INTRODUÇÃO AO *DEEP LEARNING*

152 minutos

Aula 1

FUNDAMENTOS DE *DEEP LEARNING*

O deep learning é uma categoria do machine learning que pode ser interpretada como uma evolução das redes neurais artificiais e inspiradas na profundidade arquitetônica do cérebro humano.

35 minutos

INTRODUÇÃO

Com o crescimento computacional explosivo das últimas décadas, a vastidão de dados que são gerados todos os dias se amplia exponencialmente. A necessidade de criar de modelos computacionais complexos capazes de extrair informações e de descrever esses dados tem sido o foco de pesquisas de vários cientistas durante anos. Diante desse contexto, surgiram as redes de aprendizado profundo, também chamadas de *deep learning*.

O *deep learning* é uma categoria do *machine learning* que pode ser interpretada como uma evolução das redes neurais artificiais e inspiradas na profundidade arquitetônica do cérebro humano. As técnicas de *deep learning* são responsáveis pelo crescimento atual da inteligência artificial, principalmente em aplicações com visão computacional, processamento de linguagem natural e reconhecimento de fala.

REDES NEURAIS DE APRENDIZAGEM PROFUNDA (*DEEP LEARNING*)

Com o avanço da computação, marcado principalmente pelo surgimento da Internet, o volume de dados gerados pela humanidade se tornou progressivamente maior. Assim, surgem novos conceitos e tecnologias com o propósito de processar esses dados e extrair informações relevantes para diversas aplicações. Sempre que você acessa um site, realiza uma compra ou até mesmo interage nas redes sociais, dados são criados. Diante de um mundo cada vez mais conectado, do aumento do uso de redes sociais, de objetos com acesso à internet, conceitos como Internet das Coisas e Big Data têm sido profundamente estudados.

Paralelamente a essa evolução tecnológica, pesquisas voltadas ao aprendizado de máquina e à inteligência artificial também têm crescido, eliminando ou reduzindo a necessidade de intervenção humana. Isso se dá pelo fato de o volume alto de dados só se tornar útil à medida que as informações possam ser efetivamente analisadas, correlacionadas e transformadas em tomadas de decisão. Para isso, faz-se necessário um sistema que consiga processar e agregar valor a esses dados. A necessidade de criar modelos capazes de operar com uma grande quantidade de dados e com o menor trabalho possível fez com que pesquisadores desenvolvessem arquiteturas de rede neural artificial que ficaram conhecidas como *deep learning*.

Deep learning é uma técnica de aprendizado de máquina que se baseia no sistema nervoso dos seres vivos e em sua capacidade de processar informações para construir sistemas de inteligência artificial. Pode ser interpretada como uma forma de extrair padrões úteis a partir dos dados informados, de uma maneira automatizada e com o menor trabalho possível (SCHMIDHUBER, 2015). Existe uma grande variedade de redes *deep learning*, diferenciadas, principalmente, pela arquitetura e pela aplicação, que podem ser utilizadas em problemas de classificação ou regressão. Em geral, essas arquiteturas se baseiam em três principais tipos de camadas:

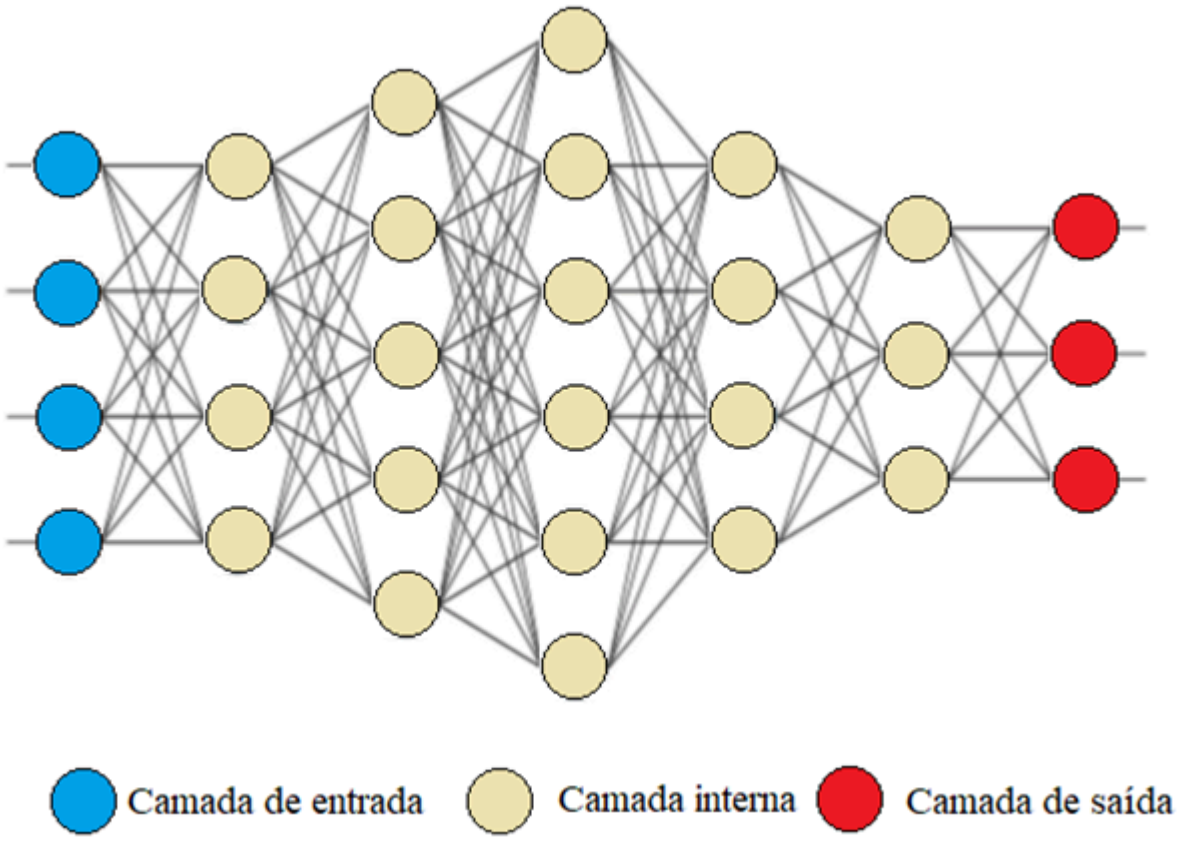
1. Camada de entrada.

2. Camada interna ou oculta.

3. Camada de saída.

Observe um exemplo na Figura 1, a seguir.

Figura 1 | Rede neural *deep learning*



Fonte: elaborada pelo autor.

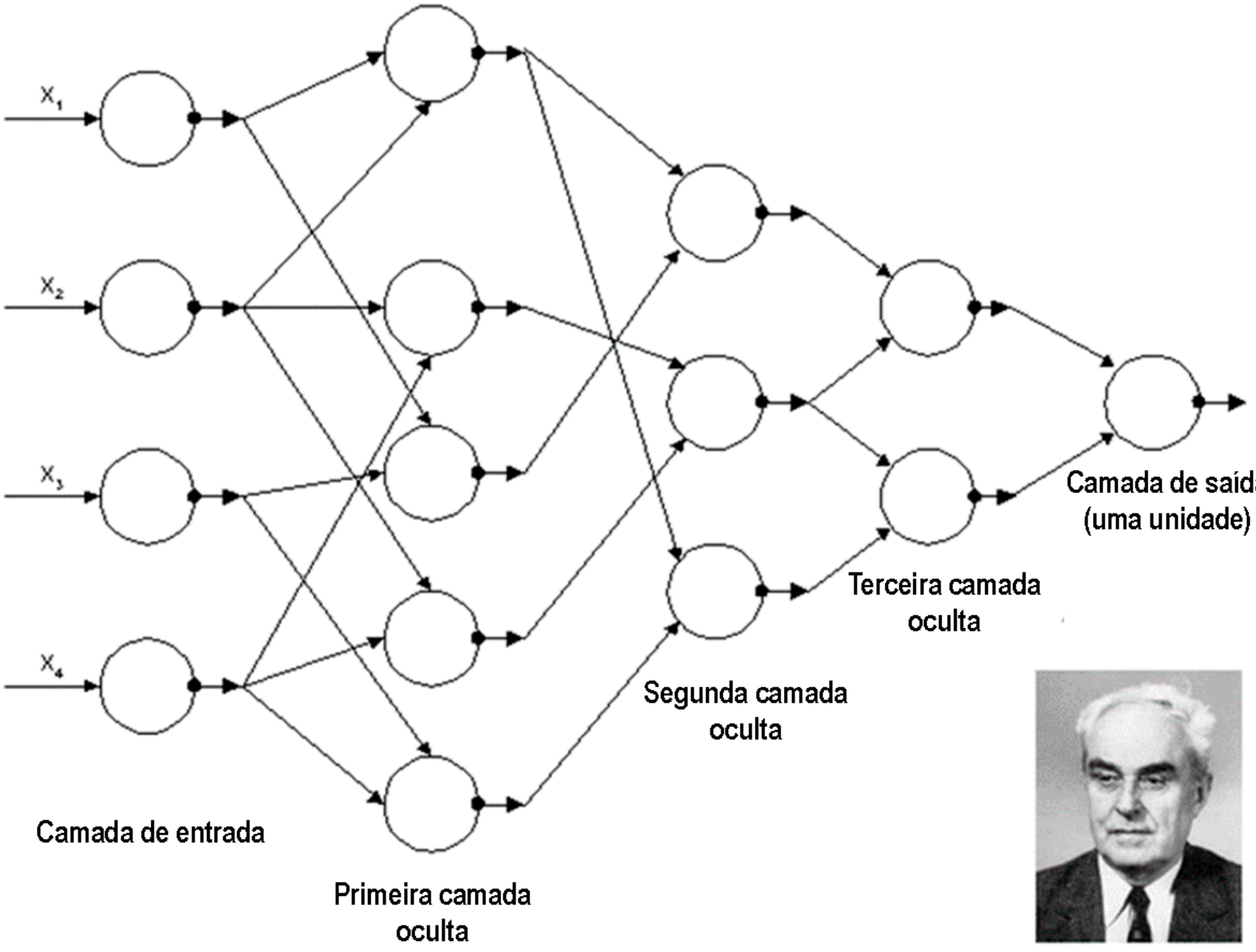
Essas redes neurais são conhecidas como redes de aprendizado profundo por possuírem várias camadas de neurônios projetadas para realizar análises complexas de grandes quantidades de dados. O *deep learning* é uma subclasse do aprendizado de máquina que se refere a uma classe ampla de técnicas e arquiteturas de inteligência artificial, cuja principal diferenciação está no uso de múltiplas camadas de processamento de informações não lineares de natureza hierárquica (DENG; YU, 2013). Ou seja, o que diferencia o *deep learning* das outras redes neurais é o número de camadas. Enquanto a maioria das redes neurais costuma ter de cinco a dez camadas de neurônios, muitas das arquiteturas modernas conhecidas como *deep learning* têm de 50 a 100 camadas de profundidade.

O *deep learning* tem se tornado uma área de pesquisa extremamente ativa e essencial para o desenvolvimento do aprendizado de máquina moderno. Essas técnicas de aprendizado profundo são responsáveis pelo avanço em aplicações de reconhecimento de padrões, tais como compreender a fala humana ou reconhecer objetos visualmente.

CLASSES DE *DEEP LEARNING*

Até pouco tempo, a maioria das técnicas de aprendizado de máquina explorava arquiteturas superficiais, que normalmente continham no máximo uma ou duas camadas de transformações de recursos não lineares. Embora arquiteturas rasas tenham se mostrado eficazes em muitas aplicações simples de classificação e regressão, sua modelagem e poder de representação apresenta limitações ao lidar com aplicações mais complexas do mundo real, como fala humana, som e linguagem natural, visão computacional (DENG; YU, 2013). A primeira arquitetura conhecida como *deep learning* que possuía múltiplas camadas de características não lineares (Figura 2) foi apresentada por Alexey Grigorevich Ivakhnenko e Valentin Grigor’evich Lapa em 1965. Nesse modelo, em cada camada eram selecionadas as melhores características por meio de métodos estatísticos, as quais eram encaminhadas para a próxima camada.

Figura 2 | Arquitetura da primeira rede profunda apresentada por Alexey Grigorevich Ivakhnenko em 1965



Fonte: adaptada de Data Science Academy ([s. d.]).

Historicamente o conceito de *deep learning* vem da evolução das pesquisas de redes neurais artificiais, sendo inspirado na profundidade arquitetônica do cérebro. Embora o primeiro modelo conhecido de rede profunda date da década de 1960, por muitos anos pesquisadores de redes neurais quiseram treinar redes multicamadas profundas, relatando resultados experimentais positivos em redes de dois a três níveis, mas, à medida que se tornavam mais profundas, os resultados ficavam mais pobres (BENGIO, 2009). O primeiro relato de sucesso remonta ao ano de 2006, quando foi desenvolvido um algoritmo de aprendizado que treinava avidamente uma camada de cada vez, explorando um algoritmo de aprendizado não supervisionado para cada camada (HINTON; OSINDERO; TEH, 2006). Com o avanço computacional, principalmente dos processadores, o aumento do uso de *deep learning* em aplicações têm se tornado cada vez mais frequente. A partir disso, têm surgido várias arquiteturas destinadas a diversas aplicações.

Dependendo de como as arquiteturas e técnicas são aplicadas, seja para reconhecimento/classificação ou síntese/geração, é possível categorizar os *deep learning* em três classes principais:

1. Aprendizado supervisionado ou de classificação.

2. Aprendizado não supervisionado ou generativo.

3. Híbrido.

As redes *deep learning* de aprendizado supervisionado, também conhecidas como redes profundas discriminativas, fornecem diretamente poder discriminativo para fins de classificação de padrões a partir de características rotuladas previamente apresentadas. Os dados de rótulo de destino estão sempre disponíveis em formas diretas ou indiretas para esse aprendizado supervisionado. Assim, a cada conjunto de dados fornecido, o sistema realizará a previsão das classes discretas predefinidas ou a previsão de um valor numérico contínuo. Existem diversos algoritmos de *deep learning* que utilizam aprendizado supervisionado.

Os algoritmos de rede profunda para aprendizado não supervisionado são empregados em problemas nos quais não são usadas informações de supervisão específicas da tarefa (por exemplo, rótulos de classe-alvo) no processo de aprendizagem. Nesse método, é possível aprender a prever percepções futuras a partir de percepções anteriores. O sistema utiliza o aprendizado para agrupar conjuntos semelhantes em determinadas classes ou detectar associações entre as características.

As redes profundas híbridas, também conhecidas como de aprendizado semissupervisionado, fazem uso do modelo generativo e discriminativo, ou seja, utilizam técnicas supervisionadas e não supervisionadas. O treinamento ocorre em duas etapas. Primeiro, o algoritmo gera um modelo contendo suposições iniciais a partir dos rótulos conhecidos e, por fim, usa os dados não rotulados para ajustar essas suposições.]

DEEP LEARNING NOS DIAS ATUAIS

Apenas há alguns anos, supunha-se que o *machine learning*, assim como o *deep learning*, só pudesse ser executado em máquinas de ponta, mas, graças aos intensos esforços de pesquisas desenvolvidas pela indústria e pela academia, esse cenário mudou drasticamente (LEVY, 2020). Com o advento de processadores cada vez mais potentes e a possibilidade de utilizar inteligência artificial em sistemas embarcados, o interesse por técnicas de *deep learning* tem disparado. A atual grande expansão da aplicabilidade de *machine learning*, assim como o interesse por ela, está relacionada aos avanços nas técnicas de *deep learning* obtidos na última década (BREUEL, 2020).

