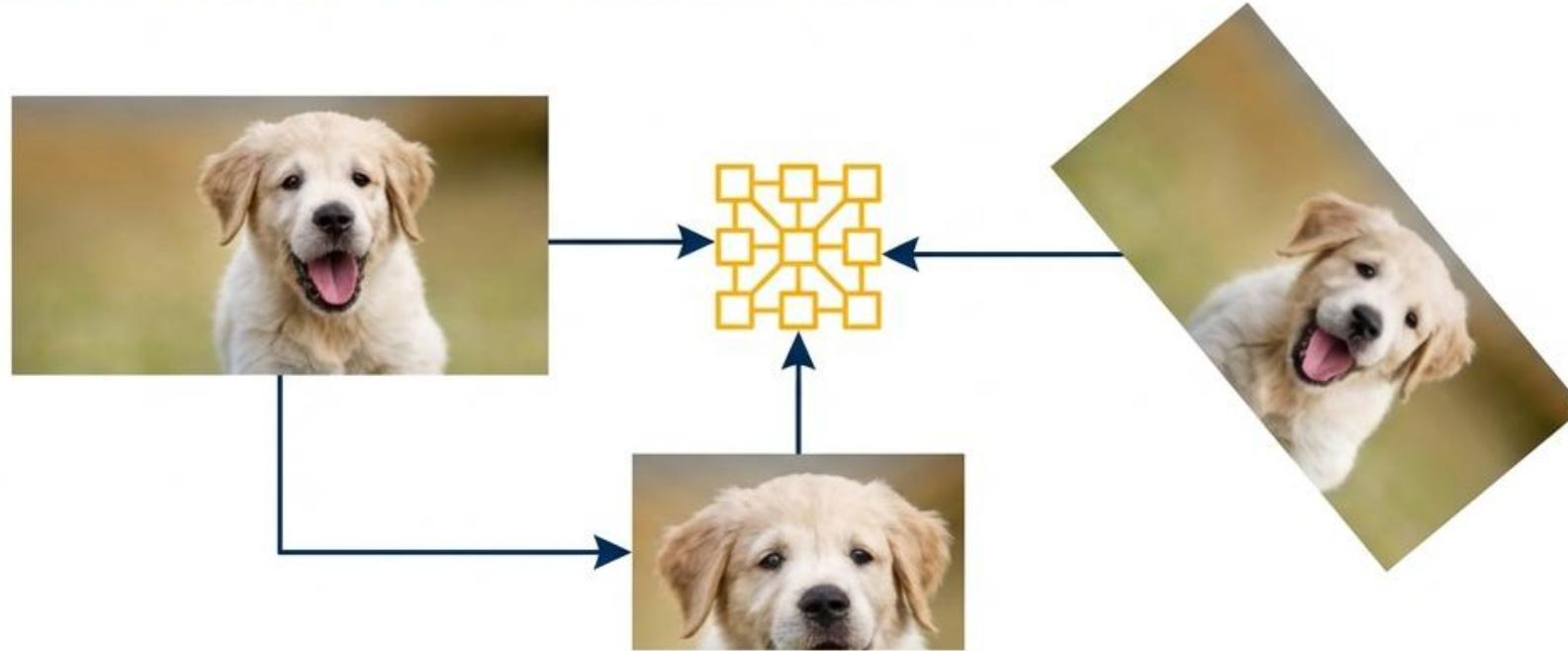
**Vetor de Entrada (1D)**

## Etapa 4: Classificação via Rede Neural Densa



A camada Softmax converte os sinais finais em probabilidades.

# Aprendizado e Robustez



- **Invariância de Posição:** A CNN encontra a "orelha" não importa onde ela esteja ou se a imagem está girada.
- **Backpropagation:** O sistema ajusta os pesos dos filtros automaticamente através da "Descida do Gradiente" para corrigir erros.

# A Operação de Convolução

**Kernel:** Filtro  
Extrator de Padrões

157	161	159	148	147	136...
-1	8	-1	142	129	122
-1	-1	-1	192	140	100
169	152	157	116	97	84
124	148	144	170	134	94
116	128	130	147	149	132

**Input:**  
Imagen de Entrada

**Feature Map:**  
Resultado da Ativação

31	$1 \cdot 157 + -1 \cdot 161 + -1 \cdot 159 + \dots + -1 \cdot 125 = 31$

**Feature Map:**  
Resultado da Ativação

O kernel desliza sobre a imagem multiplicando seus valores elemento a elemento. O resultado é a soma ponderada, indicando a presença de uma **característica** **característica** específica (como uma borda) naquela região.