Aplicações como carros autônomos, diagnósticos de cânceres e autismo, e a criação de imagens a partir de simples descrições têm tornado o aprendizado profundo cada vez mais popular. Revistas como a *Science*, *Nature*, *Nature Methods*, *Forbes*, entre outras, têm rotineiramente apresentado matérias sobre pesquisas de *deep learning* (DATA SCIENCE ACADEMY, [s. d.]). Ao longo dos últimos anos, uma série de *workshops*, tutoriais e edições ou sessões especiais de conferências foram dedicadas exclusivamente a aplicações com aprendizado profundo (DENG; YU, 2013). Em um estudo conduzido pela *Lux Research*, o *deep learning* ficou em segundo lugar entre cerca de 2.700 tecnologias desenvolvidas para as áreas de assistência médica, materiais, energia e transformação digital numa análise de dados de inovação baseada em aprendizado de máquina (RAMANATHAN, 2018).

Atualmente, as redes neurais artificiais, assim como as redes de aprendizado profundo, não são mais consideradas um campo de pesquisa independente e têm se tornado parte de integrantes de vários outros campos de pesquisa. Desde 2006, técnicas de aprendizado profundo têm sido aplicadas com sucesso não somente em problemas de reconhecimento ou classificação, mas também em regressão, redução de

dimensionalidade, modelagem de textura e movimento, segmentação de objetos, recuperação de informação, robótica, processamento de linguagem natural e filtragem colaborativa (BENGIO, 2009). No momento atual, o *deep learning* é essencial ao processamento de Big Data e à evolução da inteligência artificial.

O *deep learning* tem se difundido em diversos campos que envolvem o reconhecimento de imagens, como no reconhecimento facial utilizado pelo Facebook, na identificação de locais pelo Google, na direção autônoma utilizada pelos carros autônomos da Tesla e do Google, em controle de qualidade de fabricação, na geração de imagens médicas e em vários outros setores (VALDATI, 2020). Além disso, está sendo usada em aplicativos de segurança cibernética, reconhecimento de voz, design de medicamentos e de chips, e na otimização das operações de fabricação (RAMANATHAN, 2018). Com a evolução dessa tecnologia, é possível desenvolver muitos outros sistemas inteligentes e se aproximar ainda mais da criação de uma inteligência artificial totalmente autônoma, o que terá impacto sobre todos os segmentos da sociedade.

Diante disso, o potencial para profissionais especializados nessa área é enorme, fato que se reflete no aumento de vagas de emprego e de cursos de inteligência artificial. Serviços on-line, marketing e serviços financeiros são alguns dos mercados que mais utilizam aplicações com *machine learning* hoje em dia (PICCOLO, 2020). Trata-se de um mercado novo e dinâmico, e a tendência de crescimento do *deep learning* o torna uma área bastante atraente e promissora para estudantes de computação que pretendem ingressar no mercado de trabalho do futuro.

VÍDEO RESUMO

Olá, estudante! No vídeo a seguir, você recordará quais são as principais características do *deep learning* e compreenderá por que é essencial para um futuro profissional em *machine learning* dominar as técnicas de aprendizado profundo. Além disso, será possível conhecer algumas aplicações atuais de inteligência artificial que fazem uso dessas técnicas de *deep learning*.

Para visualizar o objeto, acesse seu material digital.

Saiba mais

No artigo *Afinal, o que é deep learning?*, você terá acesso a mais informações relacionadas ao *deep learning*, entendendo o que o diferencia de outras técnicas similares e a relação desse conceito com o mercado de trabalho.

O artigo *5 passos para criar seu 1º projeto de deep learning com Python e Keras* ensina a criar um projeto de *deep learning* na linguagem Python usando a biblioteca Keras. Vale a pena conferir!

REDES NEURAIS ARTIFICIAIS

As redes neurais artificiais (RNAs) são modelos matemáticos que têm capacidade computacional adquirida por meio de aprendizagem e generalização (BRAGA; LUDERMIR; CARVALHO, 2000).

33 minutos

INTRODUÇÃO

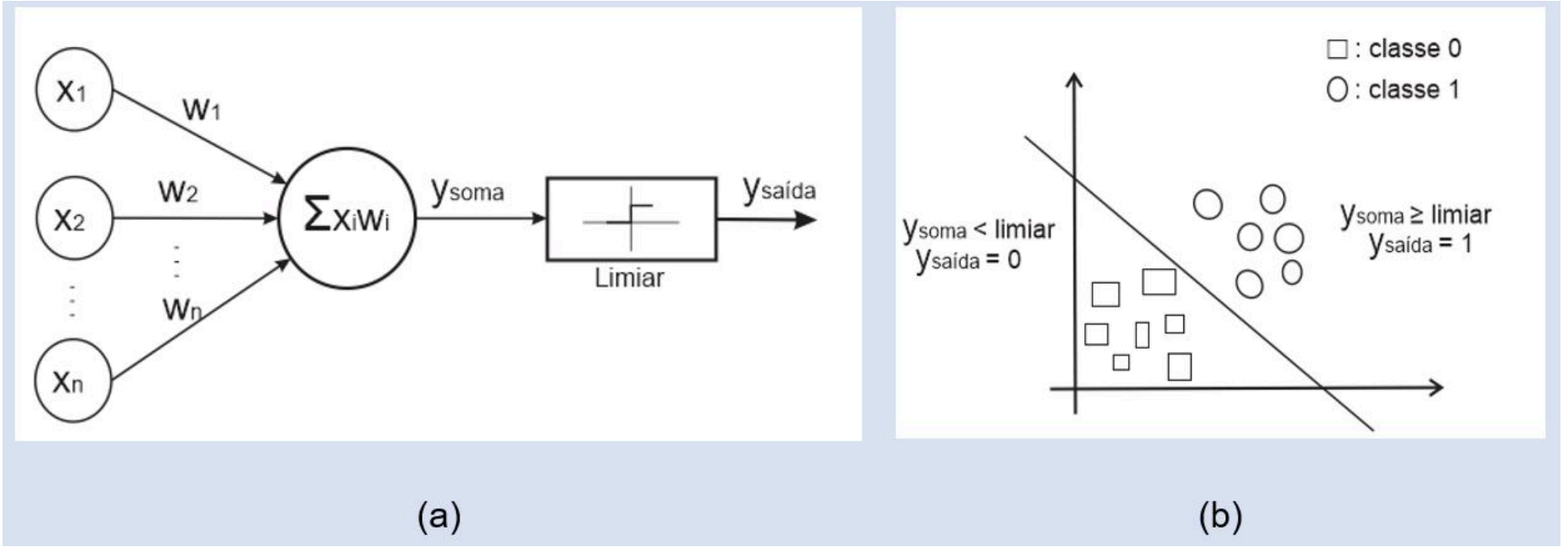
As redes neurais artificiais (RNAs) são modelos matemáticos que têm capacidade computacional adquirida por meio de aprendizagem e generalização (BRAGA; LUDERMIR; CARVALHO, 2000). O perceptron é considerado o primeiro algoritmo de rede neural. A arquitetura limitada do perceptron permite que a rede seja apenas útil para aplicações binárias (duas classes) e linearmente separáveis. Entretanto, muitos dos problemas de reconhecimento de padrões reais não possuem apenas duas classes. Além disso, mesmo problemas binários podem ser não lineares, como a porta lógica XOR. Em virtude disso, há a necessidade de utilizar RNAs com arquiteturas mais complexas. Um exemplo de classificador não linear bastante conhecido e utilizado é o perceptron multicamadas (MLP). Diferentemente do perceptron simples, no qual existe apenas um neurônio, o MLP pode relacionar o conhecimento a vários neurônios de saída, o que o torna útil para problemas multiclasses.

A PRIMEIRA REDE NEURAL ARTIFICIAL

O cérebro humano é um dispositivo poderoso e complexo, capaz de processar informações e de tomar decisões em questão de segundos. Em conjunto com o corpo, o ser humano pode aprender a andar, falar, criar e realizar inumeráveis atividades. Utilizar máquinas para replicar funções humanas é um sonho antigo, contudo a capacidade que um computador tem de reproduzi-las pode ser bem limitada. Baseando-se no funcionamento das redes neurais biológicas, principalmente quanto ao processo de aprendizagem por experiência, diversos pesquisadores têm se dedicado a criar sistemas capazes de simular a habilidade humana de processar informações. A partir dessas pesquisas, surgiram modelos que ficaram conhecidos como redes neurais artificiais (RNAs).

As RNAs são sistemas computacionais, inspirados pelas redes neurais biológicas que constituem cérebros humanos, formados por nós conectados em uma única direção, criando uma rede na qual os sinais de entrada são processados gerando um sinal de saída (GERVEN; BOHTE, 2017). O primeiro modelo matemático conhecido baseado no neurônio biológico é o neurônio booleano de McCulloch-Pitts, ou neurônio MP, proposto por McCulloch e Pitts em 1943. O neurônio MP (Figura 1a) era capaz de receber vários valores booleanos (0 ou 1) de entrada, processar esses dados atribuindo pesos a eles, comparar o resultado com um limiar preestabelecido e gerar um valor booleano de saída. O limiar funciona como uma função de ativação do tipo degrau, ou seja, valores abaixo do limiar são atribuídos a uma classe, e acima do limiar, a outra classe. Assim, esse modelo poderia ser usado para separar duas classes que estejam bem isoladas uma da outra (Figura 1b).

Figura 1 | a. Arquitetura do neurônio MP; b. Classificação pelo neurônio MP



Fonte: elaborada pelo autor.

Inspirado pelo modelo matemático desenvolvido por McCulloch e Pitts, Frank Rosenblatt, em 1958, propôs a rede perceptron. Esse modelo utiliza o princípio do neurônio MP, entradas multiplicadas por pesos e uma regra de aprendizagem, o que torna essa rede um dispositivo inteligente e, por isso, considerada o primeiro algoritmo de rede neural artificial. O processo de aprendizagem consistia na modificação dos pesos até que se resolvesse o problema ou que o período de aprendizagem acabasse, sendo, desse modo, capaz de atuar como classificador binário e realizar aprendizado supervisionado. Dado um vetor de entrada *x* e um vetor de pesos *w*, a regra de aprendizagem é determinada pela seguinte equação:

$$w(t+1) = w(t) + \eta e(t) x(t)$$

em que t indica o instante de apresentação do vetor de entrada e w(t+1), o peso atualizado. e(t) representa o erro entre a saída desejada d e a saída garantida pela rede y, obtido por:

$$e = d - y$$

A fim de tornar o processo de ajuste de pesos mais estável, é comum utilizar o fator η, chamado de passo de aprendizagem, em que $0 < \eta \ll 1$. A seguir o pseudocódigo de um perceptron.

```
Algoritmo: Rede Perceptron

1.  Início (t = 0)

1.1.  Definir valor de η entre 0 e 1.

1.2.  Iniciar w(0) com valores nulos ou aleatórios.

2.  Funcionamento

2.1.  Selecionar vetor de entrada x(t).

2.2.  Calcular o y soma.
```

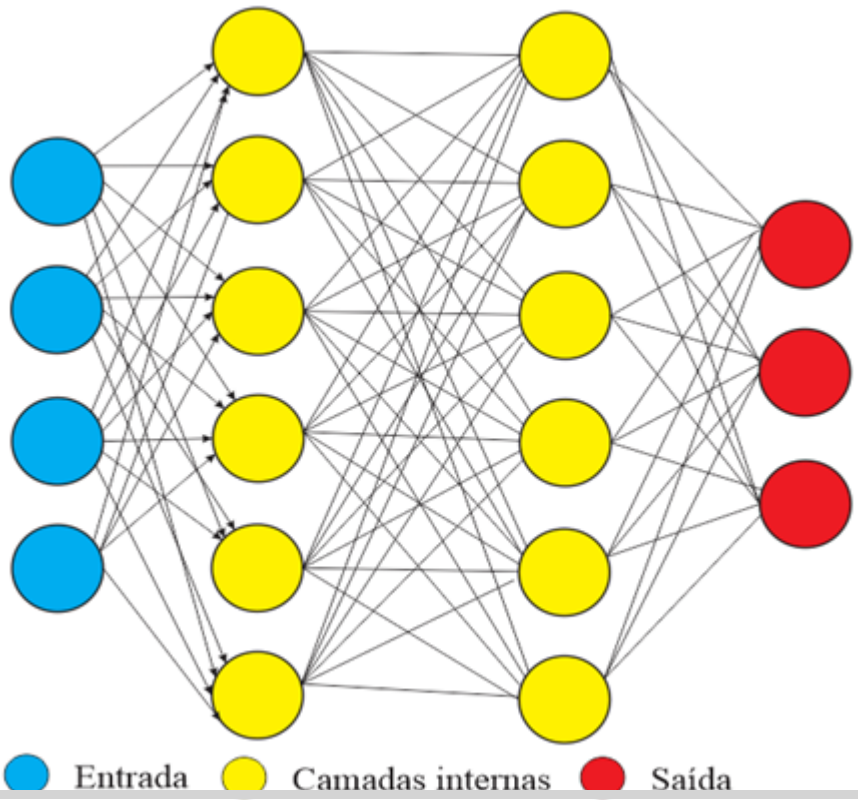
```
2.3.  Calcular o y saída.

3.  Treinamento

3.1.  Calcular erro: e(t) = d(t) - y saída (t).

3.2.  Ajustar pesos via regra de aprendizagem.

3.3.  Verificar critério de parada:

3.4.  Se atendido, finalizar o treinamento.

3.5.  Caso contrário, fazer t=t+1 e ir para o passo 2.
```

A rede perceptron é útil para classificar problemas lineares, nos quais, com apenas uma reta, é possível separar duas classes, como as portas lógicas AND, OR e NOT. Contudo, por possuir somente um neurônio, o perceptron não é capaz de classificar problemas não linearmente separáveis, como a porta lógica XOR. Diante dessa limitação, surgiram pesquisas inéditas voltadas para novas estruturas utilizando mais camadas de neurônios. Essas redes ficaram conhecidas como redes neurais multicamadas, ou perceptron de multicamadas (MLP, do inglês *multilayer perceptron*).

ELEMENTOS DAS REDES NEURAIS MULTICAMADAS

As redes neurais multicamadas se diferenciam das redes de camada simples pelo número de camadas intermediárias entre a camada de entrada e a camada de saída (AFFONSO et al., 2010). Foi possível notar, na Figura 1a, que o perceptron possui apenas entradas, pesos e uma saída. Enquanto isso, um perceptron de multicamadas (MLP) possui, além dos elementos citados anteriormente, uma ou mais camadas internas compostas por neurônios computacionais, também chamados de neurônios ocultos. A estrutura de uma MLP pode ser observada na Figura 2, a seguir.

Figura 2 | Arquitetura de uma rede neural perceptron multicamadas



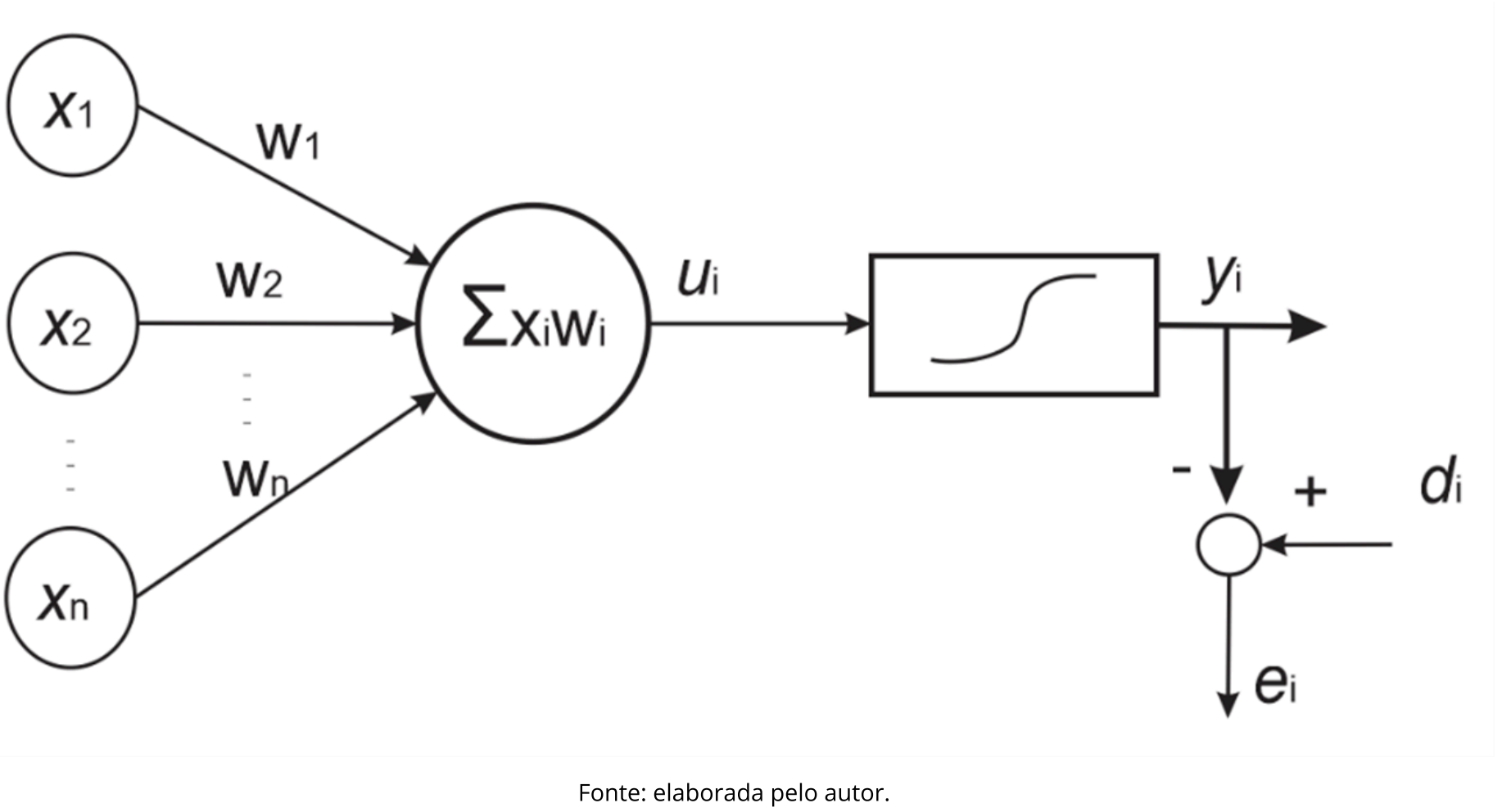
Fonte: elaborada pelo autor.

A camada de entrada é uma camada não computacional, na qual não há processamento, responsável pela recepção e propagação das informações dos dados de entrada para a camada seguinte. O tamanho da entrada é definido pela quantidade de dados informados. As camadas internas contêm os neurônios responsáveis pelo processamento não linear dos dados a partir de conexões entre a entrada e a saída. Nessas conexões, há pesos que são multiplicados pela entrada, garantindo, assim, o conhecimento da rede (SILVA, 2009). Por fim, após o processamento realizado pelas camadas internas, é gerada pelos neurônios a camada de saída. O tamanho da saída é determinado pelo número de classes do problema. A quantidade de camadas internas e de neurônios em cada camada é determinada de forma empírica. O software Weka recomenda como cálculo do número de neurônios da camada interna a seguinte equação:

$$H = (A + C)/2$$

sendo a quantidade de neurônios da camada oculta, A a quantidade de atributos de entrada e C a quantidade de saídas da rede. Um neurônio qualquer da rede MLP, seja oculto ou de saída, é representado genericamente como mostra a Figura 3, a seguir.

Figura 3 | Representação de um neurônio da rede MLP



Fonte: elaborada pelo autor.

As funções de ativação são extremamente importantes para as RNAs, pois são responsáveis por decidir se um neurônio será ativado ou não, isto é, se a informação é relevante ou deve ser ignorada (DATA SCIENCE ACADEMY, 2022). Observa-se que, diferentemente do perceptron que utilizava uma função de ativação do tipo degrau, na rede MLP é usada uma função sigmoide. Ou seja, deixa de ser uma variável binária (0 ou 1) e passa a ser uma variável analógica, que assume qualquer valor entre 0 e 1. Assim, a saída dos neurônios da camada interna e da camada de saída é dada pela função de ativação, que é escolhida empiricamente. Uma das funções comumente utilizadas na MLP é a sigmoide logística, expressa por:

$$y_i\left(t\right) = \frac{1}{1+\exp(-u_i(t))}$$

em que $y_i(t) \in (0,1)$.

A ativação do neurônio da camada interna e o cálculo do erro associado ao neurônio da camada de saída é realizado assim como no perceptron. Nas redes multicamadas, o erro dos neurônios ocultos é obtido a partir dos neurônios de saída por meio da retropropagação dos erros (*error backpropagation*), que ocorre no sentido inverso ao fluxo das informações. Dessa forma, a regra de aprendizagem utilizada no perceptron pode ser atualizada para

$$m(t+1) = m(t) + \eta e(t) y'(t) z(t)$$

em que z(t) é o vetor de entrada da camada de saída e y'(t), para a sigmoide logística, é expresso por:

$$y'(t) = y(t)[1 - y(y)]$$

A seguir o pseudocódigo do algoritmo de uma rede neural MLP.

A seguir o pseudocódigo do algoritmo de uma rede neural MLP.

Algoritmo: Rede Perceptron Multicamadas

Passo 0: Inicialize os pesos (valores aleatórios pequenos).

Passo 1: Enquanto a condição de parada é falsa, execute os passos 2-9.

Passo 2: Para cada par de treinamento, execute os passos 3-8.

Primeira fase (*feedforward*)

Passo 3: Cada sinal de entrada é transferido para todas as unidades na camada interna.

Passo 4: Cada neurônio das camadas internas soma suas entradas multiplicadas pelo peso, aplica a função de ativação para computar seu sinal de saída e envia o sinal para todas as unidades na camada de saída.

Passo 5: Cada neurônio de saída (y_k) soma suas entradas multiplicadas pelo peso e aplica sua função de ativação para computar seu sinal de saída.

Segunda fase (*error backpropagation*)

Passo 6: Cada neurônio de saída (y_k) computa seu termo de erro de informação e envia o erro para os neurônios da camada anterior.

Passo 7: Cada neurônio de saída (z_k) soma suas entradas, multiplica pela derivada de sua função de ativação para calcular seu termo de erro de informação.

Terceira fase (atualização dos pesos)

Passo 8: Cada neurônio de saída e da camada interna atualiza seus pesos.

Passo 9: teste a condição de parada.

CLASSIFICAÇÃO POR REDE NEURAL ARTIFICIAL

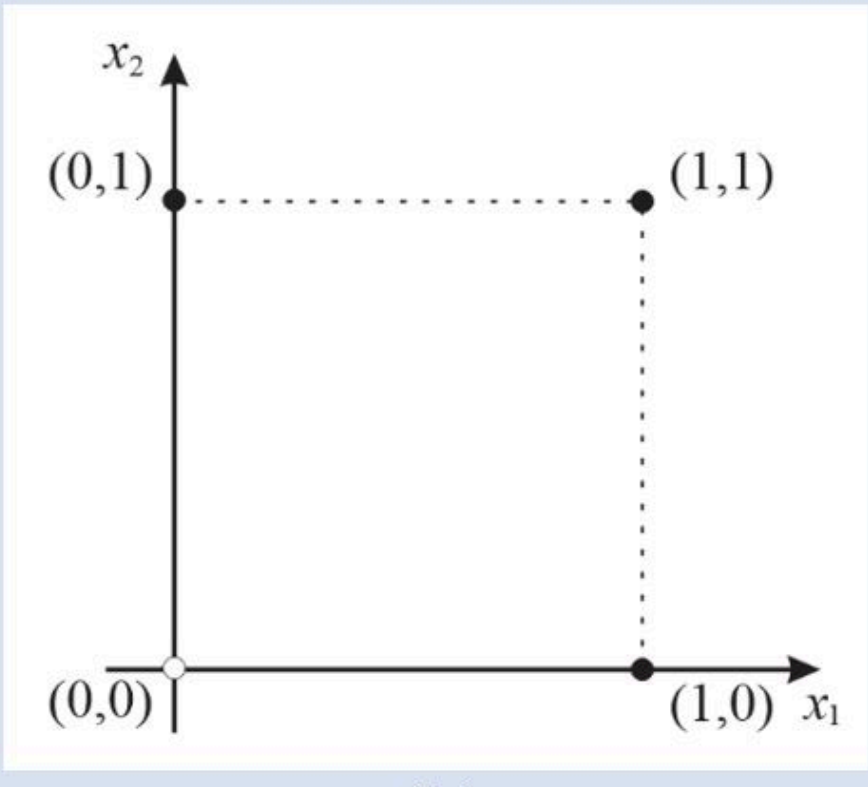
As redes neurais artificiais (RNAs) foram criadas com o propósito de resolver problemas computacionais em áreas como reconhecimento e classificação de padrões, predição de séries, entre outras esferas. Existem diversos modelos para a implementação de uma estrutura de RNA, como *self-organizing map* (SOM), *least mean square* (LMS), *radial basis function* (RBF), perceptron simples e de multicamadas (MLP), e, dependendo de sua arquitetura, podem ser usados em problemas de generalização e/ou classificação.

Considere a representação da porta lógica OR na Figura 6, a seguir. Nota-se que esse problema é linearmente separável, portanto é possível encontrar uma reta capaz de separar os pontos da classe 0 (y = 0) e os pontos da classe 1 (y = 1). Na verdade, podem-se encontrar infinitas retas que separam as duas classes. Implementando o algoritmo da rede perceptron, é possível encontrar os valores para os pesos de maneira que seja capaz de separar as duas classes. A Figura 7 representa um neurônio que implementa a porta OR.

Figura 6 | a. Tabela-verdade da porta lógica OR; b. Representação gráfica da porta lógica OR

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

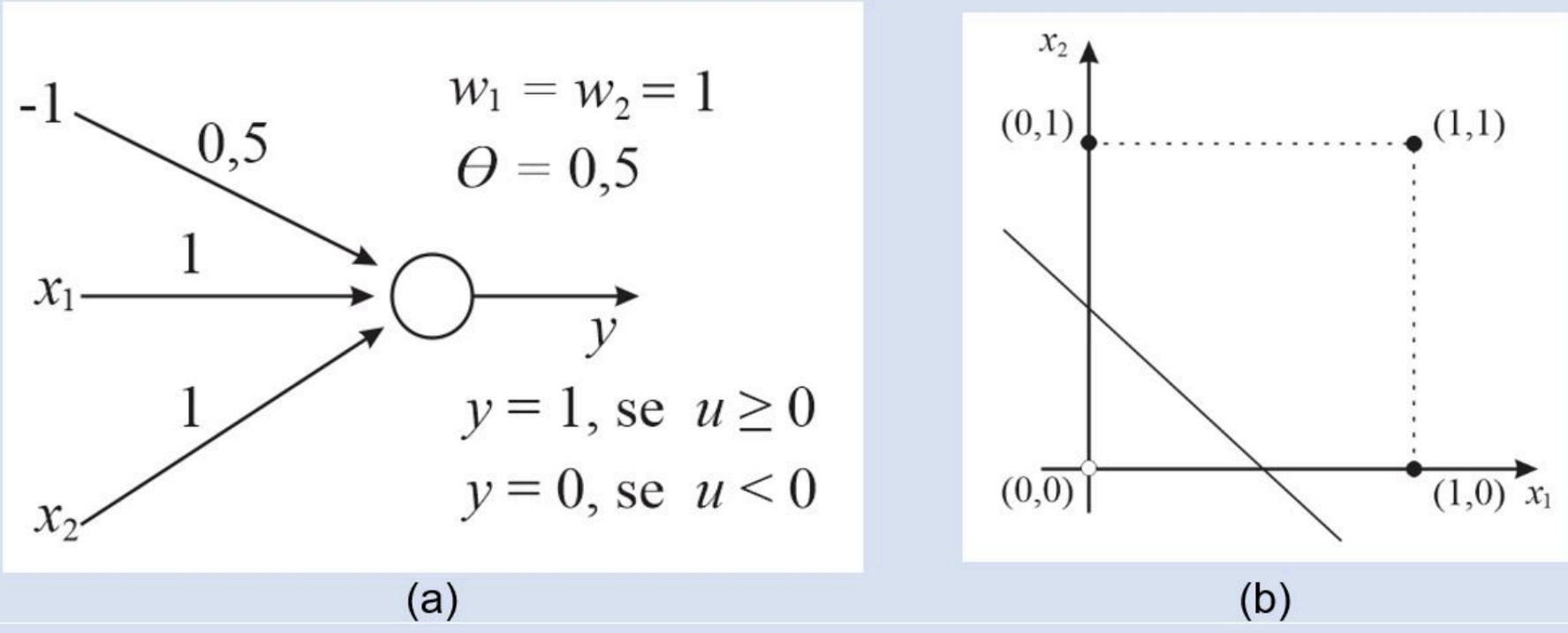
(a)



(b)

Fonte: elaborada pelo autor.

Figura 7 | a. Neurônio que implementa a porta OR; b. Representação gráfica da classificação da porta OR



Fonte: elaborada pelo autor.

Em problemas com mais de duas classes, classificadores lineares não são eficientes, portanto utilizam-se classificadores não lineares, como a rede MLP. Um exemplo de aplicação é o uso de uma rede MLP para separar as três classes de flores *Iris* (Figura 8). O *data set* da *Iris* possui quatro dados de informação sobre cada amostra das flores, logo, a camada de entrada tem quatro unidades. Como a classificação é em três classes, a camada de saída possui três neurônios. A quantidade de neurônios na camada interna deve ser determinada de forma empírica, ou seja, com base em testes, a fim de obter o melhor resultado no menor tempo possível.

Figura 8 | Classes das flores Iris

Fonte: UCI Machine Learning Repository (1988).

O algoritmo da rede MLP pode ser utilizado para a classificação de problemas mais complexos. No reconhecimento de caracteres, o número de neurônios de saída é definido pela quantidade de caracteres que o sistema deseja classificar. Já o tamanho da camada de entrada é determinado pelo vetor de características das amostras, ou seja, pela quantidade de informação numérica usada para descrever o caractere. A quantidade de camadas internas utilizadas, assim como o número de neurônios em cada camada, é definida com base em testes. Desse modo, é possível observar que não existe um modelo-padrão de rede MLP e que esta é moldada a partir do problema ao qual será aplicada.

As RNAs são atualmente utilizadas para várias finalidades. A ideia fundamental por trás da natureza das redes neurais é que, se funcionam na natureza, devem ser capazes de funcionar num sistema computacional (DATA SCIECEN ACADEMY, [s. d.]). Há vários projetos que apresentam classificadores aplicados ao reconhecimento

de padrões, como o uso da rede MLP para a segmentação de imagens médicas (JARRAR *et al.*, 2016), processamento de imagens de equipamentos elétricos (WENG *et al.*, 2016) e máquinas de aprendizagem extrema (TANG; DENG; HUANG, 2016).

VÍDEO RESUMO

Olá, estudante! No vídeo a seguir, você conhecerá mais detalhes sobre o surgimento das redes neurais e o que motivou seu desenvolvimento. O conteúdo apresenta as principais características das redes neurais artificiais perceptron simples e perceptron multicamadas. Além disso, será possível entender as possíveis aplicações dessas redes em problemas de classificação e saber como efetuar a implementação desses algoritmos.

Para visualizar o objeto, acesse seu material digital.