# A Matemática da Convolução

Input: Valor do Pixel de Entrada

$$z_{i,j,k} = b_k + \sum \sum x_{i',j',k'} \times w_{u,v,k',k}$$

Bias: Ajuste de ativação

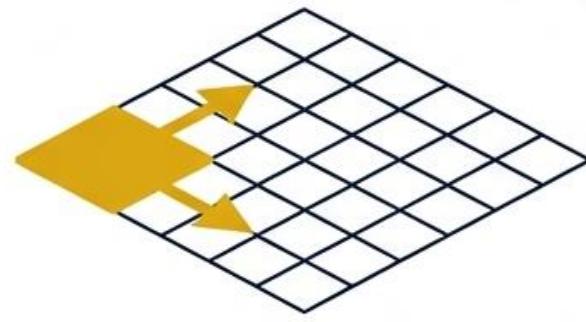
Pesos (Weights): Parâmetros aprendidos pelo Kernel

Soma Ponderada: Correlação espacial local

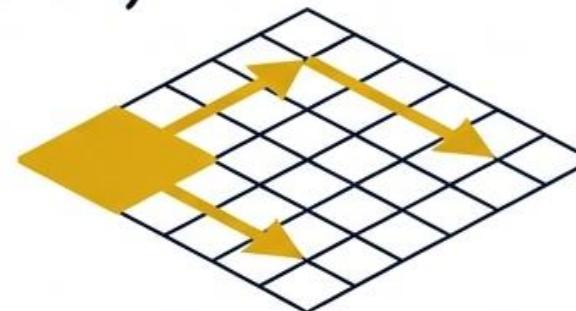
Matematicamente, a convolução é uma busca por similaridade. Quanto maior o resultado da soma, mais o trecho da imagem se parece com o padrão que o kernel procura.