Saiba mais

Para implementar um algoritmo de rede perceptron utilizando a linguagem Python, você pode seguir o passo a passo descrito no artigo *Perceptron com Python, uma introdução.*

Você também pode aprofundar seus estudos sobre a matemática dos perceptrons no artigo *Introdução ao perceptron passo a passo.*

Além disso, o artigo *Rede neural perceptron multicamadas* ensina a criar a própria MLP do zero, usando a biblioteca TensorFlow. Vale a pena conferir!

Aula 3

REDES NEURAIS CONVOLUCIONAIS

O termo “redes neurais convolucionais” até parece uma estranha combinação entre biologia, matemática e um pouco de ciência da computação.

32 minutos

INTRODUÇÃO

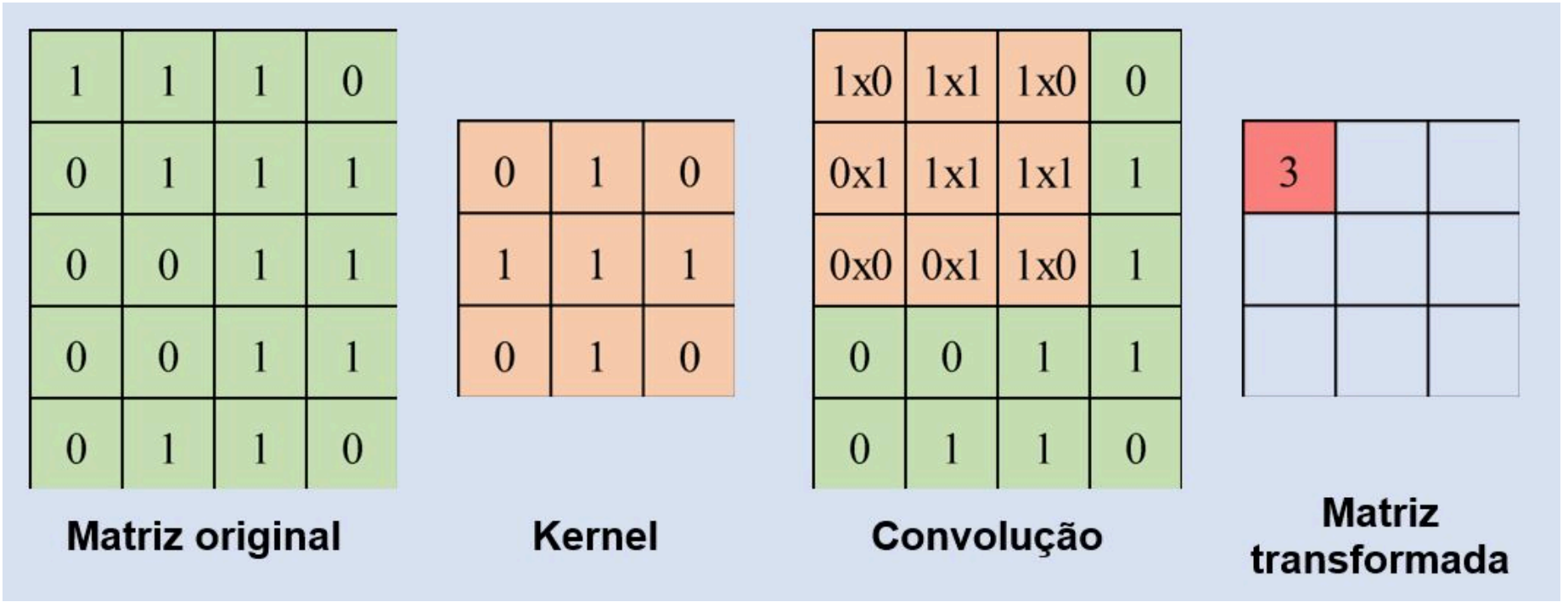
O termo “redes neurais convolucionais” até parece uma estranha combinação entre biologia, matemática e um pouco de ciência da computação. E em parte isso é verdade. As redes convolucionais (CNNs) são modelos computacionais inspiradas no córtex visual, o qual tem pequenas regiões de células sensíveis a regiões

específicas do campo visual. Consistem em camadas convolucionais, que utilizam o algoritmo de *backpropagation* para aprenderem os parâmetros/filtros de cada camada e, então, serem capazes de classificar imagens. As redes neurais convolucionais desempenham um papel importante no crescimento de aplicações com visão computacional, isso porque sua arquitetura espacial é particularmente mais bem adaptada para aplicações com imagens do que as redes neurais densas.

REDE CONVOLUCIONAL

Antes de discorrer sobre as redes convolucionais, é preciso entender o que é uma convolução. Matematicamente, uma convolução é uma operação linear que, a partir de duas funções, gera uma terceira (CLAPPIS, 2019). Visualizando no contexto de sinais, pode-se entender como um sinal de entrada que, por meio de um filtro/kernel com pesos, gera um sinal de saída. A Figura 1, a seguir, mostra como funciona esse processo. Um kernel é uma matriz utilizada para multiplicação de matrizes, sendo menor do que a matriz de entrada em largura e altura. A convolução pode ser realizada tanto em matrizes 2D como em 3D, ou seja, pode ser aplicada em imagens em escala de cinza e em imagens RGB.

Figura 1 | Operação de convolução de uma matriz 5x5 por um filtro 3x3

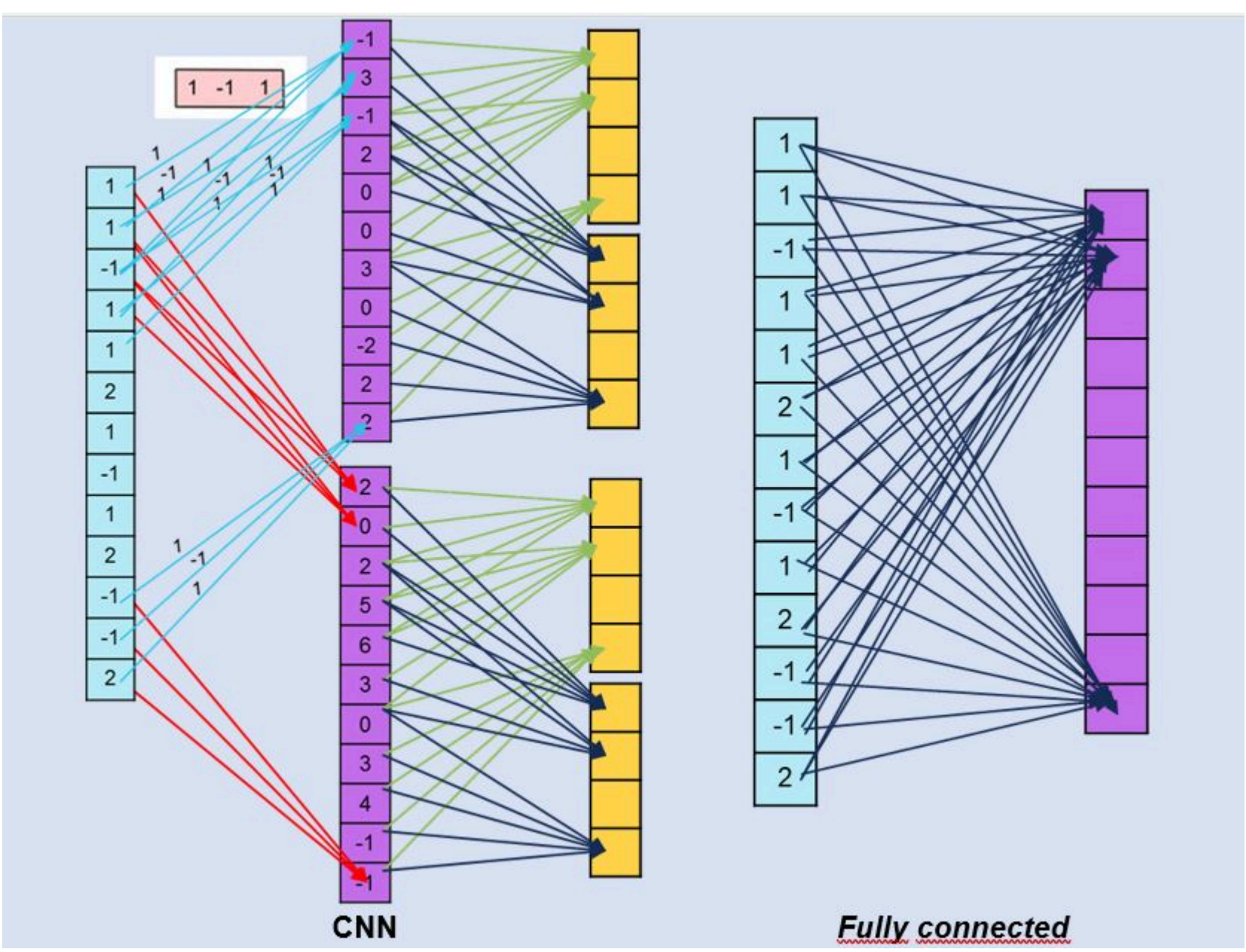


Fonte: elaborada pelo autor.

Uma rede neural convolucional (ConvNet/CNN, do inglês *convolutional neural network*) é um algoritmo de *deep learning* que pode captar uma imagem como sinal de entrada, atribuir importância (pesos e vieses que podem ser aprendidos) a vários aspectos da imagem e, assim, se tornar capaz de diferenciar objetos (GAUI, 2019). Sua arquitetura é análoga ao padrão de conectividade de neurônios no cérebro humano. Ou seja, essa rede pode ser compreendida como uma variação das redes perceptron multicamadas (MLP). A CNN é inspirada na organização do córtex visual, onde os neurônios individuais respondem a estímulos apenas em determinadas regiões do campo visual, as quais compõem o chamado campo receptivo (DATA SCIENCE ACADEMY, [s. d.]). O campo receptivo é um hiperparâmetro referente à extensão espacial da conexão entre a entrada e cada neurônio. Em outras palavras, é o tamanho do filtro da convolução.

Redes neurais *feedforward*, como a MLP, são redes que possuem camadas totalmente conectadas (*fully connected*), ou seja, todas as unidades de entrada são conectadas aos neurônios, os quais, por sua vez, são totalmente conectados aos seguintes. Por isso, também são conhecidas como redes neurais densas. O uso dessas redes pode não ser muito útil para classificar imagens, pois a arquitetura totalmente conectada não leva em consideração a estrutura espacial das imagens. Em imagens em escala de cinza, as redes *feedforward* podem até apresentar bons resultados, mas em imagens mais complexas, com dependências de pixels, teriam pouca ou nenhuma precisão. Entretanto, nas CNNs apenas um subconjunto de entradas é conectado a cada neurônio. Na Figura 2, a seguir, é possível ver a comparação das conexões dessas redes. A arquitetura espacial das CNNs é adaptada de maneira que reduz as imagens para uma forma mais fácil de processar, sem perder características críticas para a classificação, o que faz com que essas redes ou alguma variante semelhante às redes neurais sejam mais usadas para o reconhecimento de imagens.

Figura 2 | Conexões de uma rede neural convolucional e de uma fully connected



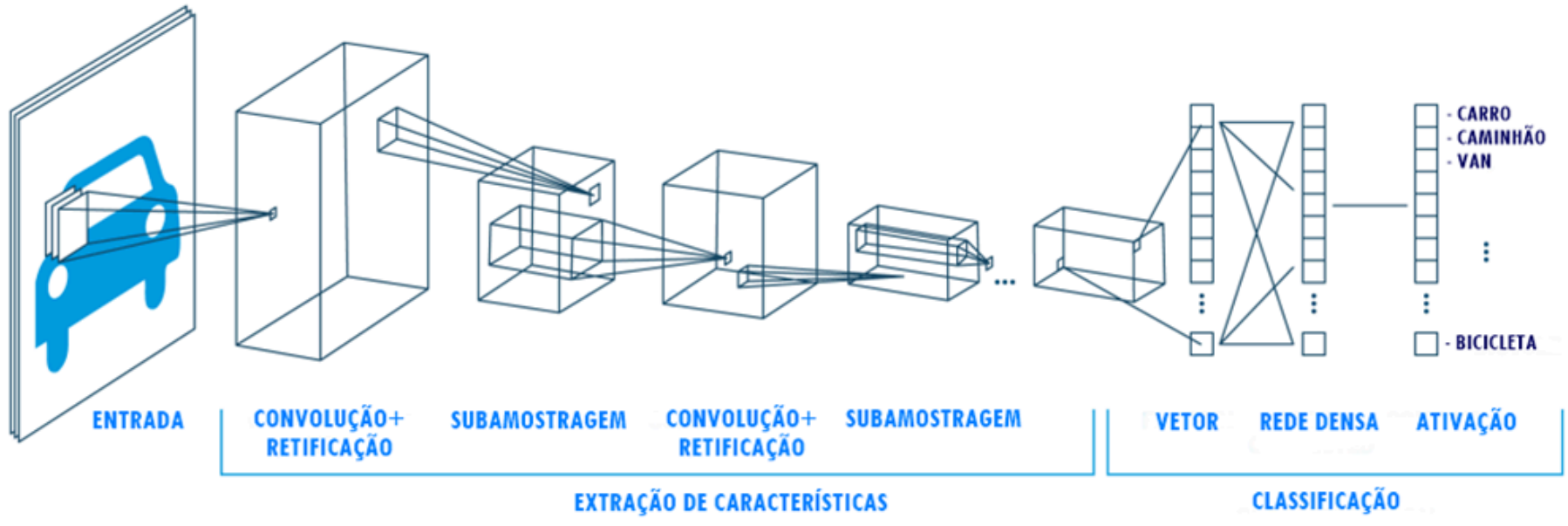
Fonte: elaborada pelo autor.

Em contraste com as redes *feedforward*, o pré-processamento exigido numa CNN é muito menor, pois, enquanto nos outros métodos os filtros são feitos à mão, a CNN, com treinamento suficiente, é capaz de aprender esses filtros. Contudo, essa rede acaba sendo computacionalmente mais cara e difícil de ajustar por precisar definir elementos como: escolha de arquitetura; tipos de parâmetros; hiperparâmetros; e algoritmo de aprendizado. Além disso, para obter uma alta acurácia, a CNN requer muitos dados.

TIPOS DE CAMADAS DA CNN

Atualmente, existem diversas arquiteturas de redes convolucionais conhecidas e variações destas. Embora as redes neurais densas (*fully connected*) possuam inúmeras camadas com pesos e resultados de treinamento diferentes, as redes convolucionais conseguem uma maior eficiência ao fragmentar as camadas em várias características, menos abstratas e mais especializadas na resolução da aplicação (SILVA *et al.*, 2019). Essa é uma das principais características das CNNs. A arquitetura de uma CNN típica consiste em quatro tipos de camadas: campo receptivo, camada de retificação, camada de subamostragem e camada de classificação, como você pode observar na Figura 3.

Figura 3 | Arquitetura de uma rede neural convolucional

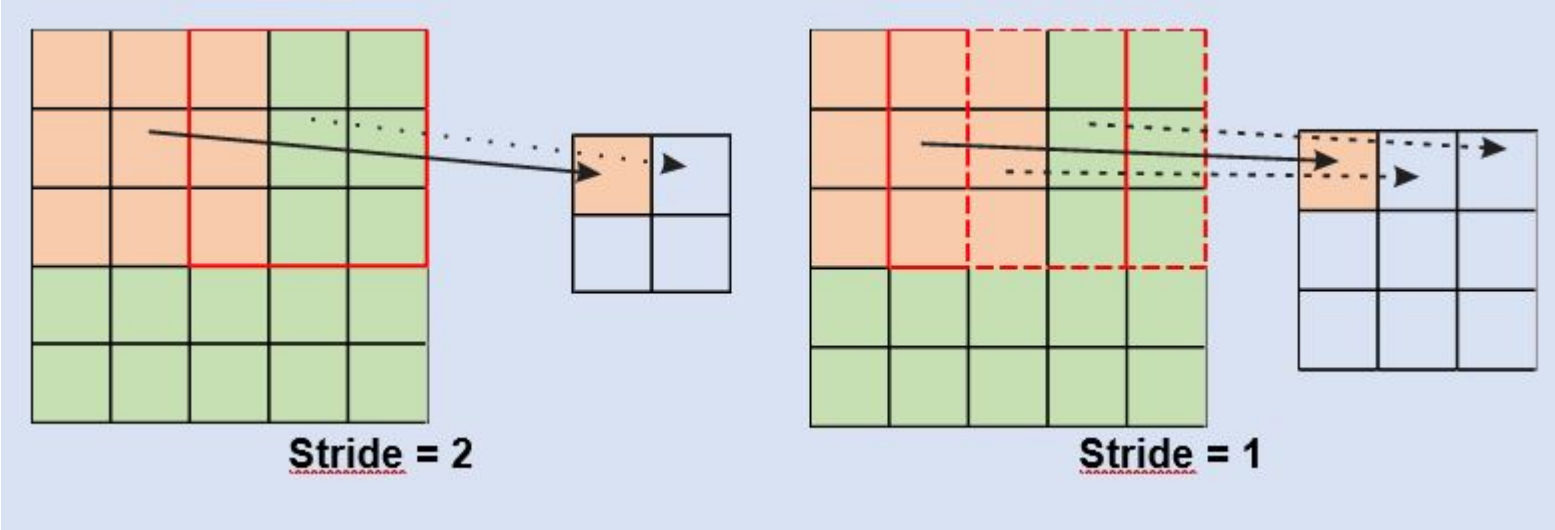


Fonte: adaptada de Saha (2018).

O campo receptivo, ou camada convolucional, é a primeira camada da CNN. Essa camada recebe a imagem de entrada e realiza a convolução com um conjunto de filtros para produzir uma imagem de saída. Cada filtro/kernel é útil para uma tarefa específica, seja nitidez, desfoque, detecção de borda, entre outras. Ou seja, depende das particularidades da imagem e do problema. Os pesos são calculados automaticamente com uso do *backpropagation*. Dessa forma, cada filtro aprendido se torna um extrator de características e cada imagem é resultante de um mapa de características (*feature map*).

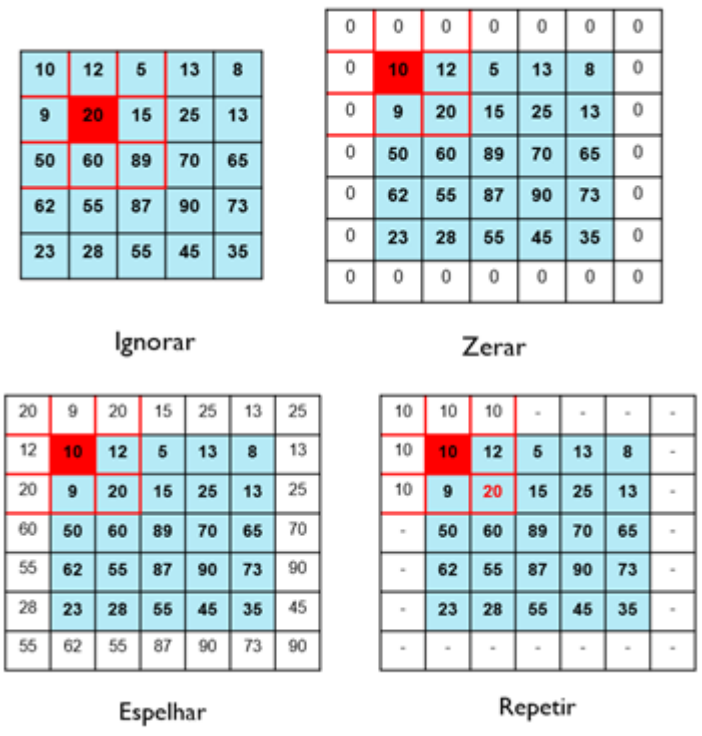
Além disso, a camada convolucional possui três principais hiperparâmetros: profundidade, *stride* e *padding*. A profundidade é referente ao número de filtros usados na convolução. O *stride* corresponde ao passo que o filtro desloca a cada iteração. Logo, quanto maior o *stride*, menos sobreposição haverá entre os campos receptivos e menor será a matriz resultante (Figura 4). Já o *padding* indica se a imagem de entrada deve ser preenchida ao redor da borda e como precisa acontecer tal preenchimento. A Figura 5 exibe alguns exemplos de *padding*.

Figura 4 | Exemplos de strides num filtro 3x3 aplicado a uma imagem 5x5



Fonte: elaborada pelo autor.

Figura 5 | Exemplos de padding de um filtro 3x3 numa imagem 5x5

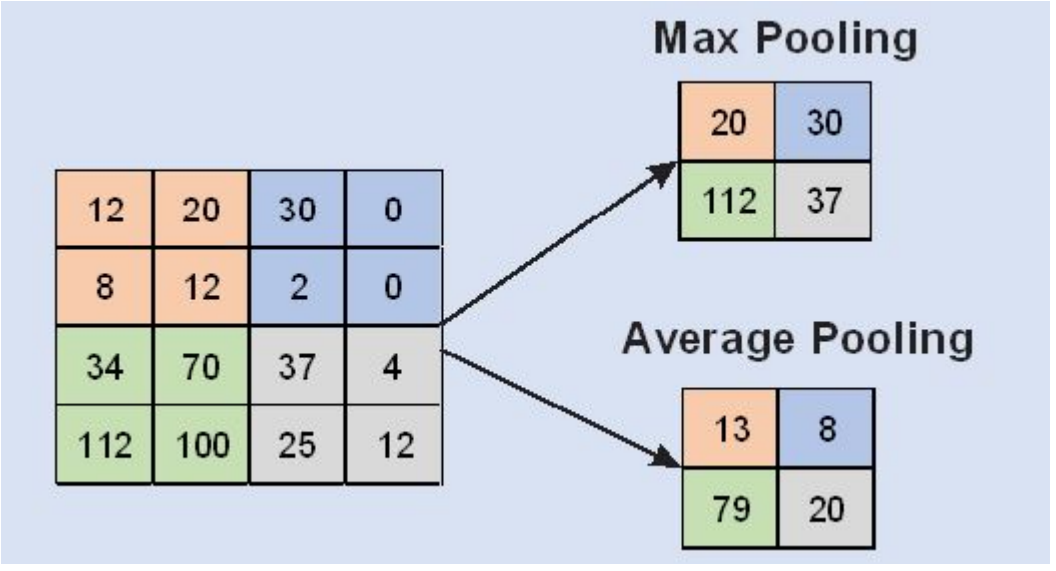


Fonte: elaborada pelo autor.

A camada de retificação é uma função de ativação, presente em cada neurônio, utilizada para aplicar uma transformação nos dados recebidos. Dependendo da necessidade, pode ser usada para várias funções diferentes. A função de ativação mais comumente utilizada é a ReLU (*rectified linear unit*), expressa por $f(x)=max(0,x)$, que possui um desempenho parecido com o da sigmoide e tangente hiperbólica, mas com um treinamento muito mais rápido.

A camada de subamostragem é comumente chamada de *pooling*. Essa camada funciona agrupando um conjunto de dados com a função de reduzir a dimensionalidade do *feature map*. É uma etapa importante para agilizar o treinamento, mas principalmente para criar invariância espacial. Além disso, é útil para a extração de características que são invariantes rotacionais e posicionais. Os modelos mais comuns de *pooling* são o *max pooling* e *average pooling* (Figura 6).

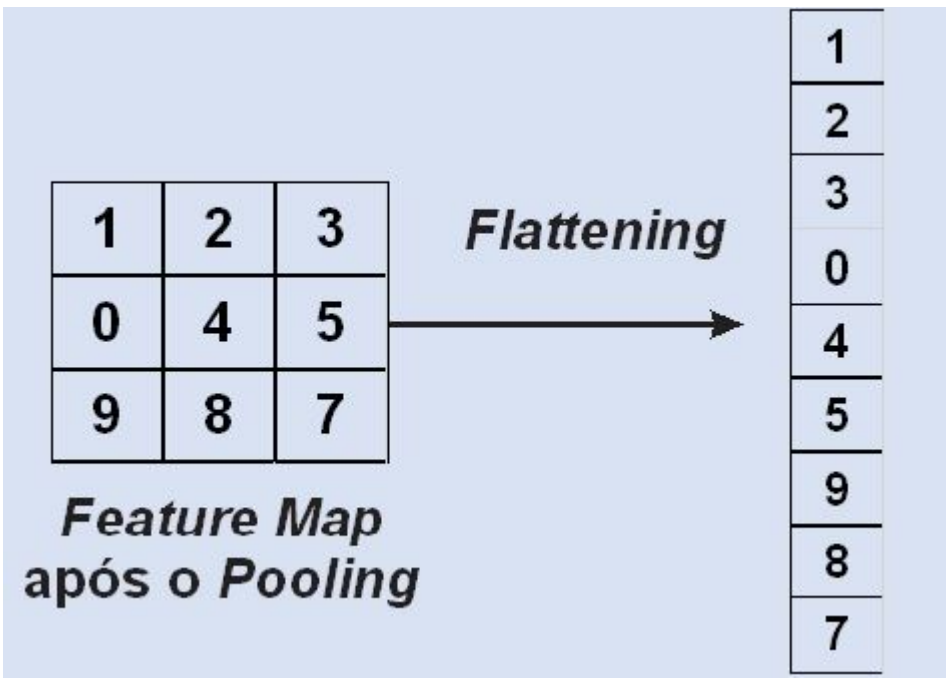
Figura 6 | Operação de pooling com max pooling e average pooling



Fonte: elaborada pelo autor.

Na divisão da CNN em extração de características e classificação (Figura 3), é utilizada uma camada *flatten*, que opera uma transformação na matriz da imagem, alterando seu formato para um vetor. Isso é necessário porque as RNAs recebem os dados de entrada como vetor. A Figura 7 apresenta essa operação. Quando a aplicação requer uma classificação, acrescenta-se, após o conjunto das camadas de convolução e *pooling*, pelo menos uma rede densa. Essa camada é fundamental para o treinamento, pois tem influência sobre o aprendizado dos filtros e, consequentemente, no resultado da rede. Depois da camada de *pooling* e antes dessa camada, é comum utilizar uma camada chamada de *dropout*, que é responsável por eliminar um conjunto aleatório de ativações para evitar que determinadas partes da rede neural fiquem muito sensíveis a pequenas alterações.

Figura 7 | Operação de flattening



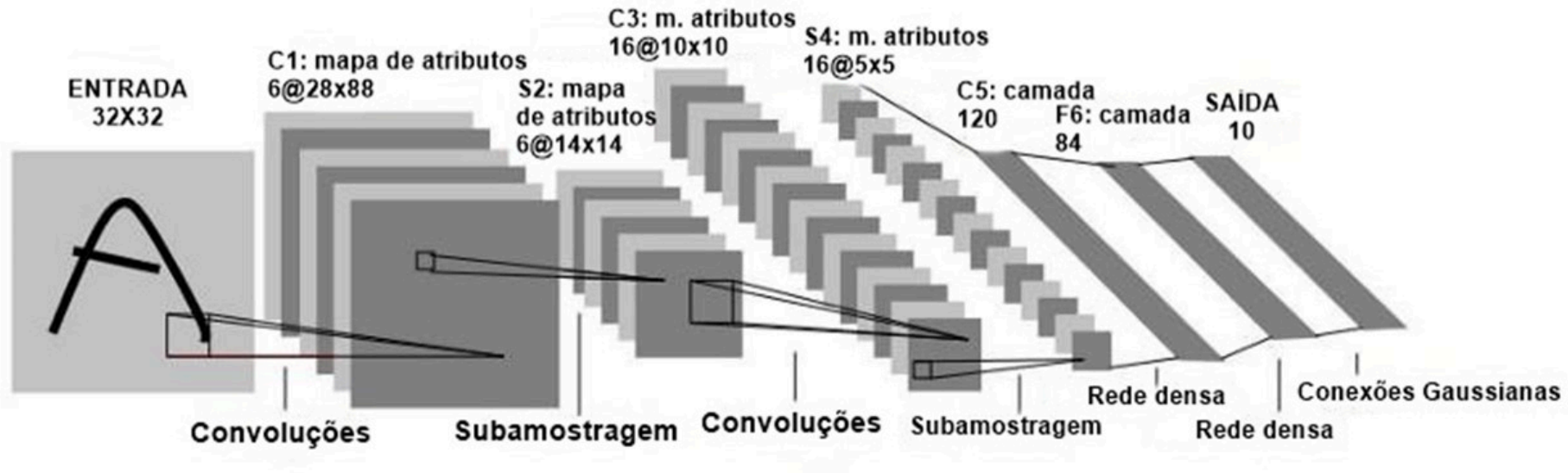
Fonte: elaborada pelo autor.

APLICAÇÕES COMERCIAIS DE CNN

Os primeiros estudos sobre redes neurais convolucionais remontam aos anos 1970, contudo foi o artigo *Gradient-based learning applied to document recognition* (traduzindo para o português, "Aprendizado baseado em gradiente aplicado ao reconhecimento de documentos"), de Yann LeCun, Léon Bottou, Yoshua Bengio e Patrick Haffner, publicado em 1998, que estabeleceu o tema moderno das redes convolucionais (DATA SCIENCE ACADEMY, [s. d.]). Nesse trabalho, apresentou-se a primeira implementação bem-sucedida de uma CNN, LeNet-5, aplicada ao reconhecimento de dígitos manuscritos. Essa rede possui duas camadas

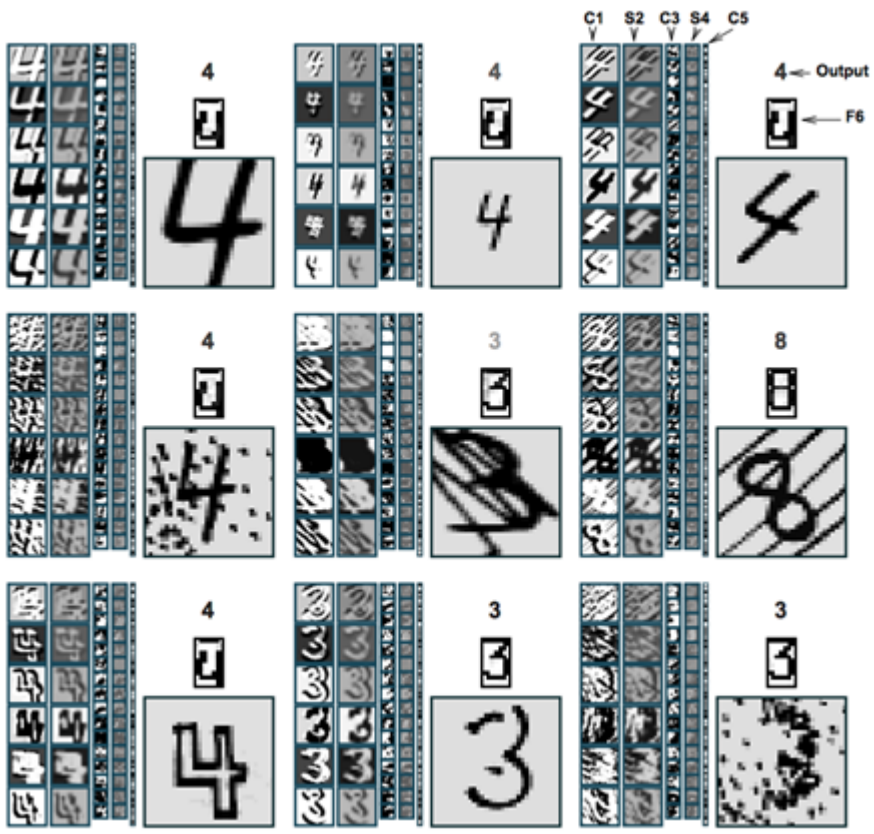
convolucionais, duas de *pooling* e uma camada de classificação. É possível observar sua arquitetura na Figura 8, a seguir. Além disso, a LeNet-5 apresenta uma classificação resistente a ruídos, inclinação e formatação, como mostra a Figura 9.