# Controlando a Visão: Stride e Padding

**Stride (Passo)**



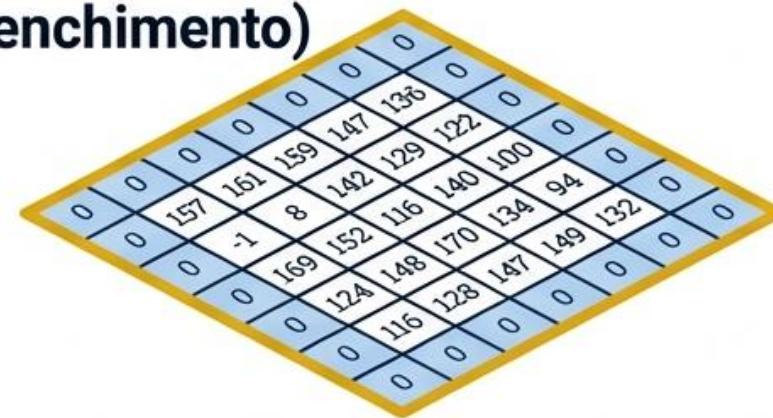
Stride 1



Stride 2 - Downsampling

**Stride:** Define a velocidade do deslocamento. Passos largos reduzem a dimensão da saída.

**Padding (Preenchimento)**

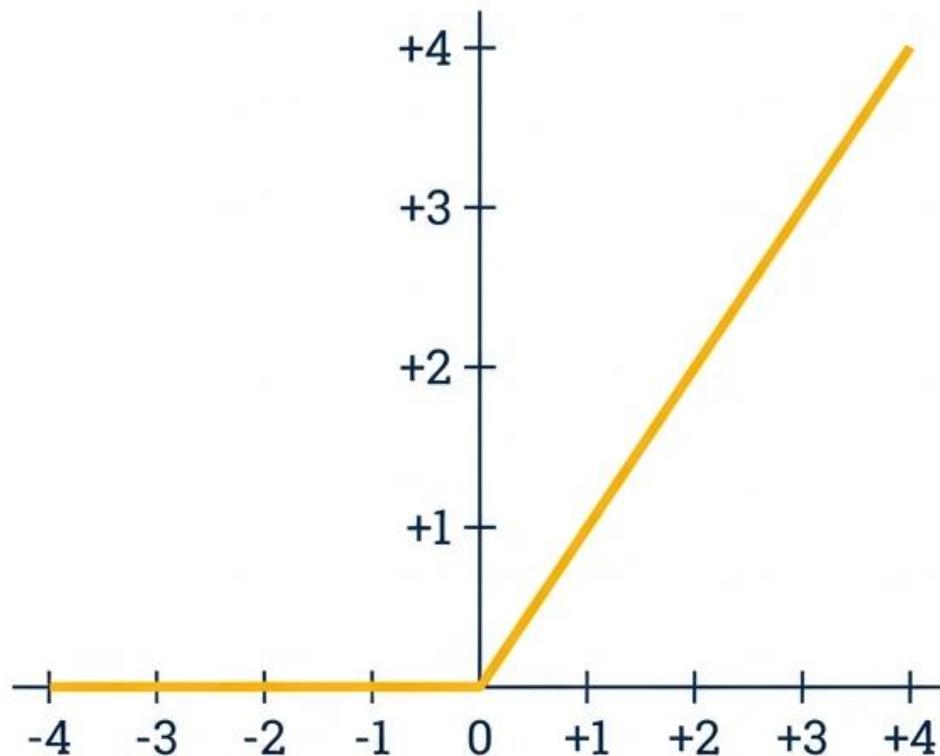


**Padding:** Adiciona zeros bordas para evitar que a imagem encolha a cada camada.

Zero Padding: Preserva as dimensões espaciais da imagem.

# Não-Linearidade: A Função ReLU

$$f(x) = \max(0, x)$$