Figura 8 | Arquitetura da rede convolucional LeNet-5



Fonte: adaptada de LeCun et al. (1998).

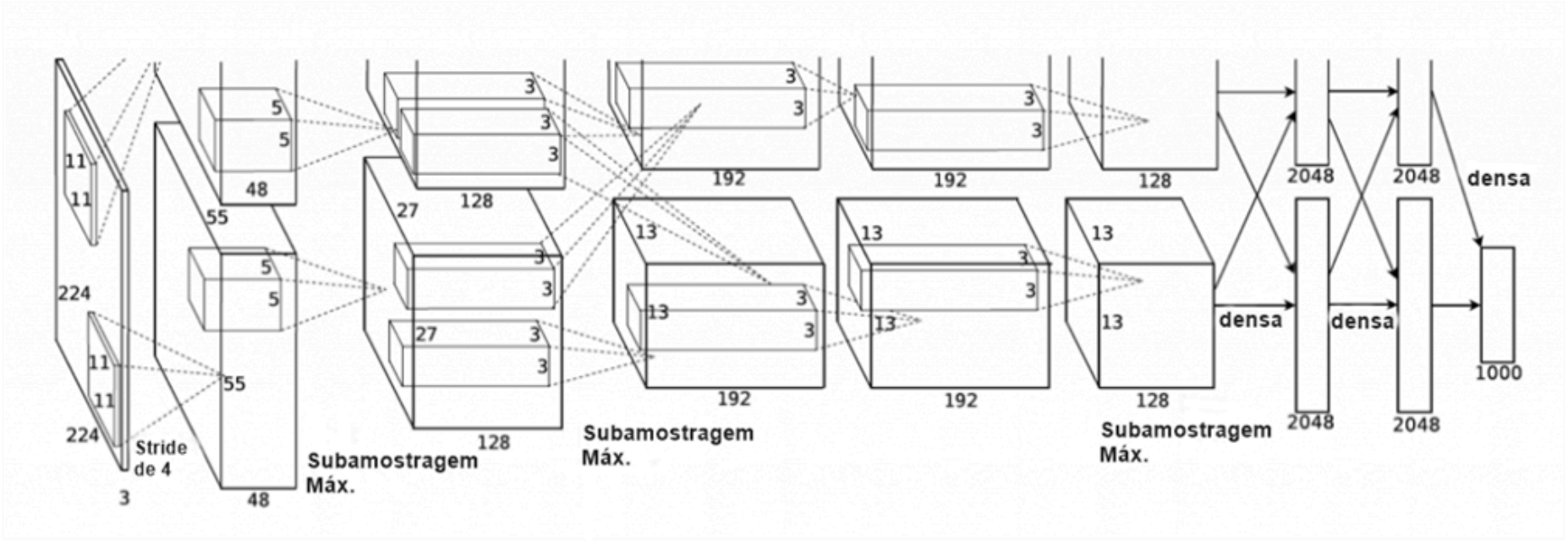
Figura 9 | Classificação da rede convolucional LeNet-5



Fonte: LeCun et al. (1998).

As CNNs só passaram a ganhar mais protagonismo no meio das aplicações de *machine learning* em 2012, quando Alex Krizhevsky apresentou um modelo que venceu o desafio LSVRC-2012 da ImageNet, a "olimpíada" da visão computacional. A rede, que ficou conhecida como AlexNet, alcançou um erro de 15,3%, em contraste com o segundo lugar, que obteve 26,2%. Sua arquitetura contém cinco camadas convolucionais seguidas de *pooling* e três camadas densas, como indica a Figura 10.

Figura 10 | Arquitetura da rede convolucional AlexNet



Fonte: adaptada de Krizhevsky, Sutskever e Hinton (2017).

A partir desses trabalhos, pesquisas voltadas para o desenvolvimento de diferentes arquiteturas de redes convolucionais passaram a ganhar mais atenção. Existem muitas arquiteturas conhecidas atualmente e comumente usadas para realizar a detecção de objetos e a classificação de imagens. Alguns exemplos de redes bem populares são: rede neural convolucional baseada em região (R-CNN, do inglês *region-based convolutional neural networks*); Fast R-CNN, que é uma versão mais simplificada da arquitetura da R-CNN; GoogLeNet, também chamada de Inception v1; VGGNet; e ResNet.

Nos últimos anos, algoritmos de detecção de objetos baseados em redes neurais melhoraram muito, tanto em precisão quanto em velocidade, em imagens e vídeos (LIU *et al.*, 2021). Isso aconteceu principalmente pela popularidade das CNNs. A arquitetura espacial das redes convolucionais as torna modelos essenciais para quem deseja trabalhar com aplicações com imagens, seja em problemas de classificação, localização, detecção ou segmentação.

Existem diversas aplicações comerciais de classificação de imagens que fazem uso das redes convolucionais. Em sistemas de tags de imagens, é possível encontrar imagens a partir de uma palavra ou combinação de palavras que as descrevem. Google, Facebook e Amazon usam essa tecnologia. Aplicações com pesquisa visual fazem combinações de uma imagem de entrada com um banco de dados, ou seja, analisam uma imagem e procuram outras imagens com as informações identificadas. O Google Imagens, por exemplo, permite que se realize uma pesquisa inserindo uma imagem no lugar da descrição. Outro exemplo são sistemas de recomendações, como em sites de compras, que usam o reconhecimento de imagens com CNN para recomendações de produtos, algo bastante utilizado pela Amazon.

Além da classificação de imagens, existem diversas pesquisas sobre o uso de CNN em análise de imagens médicas. Esses sistemas podem analisar sequências de imagens e identificar diferenças sutis que os analistas humanos podem não perceber, principalmente quando essas avaliações são efetuadas durante um longo período de tempo. Assim como a primeira implementação de uma CNN, aplicações com o reconhecimento ótico de caracteres (OCR, do inglês *optical character recognition*) também são alvos de pesquisas com CNN, isso porque podem ser usadas para melhorar a pesquisa em conteúdo de mídia avançada e identificar texto em documentos escritos, mesmo no caso de conteúdos difíceis de se reconhecer. Assim, nota-se que o mercado é amplo e versátil para quem deseja trabalhar com visão computacional.

VÍDEO RESUMO

Olá, estudante! No vídeo a seguir, você aprenderá mais informações sobre a operação de convolução e o funcionamento de uma rede neural convolucional (CNN). Além disso, você saberá quais são os tipos de camadas que uma CNN comum possui e suas particularidades. Por fim, também será possível conhecer as possíveis aplicações das CNNs na atualidade.

Para visualizar o objeto, acesse seu material digital.

Saiba mais

Uma das arquiteturas de rede convolucional mais simples é a LeNet-5. É um ótimo ponto de partida para quem quer implementar seu primeiro algoritmo de CNN. No artigo *Tutorial completo no LeNet-5 | Guia para começar com CNNs*, você aprenderá a efetuar a implementação dessa rede.

Se deseja implementar uma rede um pouco mais complexa, você pode fazer a leitura do guia para a rede AlexNet, intitulado *Redes neurais convolucionais profundas (AlexNet).*

Aula 4

REDES NEURAIS RECORRENTES

A rede neural recorrente (RNN) é um tipo de rede neural artificial que usa dados sequenciais ou dados de séries temporais

37 minutos

INTRODUÇÃO

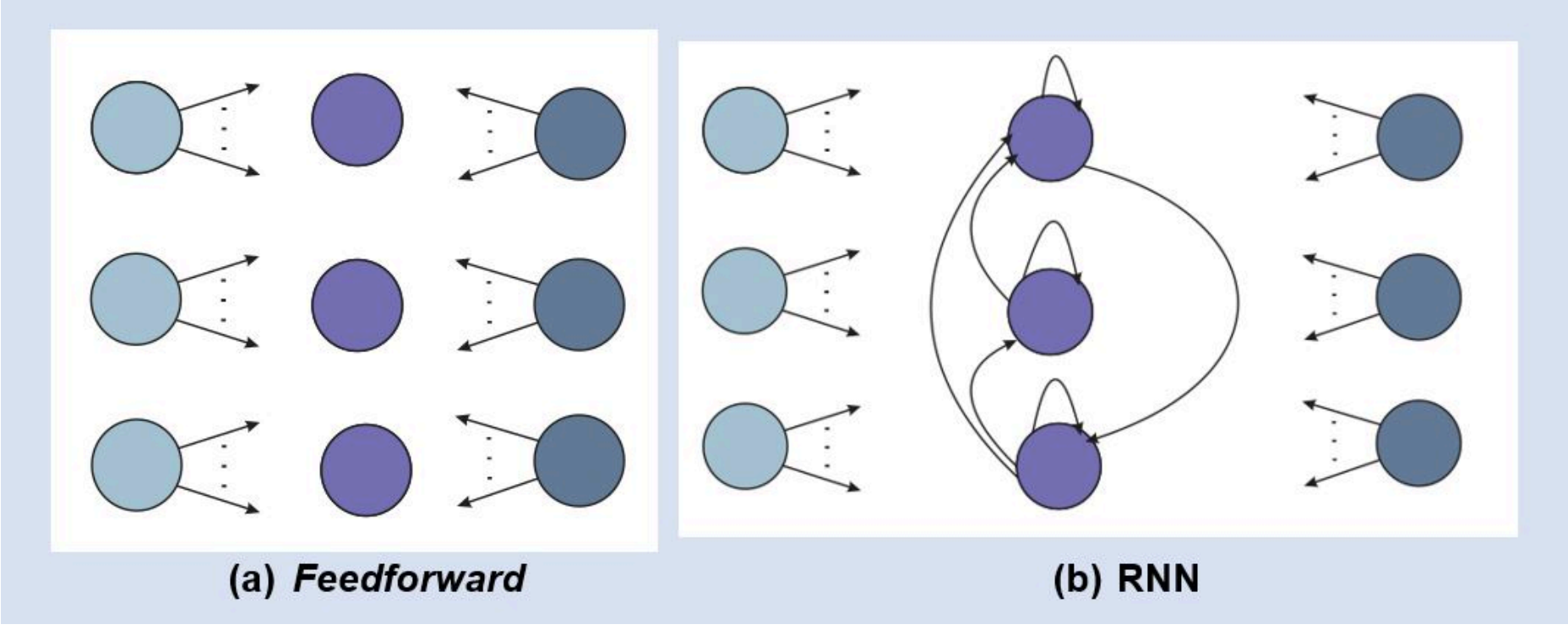
Depois de treinada, a rede *feedforward* se torna uma estrutura estática, ou seja, não é capaz de manter memórias entre entradas ou alterar o modo como processa uma entrada com base nas entradas que viu no passado. Entretanto, muitos problemas exigem o reconhecimento de padrões em sequências de dados. A rede neural recorrente (RNN) é um tipo de rede neural artificial que usa dados sequenciais ou dados de séries temporais. Assim como as redes feedforward, as RNNs utilizam dados de treinamento para aprender, contudo são diferenciadas por sua “memória”, pois recebem informações de entradas anteriores para influenciar a entrada e a saída atuais. Um modelo bastante conhecido de RNN é a “memória longa de curto prazo” (LSTM).

REDES RECORRENTES

Uma rede *feedforward* é treinada de maneira que os dados rotulados são apresentados e a rede procura criar um modelo que minimize o erro ao classificar suas categorias. Mesmo que essa rede treinada seja exposta a qualquer sequência aleatória de amostras, a classificação da primeira amostra não alterará necessariamente a classificação da segunda. Ou seja, a sequência não importa, porque esses modelos não têm noção de ordem no tempo. Contudo, a memória é um fator essencial quando pensamos no aprendizado humano. Levando em consideração esse objetivo, surgiram as redes neurais recorrentes (RNNs, do inglês *recurrent neural networks*).

As RNNs são diferentes das redes *feedforward* porque aproveitam um tipo especial de camada neural, conhecida como camada recorrente, a qual permite que a rede mantenha o estado entre os usos da rede (BUDUMA; LOCASCIO, 2017). Observa-se, na Figura 1a, que os neurônios das redes *feedforward* possuem conexões de entrada, que partem de todos os neurônios da camada anterior, e conexões de saída, que levam todos os neurônios à camada subsequente. Já na Figura 1b, nota-se que, além dessas conexões, as camadas recorrentes possuem conexões que propagam informações entre os neurônios da mesma camada. Em uma camada recorrente totalmente conectada, há um fluxo de informações de cada neurônio para todos os outros neurônios em sua camada, incluindo o mesmo neurônio.

Figura 1 | Neurônios nas redes recorrentes e nas redes *feedforward*



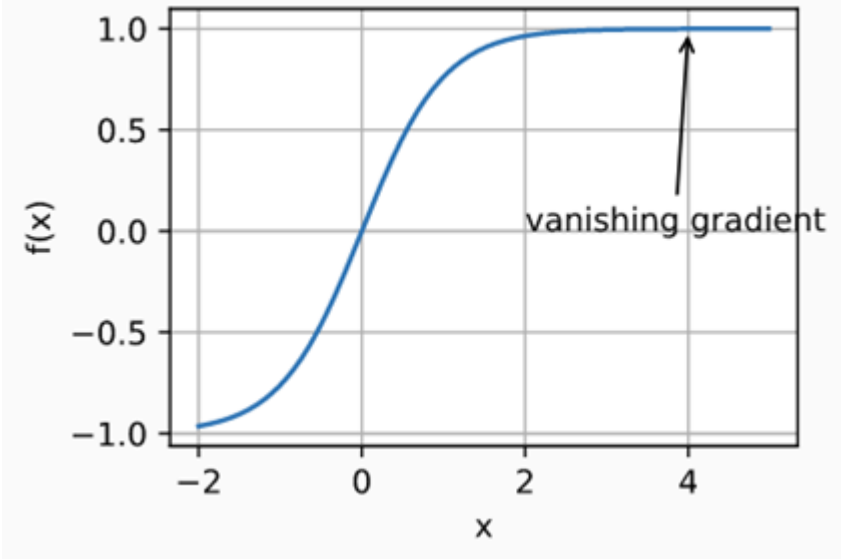
Fonte: elaborada pelo autor.

As conexões das redes *feedforwards* representam o fluxo de informação que ocorre de um neurônio para outro. Ou seja, os dados que estão sendo transferidos são a ativação do neurônio em um passo de tempo real. Já nas RNNs, as conexões representam o fluxo de informações em que os dados são a ativação neuronal armazenada da etapa de tempo anterior. Portanto, as ativações dos neurônios simbolizam o estado acumulativo da instância da rede, sendo a ativação inicial dos neurônios na camada recorrente o parâmetro para o modelo (BUDUMA; LOCASCIO, 2017). Assim, é possível observar que as RNNs são projetadas para lidar melhor com as informações sequenciais do que os modelos *feedforward*.

As pesquisas em redes neurais recorrentes foram primeiramente introduzidas na década de 1980, mas só passaram a ganhar popularidade recentemente, por causa dos avanços intelectuais de hardware, isso porque é difícil treiná-las para capturar dependências de longo prazo, dando origem ao desaparecimento do

gradiente (DENG; YU, 2013). Certas funções de ativação, como a função sigmoide, comprimem um grande espaço de entrada num pequeno espaço de saída entre 0 e 1. Portanto, uma grande alteração na entrada da função sigmoide causará uma pequena alteração na saída. Para redes com apenas algumas camadas que usam essas ativações, isso não é um grande problema. No entanto, quando mais camadas são usadas, isso pode fazer com que o gradiente seja muito pequeno em comparação ao necessário para que o treinamento funcione efetivamente. Esse problema é conhecido como desaparecimento do gradiente, ilustrado na Figura 2. O avanço das pesquisas mostra que atualmente esse problema pode ser resolvido com mais facilidade, e algumas técnicas de otimização podem ser utilizadas para solucioná-lo, como a otimização livre hessiana (MARTENS, 2010) ou o gradiente descendente estocástico (BENGIO, 2013).

Figura 2 | Desaparecimento do gradiente *(vanishing gradient)*



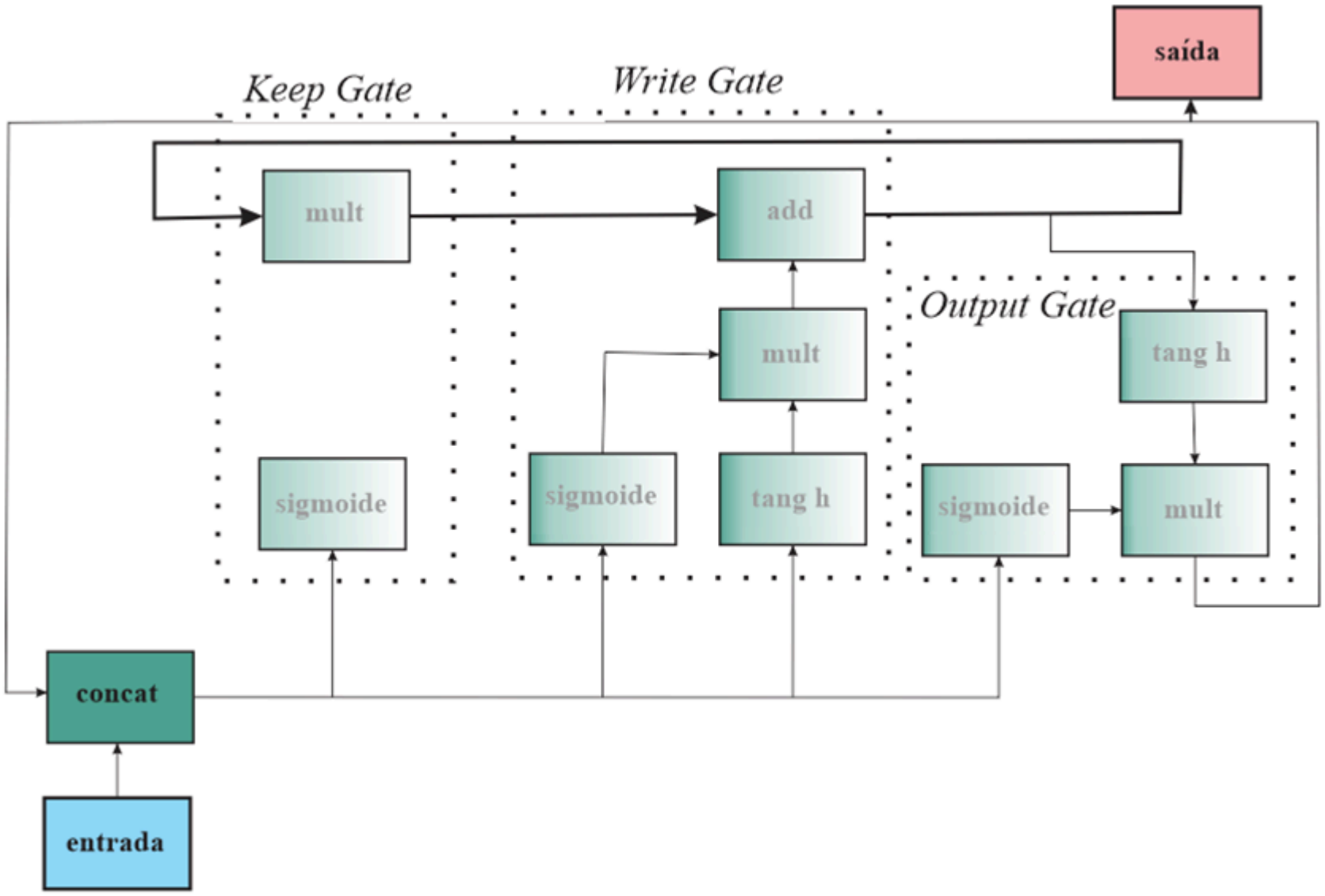
Fonte: Zhang et al. (2021).

MEMÓRIA LONGA DE CURTO PRAZO (LSTM)

Em desafios de previsão de sequência, as redes neurais tradicionais acabam não sendo aplicáveis. Nesse contexto, surgiram as redes recorrentes, que podem aprender a dependência de ordem. Entretanto, os primeiros estudos sobre as RNNs encontraram algumas limitações e problemas para implementá-las. Na intenção de solucionar o problema do gradiente desaparecendo, Sepp Hochreiter e Jürgen Schmidhuber apresentaram o método “longa memória de curto prazo” (LSTM, do inglês *long short-term memory*) (HOCHREITER; SCHMIDHUBER, 1997).

As redes recorrentes possuem memória de longo prazo na forma de pesos, os quais mudam lentamente conforme o treinamento, codificando o conhecimento geral sobre os dados. Além disso, possuem memória de curto prazo em forma de ativações efêmeras, que passam de cada neurônio para os neurônios seguintes (ZHANG *et al.*, 2021). Em vista dessa intuição, surgiu o termo “longa memória de curto prazo”. A rede LSTM foi projetada com o propósito de transmitir de maneira confiável informações importantes, por vezes para projetos futuros. A arquitetura de uma unidade LSTM pode ser observada na Figura 3, a seguir.

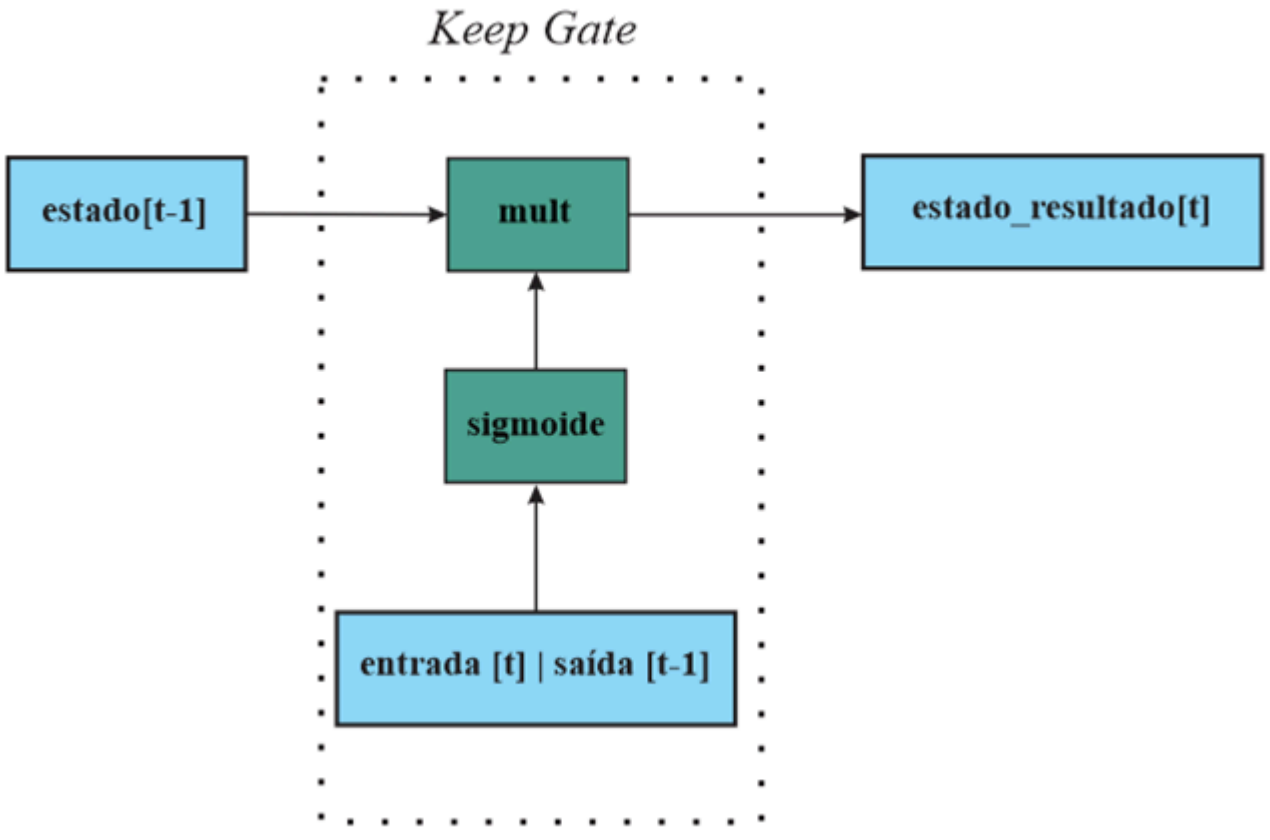
Figura 3 | Arquitetura de uma unidade LSTM



Fonte: elaborada pelo autor.

Com um design inspirado pelas portas lógicas de um computador, a LSTM introduz um tipo intermediário de armazenamento via célula de memória que contém as informações críticas assimiladas ao longo do tempo. A cada passo de tempo , a unidade LSTM modifica a célula de memória com novas informações com três fases diferentes (*keep gate*, *write gate* e *output gate*). Primeiro, a unidade determina quanto da memória anterior deve ser mantida. Isso é definido pelo *keep gate*, ilustrado na Figura 4. Esse componente analisa quais elementos na memória ainda são relevantes e quais são obsoletos e devem ser apagados. Para fazer essa descoberta, utilizam-se a entrada da etapa t e a saída da unidade da etapa t-1, e aplica-se uma função de ativação na concatenação dessas entradas. Assim, tal componente pode ser utilizado como um tensor de bits (0 ou 1), para verificar quais informações são relevantes ao realizar uma multiplicação. Se o resultado for 0, pode-se descartar; caso seja 1, deve-se manter.

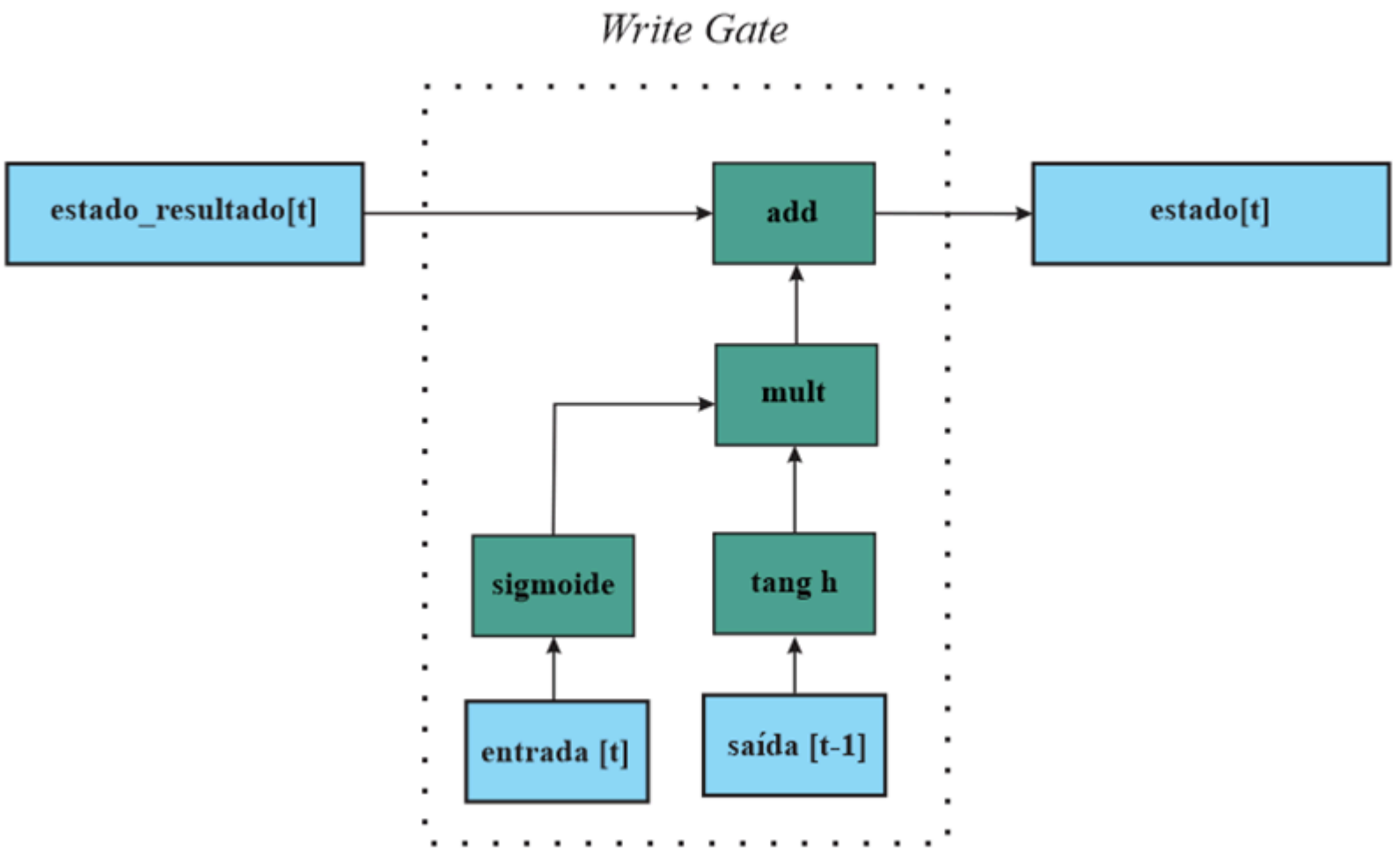
Figura 4 | Arquitetura do keep gate de uma unidade LSTM



Fonte: elaborada pelo autor.

Depois de definir as informações relevantes, o *write gate* (Figura 5) decide quais informações serão gravadas no estado de memória. A informação de entrada é regulada por uma função sigmoide e filtrada, como no *keep gate*, enquanto para a saída t-1 aplica-se uma função tangente hiperbólica (que gera uma saída de -1 a +1). Os resultados são, então, multiplicados e adicionados com o objetivo de criar o novo vetor de estado para o LSTM.