Input

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

ReLU

Output

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

**ReLU (Rectified Linear Unit):** Remove a linearidade do sistema, permitindo que a rede aprenda fronteiras de decisão complexas.

**Lógica:** Se negativo, zera; se positivo, mantém.

# Pooling: Subamostragem e Eficiência

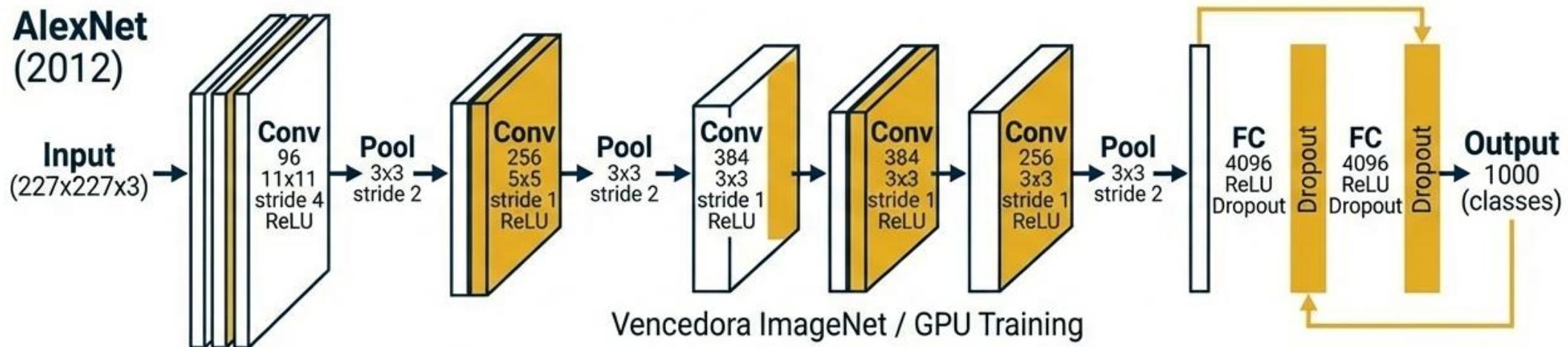
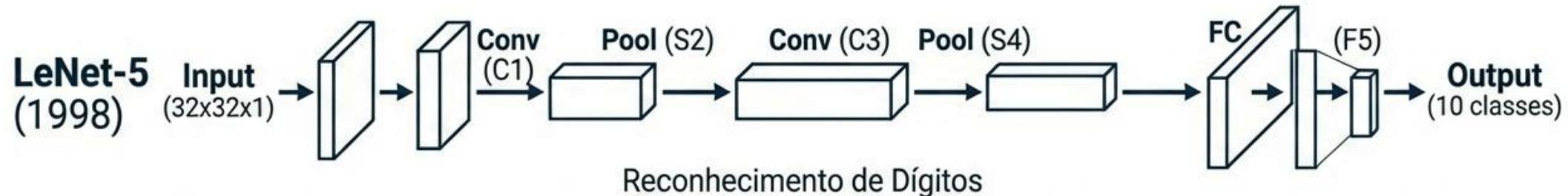
7	3	1	8
2	5	6	4
9	1	0	2
4	3	5	8

Max Pooling  
(2x2 window, Stride 2)

7	8
9	8

- **Max Pooling:** Seleciona o valor máximo dentro de uma janela.
- **Redução de Dimensionalidade:** Menos parâmetros e menor consumo de RAM.
- **Invariância Local:** Torna a rede robusta a pequenas distorções e translações.

# A Era Clássica: LeNet-5 e AlexNet

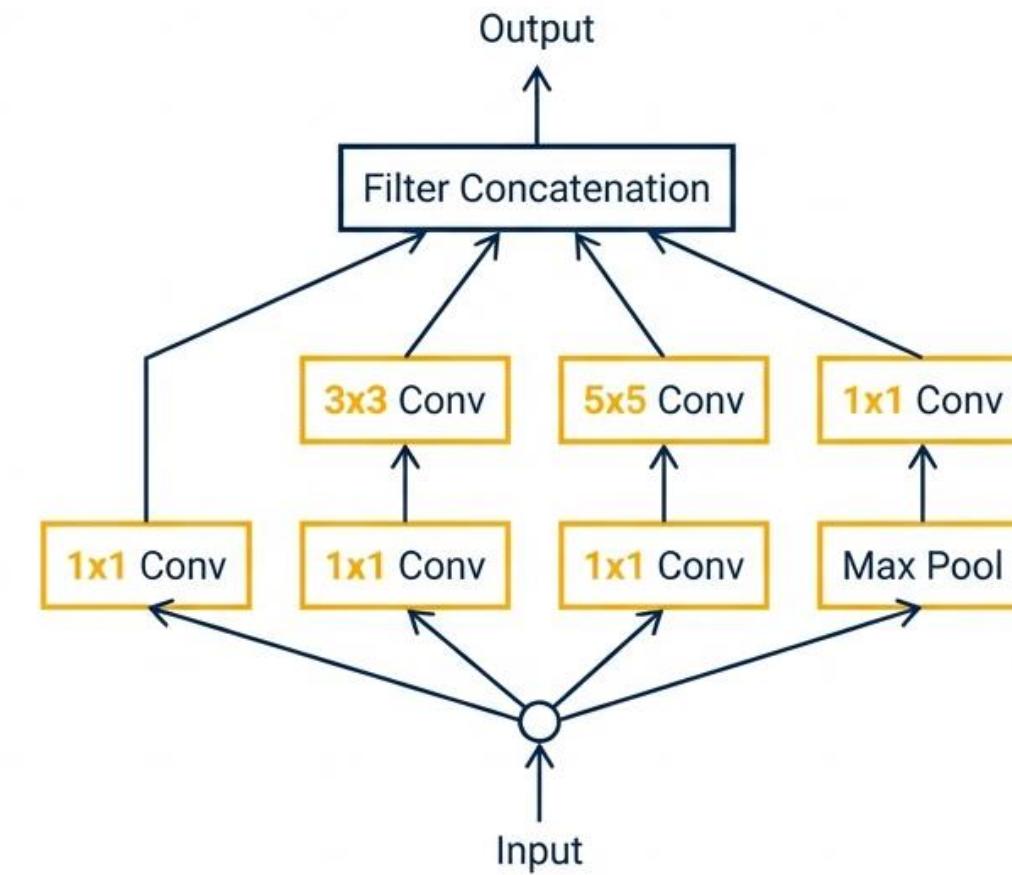


**LeNet-5:** A pioneira em leitura de cheques bancários.

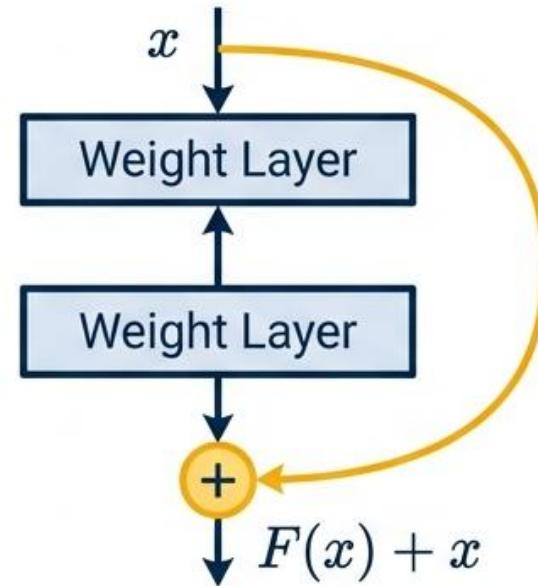
**AlexNet:** O marco do Deep Learning. Introduziu ReLU, Dropout e provou que redes profundas eram viáveis.

# Aprofundando a Arquitetura: VGG e GoogLeNet

- **VGGNet:** Padronizou o uso de filtros pequenos ( $3 \times 3$ ) em sequência.
- **GoogLeNet (Inception):** Processamento paralelo com múltiplos tamanhos de filtro. Usa convoluções  $1 \times 1$  como "gargalo" para eficiência.



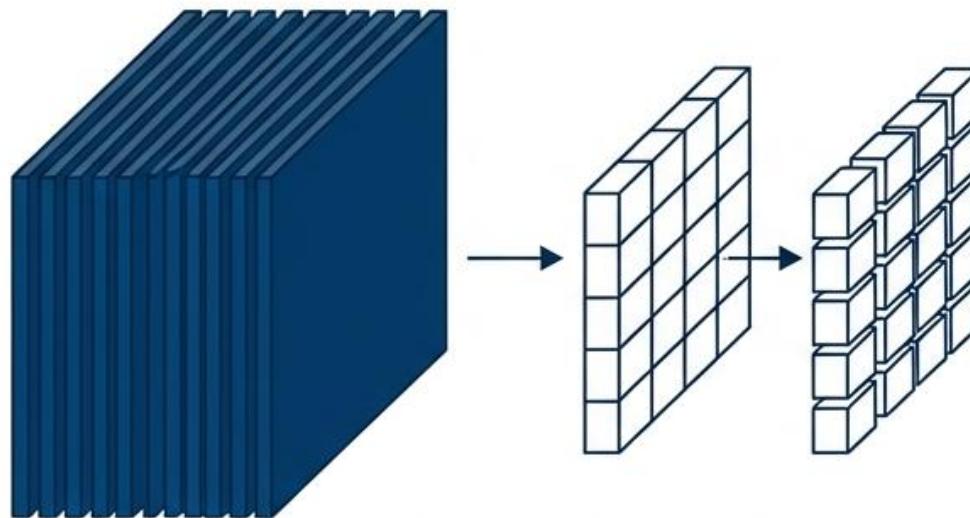
# A Revolução Residual: ResNet



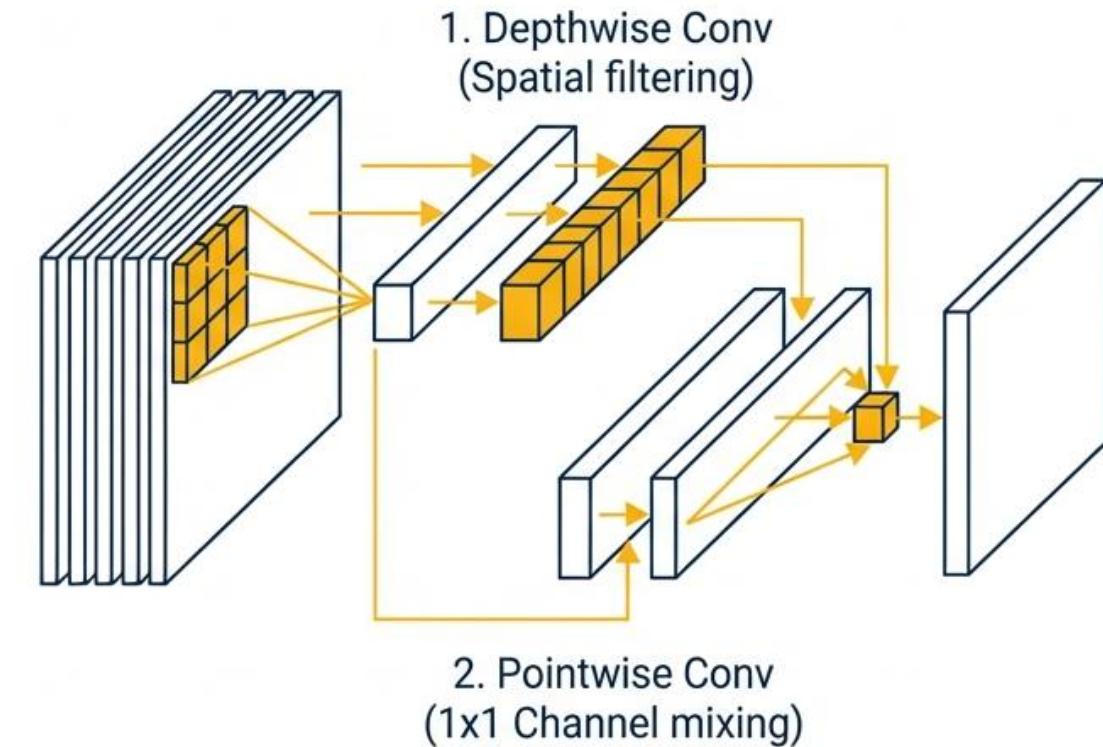
- **O Problema:** Redes muito profundas sofrem com o desaparecimento do gradiente (Vanishing Gradient).
- **A Solução:** Conexões de salto (Skip Connections) criam "atalhos" para o sinal, permitindo o treinamento de redes com mais de 100 camadas.

# Eficiência Computacional: Convoluçãoções Separáveis

Convolução Padrão



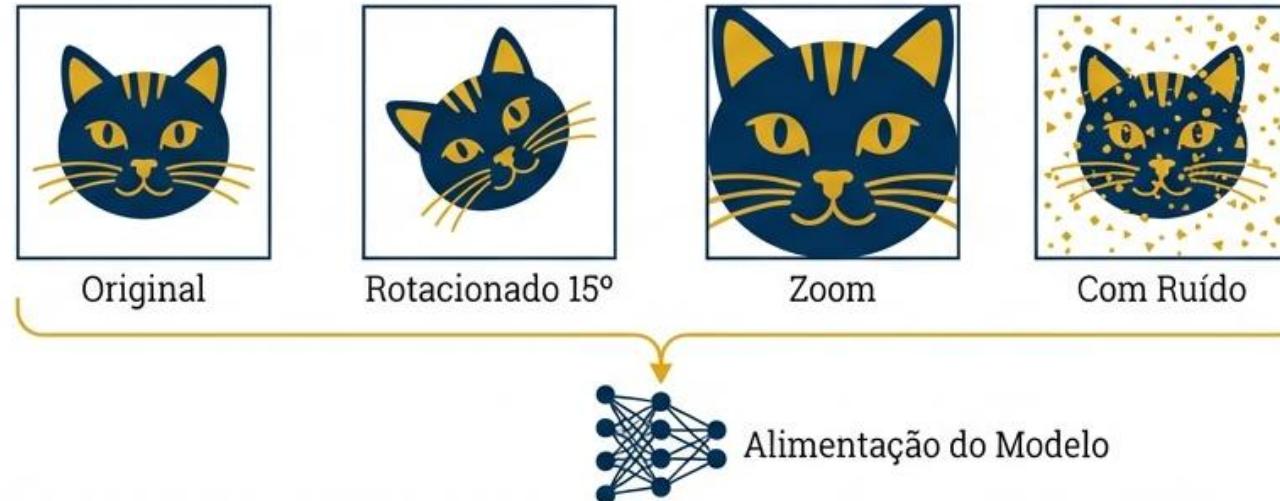
Convolução Separável (MobileNet)



- Separa o aprendizado espacial da mistura de canais.
- Reduz drasticamente o número de parâmetros, viabilizando IA em dispositivos móveis.

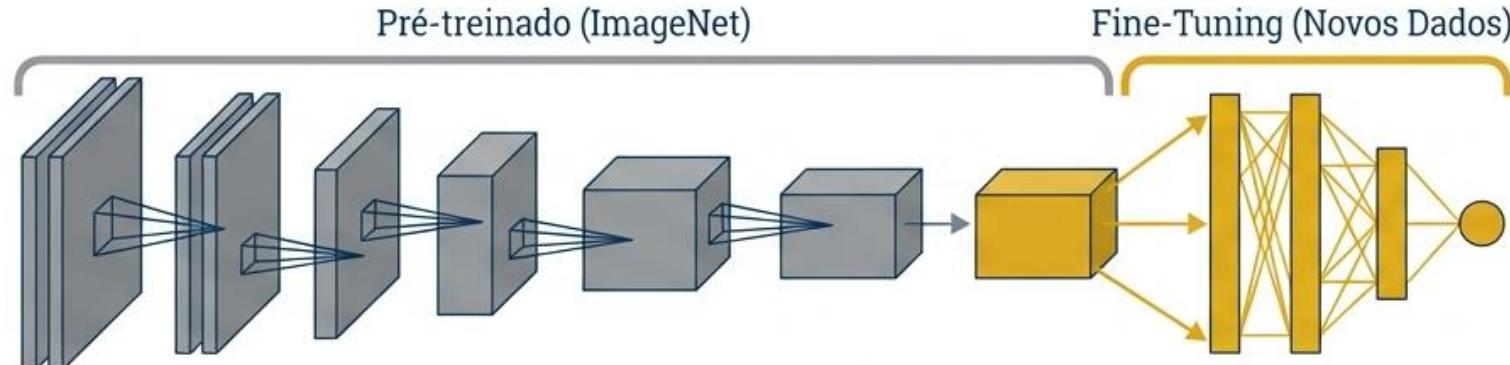
# Estratégias de Treinamento

## Data Augmentation



Multiplica os dados de treino aplicando transformações nas imagens originais para melhorar a generalização.

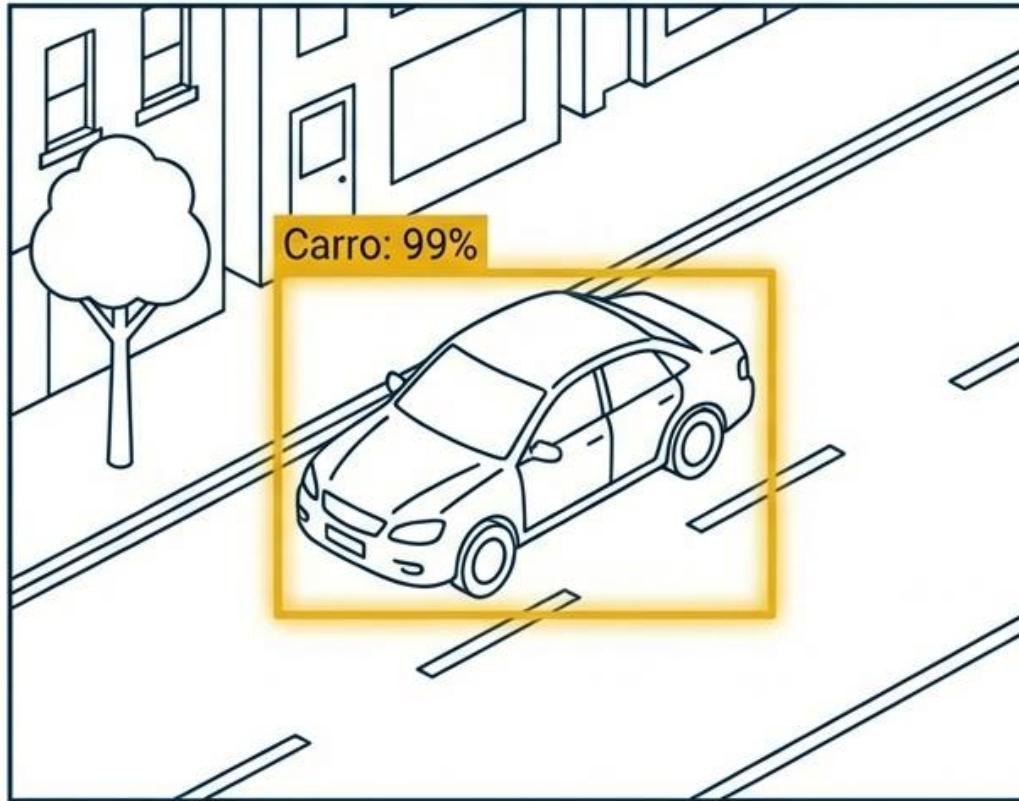
## Transfer Learning



Reutilizar o conhecimento visual de grandes datasets para resolver problemas com poucos dados.

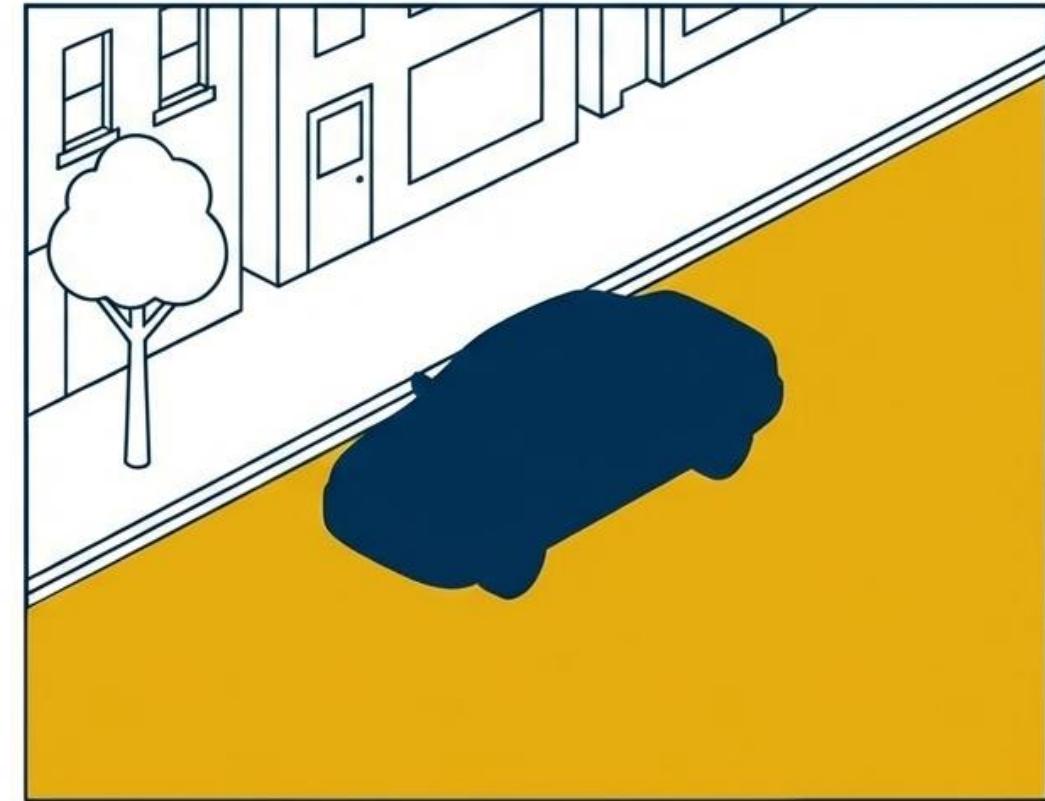
# Além da Classificação: Detecção e Segmentação

Detecção de Objetos (YOLO)



**Detecção:** Localiza objetos com Bounding Boxes em tempo real.

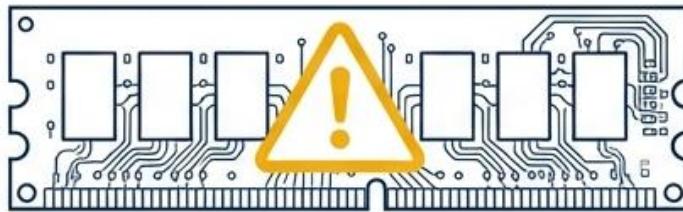
Segmentação Semântica (U-Net)



**Segmentação:** Classificação pixel a pixel, exigindo redes que recuperam a resolução espacial (Upsampling).

# Desafios Atuais e Fronteiras

## Custo de Memória



Backpropagation consome  
GBs de memória.

## Explicabilidade



A dificuldade de auditar  
decisões da IA.

## Ataques Adversariais



Panda + Ruído → Gibbon

Imagens com ruído projetado  
para enganar a rede.

O futuro aponta para Vision Transformers (ViT) e aprendizado auto-supervisionado.