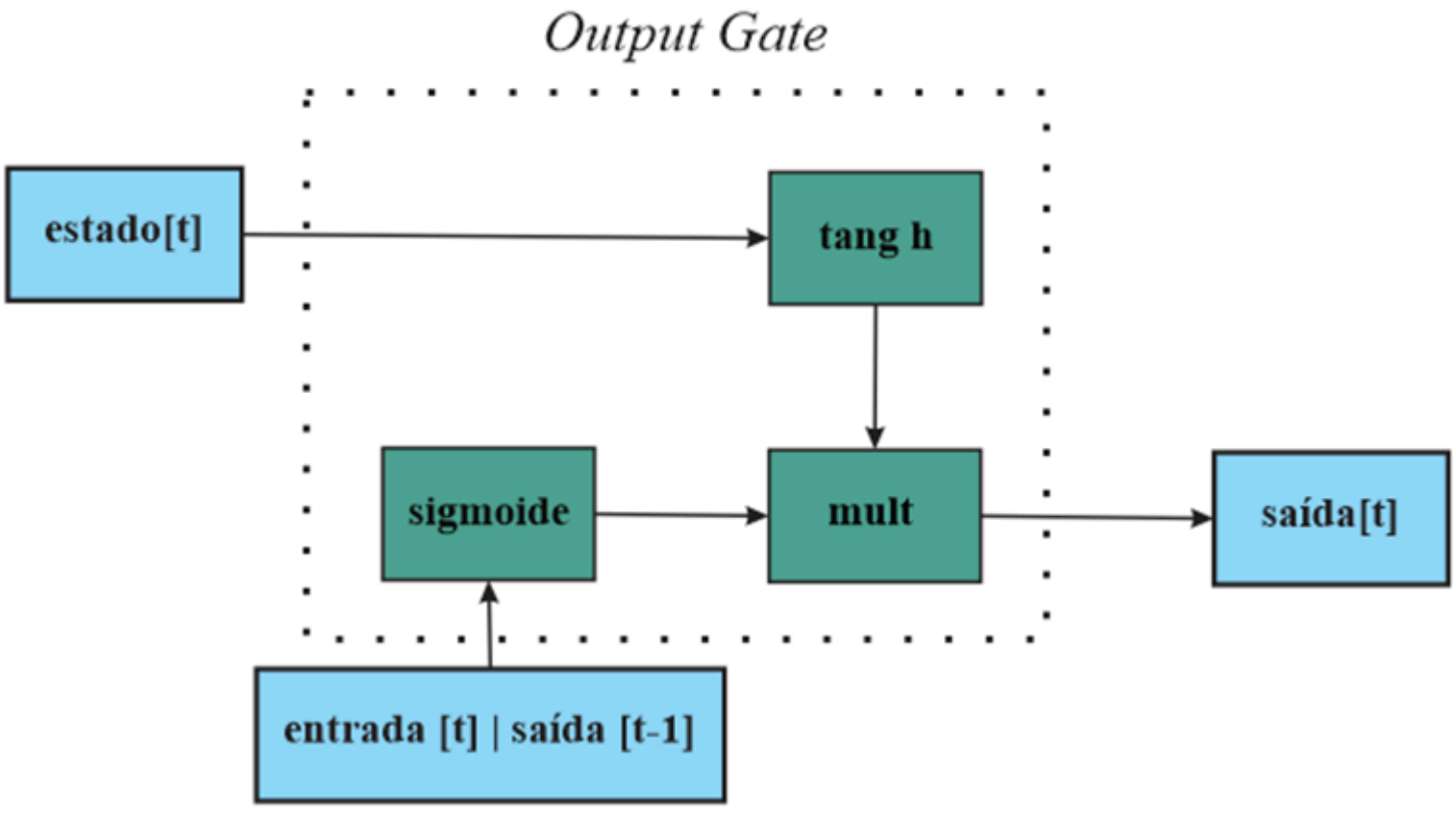
Figura 5 | Arquitetura do *write gate* de uma unidade LSTM



Fonte: elaborada pelo autor.

Por fim, a saída com as informações úteis do estado da célula atual é feita pelo *output gate*, como mostra a Figura 6. Sua arquitetura é similar à do *keep gate*, exceto pelo fato de que na informação obtida pelo *write gate* é aplicada a função tangente hiperbólica. A saída final é obtida pela multiplicação dos resultados das funções de ativação.

Figura 6 | Arquitetura do *output gate* de uma unidade LSTM



Fonte: elaborada pelo autor.

Dessa forma, é possível observar a propagação do vetor de estados. Assim, o gradiente que relaciona uma entrada com várias etapas de tempo no passado à saída atual não se atenua tão drasticamente quanto na primeira formulação de arquitetura RNN. Isso significa que o LSTM pode aprender relacionamentos de longo prazo com muito mais eficácia.

APLICAÇÕES COMERCIAIS DE REDES RECORRENTES

Até então, assumimos aplicações em que os dados são distribuídos de forma independente e idêntica, como em modelos que analisam uma única imagem e produzem apenas uma previsão. Contudo, a maioria dos dados não funciona assim. Existem aplicações nas quais nos deparamos com uma sequência de imagens, como em vídeos ou numa previsão estruturada sequencialmente, por exemplo no caso da legendagem de imagens. Embora as CNNs possam processar informações espaciais com eficiência, as redes neurais recorrentes possuem uma arquitetura projetada para problemas com informações sequenciais.

As RNNs podem ser consideradas como uma classe de *deep learning* de aprendizado tanto supervisionado como não supervisionado. No aprendizado não supervisionado, essa rede pode ser usada para uma sequência de dados no futuro, usando as amostras de dados anteriores sem a necessidade de informação de classe adicional. Inúmeras tarefas de aprendizado implicam a necessidade de lidar com dados sequenciais e fazem uso de modelos de RNNs, como modelagem de linguagem, tradução de idiomas, legendas em imagens, geração de texto, chatbots, entre outros.

Em aplicações de modelagem de linguagem e geração de texto, o sistema toma uma sequência de palavras como entrada e, então, tenta prever a possibilidade da próxima palavra. Quase todos os sistemas de tradução usados atualmente usam alguma versão avançada de RNN. Um dos exemplos mais conhecidos é o Google Tradutor, que atualmente, além de traduzir palavra por palavra, efetua a tradução de uma sentença mantendo o significado original. Plataformas de comércio eletrônico, como Flipkart, Amazon e eBay, fazem uso da tradução automática em muitas áreas e também cooperam com a eficiência dos resultados da pesquisa. Interfaces de conversação e chatbots correspondem a alguns dos recursos que mais se utilizam da geração de texto. Outro exemplo de aplicação é em resumo de textos, como para resumir materiais de literatura e personalizá-los para a entrega em aplicativos que não suportam grandes volumes de texto.

As RNNs também podem ser aplicadas em problemas de reconhecimento de fala, prevendo segmentos fonéticos a partir da consideração de ondas sonoras de um meio como fonte de entrada. São comuns em assistentes pessoais, como o Google Assistant, Microsoft Cortana, Amazon Alexa e Apple Siri. Uma das principais aplicações de RNNs no processamento de áudio acontece na análise de *call center*. Nessas aplicações, é possível extrair informações no final de uma chamada e fornecer às empresas soluções para o sucesso da equipe de suporte, descrevendo quais foram as etapas desenvolvidas para resolver o problema do cliente.

Sistemas para a geração de descrição de imagens são um exemplo de combinação de redes convolucionais e redes recorrentes. A CNN pode ser utilizada para a segmentação do objeto, e a RNN, para prover uma descrição do que foi segmentado. Além disso, as RNNs também podem ser usadas em outros problemas de imagens, como tags de vídeos, detecção facial, aplicações de OCR e reconhecimento de imagens.

A quantidade de aplicações comerciais é vasta e continua aumentando. Com a expansão dos dados gerados a cada dia e a necessidade de processá-los, as redes recorrentes se tornam cada vez mais comuns em aplicações de *machine learning*. As RNNs estão na base das maravilhas modernas da inteligência artificial, de forma que o amplo conhecimento dessas redes se tornou um dos requisitos mais procurados em profissionais de inteligência artificial.

VÍDEO RESUMO

Olá, estudante! Até este momento, você já conheceu alguns modelos de redes *deep learning* poderosas no reconhecimento de padrões. Contudo, tais redes, depois de treinadas, consideram que os dados são independentes entre si. Neste vídeo, você aprenderá mais detalhes sobre o modelo de *deep learning* capaz de guardar “memória” sobre iterações anteriores, que são as RNNs. Além disso, será possível verificar algumas aplicações e modelos conhecidos.

Para visualizar o objeto, acesse seu material digital.

Saiba mais

Caso você queira desenvolver sua primeira rede neural recorrente, a seção 5 do capítulo 8 do livro *Dive into deep learning (Implementação de redes neurais recorrentes do zero*) ensina a implementar do zero uma RNN na linguagem Python.

Além disso, na seção 1 do capítulo 9 do mesmo livro, você poderá aprofundar seu entendimento sobre a *gated recurrent unit* (GRU), outra arquitetura de RNN, aprendendo a implementá-la.

Por fim, na seção 2 do capítulo 9 do mesmo livro, você saberá como implementar a *Memória Longa de Curto Prazo (LSTM)*.

REFERÊNCIAS

15 minutos

Aula 1

AFINAL, o que é Deep Learning? **Gaea**, 21 mar. 2019. Disponível em: https://gaea.com.br/afinal-o-que-e-deep-learning/ . Acesso em: 3 out. 2022.

BENGIO, Y. Learning Deep Architectures for AI. **Foundations and Trends® in Machine Learning**, v. 2, n. 1, p.1-127, 2009. Disponível em: https://www.iro.umontreal.ca/~lisa/pointeurs/TR1312.pdf. Acesso em: 3 out. 2022.

BREUEL, C. 2020: o ano da realidade na inteligência artificial. **LinkedIn**, 23 jan. 2020a. Disponível em: https://www.linkedin.com/pulse/2020-o-ano-da-realidade-na-intelig%C3%AAncia-artificial-cristiano-breuel/ . Acesso em: 5 out. 2022.

DATA SCIENCE ACADEMY. **Deep learning book**, [s. d.]. Disponível em: https://www.deeplearningbook.com.br/ . Acesso em: 4 out. 2022.

DENG, L.; YU, D. Deep learning: methods and applications. **Foundations and Trends® in Signal Processing**, v. 7, n. 3-4, p. 197-387, 2013. Disponível em: https://www.nowpublishers.com/article/DownloadEBook/SIG-039. Acesso em: 3 out. 2022.

HINTON, G. E.; OSINDERO, S.; TEH, Y. A fast learning algorithm for deep belief nets. **Neural Computation**, v. 18, p. 1527-1554, 2006. Disponível em: https://www.cs.toronto.edu/~hinton/absps/ncfast.pdf. Acesso em: 3 out. 2022.

LEVY, M. Deep Learning on MCUs is the Future of Edge Computing. **EE Times**, 21 ago. 2020. Disponível em: https://www.eetimes.com/deep-learning-on-mcus-is-the-future-of-edge-computing/ . Acesso em: 5 out. 2022.

PICCOLO, L. Carreiras em machine learning: o presente e o futuro. **SBC Horizontes**, 26 jul. 2020. Disponível em: http://horizontes.sbc.org.br/index.php/2020/07/26/carreiras-em-machine-learning/ . Acesso em: 4 out. 2022.

RAMANATHAN, S. 5 Challenges for Deep Learning. **EE Times**, 31 jul. 2018. Disponível em: https://www.eetimes.com/5-challenges-for-deep-learning/ . Acesso em: 5 out. 2022.

RIBEIRO, V. Z. Cinco passos para criar seu 1º projeto de deep learning com Python e Keras. **Insight**, 7 maio 2021. Disponível em: https://insightlab.ufc.br/5-passos-para-criar-seu-1o-projeto-de-deep-learning-com-python-e-keras/. Acesso em: 3 out. 2022.

SCHMIDHUBER, J. Deep learning in neural networks: an overview. **Neural Networks**, v. 61, p. 85-117, jan. 2015. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0893608014002135. Acesso em: 3 out. 2022.

VALDATI, A. de B. **Inteligência artificial – IA**. Curitiba: Contentus, 2020.

Aula 2

AFFONSO, E. T. F. *et al*. Uso redes neurais multilayer perceptron (MLP) em sistema de bloqueio de websites baseado em conteúdo. **Mecánica Computacional**, Buenos Aires, v. 29, p. 9075-9090, 2010. Disponível em: https://cimec.org.ar/ojs/index.php/mc/article/view/3654/3567. Acesso em: 9 out. 2022.

BRAGA, A. de P.; LUDERMIR, T. B.; CARVALHO, A. C. P. de L. F. **Redes neurais artificiais**: teoria e aplicações. Rio de Janeiro: LTC, 2000.

DATA SCIENCE ACADEMY. **Deep learning book**, [s. d.]. Disponível em: https://www.deeplearningbook.com.br/. Acesso em: 9 out. 2022.

DUA, D.; GRAFF, C. **UCI Machine Learning Repository**. Irvine: University of California/School of Information and Computer Science, 2019. Disponível em: http://archive.ics.uci.edu/ml. Acesso em: 10 out. 2022.

GERVEN, M. V.; BOHTE, S. Artificial neural networks as models of neural information processing. **Frontiers in Computational Neuroscience**, v. 11, dez. 2017. Disponível em: https://www.frontiersin.org/articles/10.3389/fncom.2017.00114/full. Acesso em: 10 out. 2022.

IRIS data set. **UCI Machine Learning Repository**, 1 jul. 1988. Disponível em: https://archive.ics.uci.edu/ml/datasets/iris. Acesso em: 10 out. 2022.

JARRAR, M. *et al*. MLP neural network classifier for medical image segmentation. *In*: INTERNATIONAL CONFERENCE COMPUTER GRAPHICS, IMAGING AND VISUALIZATION, 13., 2016. Londres. **Anais...** Londres: CGiV, 2016.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The Bulletin of Mathematical Biophysics**, v. 5, n. 4, p. 115-133, 1943.

MOREIRA, S. Rede neural perceptron multicamadas. **Medium**, 24 dez. 2018. Disponível em: https://medium.com/ensina-ai/rede-neural-perceptron-multicamadas-f9de8471f1a9. Acesso em: 10 out. 2022.

ROCHA, J. C. P. Introdução ao perceptron passo a passo. **GlobalMin**, 27, jul. 2017. Disponível em: https://juliocprocha.wordpress.com/2017/07/27/perceptron-para-classificacao-passo-a-passo/. Acesso em: 10 out. 2022.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, p. 386-408, 1958. Disponível em: https://doi.org/10.1037/h0042519. Acesso em: 10 out. 2022.

SILVA, A. M. Utilização de Redes Neurais Artificiais para Classificação de Spam.

Dissertação - CEFET-MG - Centro Federal de Educação Tecnológica de Minas Gerais, Programa de Pós-Graduação em Modelagem Matemática e Computacional. Belo Horizonte,

2009.

TANG, J.; DENG, C.; HUANG, G. B. Extreme learning machine for multilayer perceptron. **IEEE Transactions on Neural Networks and Learning Systems**, v. 27, n. 4, p. 809-821, 2016.

URAPYTHON COMMUNITY. Perceptron com Python, uma introdução. **Medium**, 21 nov. 2019. Disponível em: https://medium.com/@urapython.community/perceptron-com-python-uma-introdu%C3%A7%C3%A3o-f19aaf9e9b64#:~:text=Inventado%20em%201957%20por%20Frank,prever%20a%20qual%20grupo%20pertencer%C3%A1. Acesso em: 10 out. 2022.

WENG, D. *et al*. Techniques and applications of electrical equipment image processing based on improved MLP network using BP algorithm. *In*: INTERNATIONAL POWER ELECTRONICS AND MOTION CONTROL CONFERENCE, 8., 2016, [s. l.]. **Anais...** [S. l.]: IEEE, 2016.

Aula 3

CLAPPIS, A. M. Uma introdução às redes neurais convolucionais utilizando o Keras. **Data Hackers**, 12 jul. 2019. Disponível em: https://medium.com/data-hackers/uma-introdu%C3%A7%C3%A3o-as-redes-neurais-convolucionais-utilizando-o-keras-41ee8dcc033e . Acesso em: 13 out. 2022.

DATA SCIENCE ACADEMY. **Deep learning book**, [s. d.]. Disponível em: https://www.deeplearningbook.com.br/ . Acesso em: 13 out. 2022.

GAUI, R. Inteligência artificial - redes neurais: você sabe o que são redes convolucionais? **LinkedIn**, 15 nov. 2019. Disponível em: https://www.linkedin.com/pulse/intelig%C3%AAncia-artificial-redes-neurais-voc%C3%AA-sabe-o-que-gaui-msc/?originalSubdomain=pt . Acesso em: 13 out. 2022.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet classification with deep convolutional neural networks. **Communications of the ACM**, v. 60, n. 6 p. 84-90, 2017. Disponível em: https://doi.org/10.1145/3065386. Acesso em: 10 out. 2022.

LECUN, Y. *et al*. Gradient based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278-2324, 1998. Disponível em: http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf. Acesso em: 10 out. 2022.

LIU, T. *et al*. Sea surface object detection algorithm based on YOLO v4 fused with reverse depthwise separable convolution (RDSC) for USV. **Journal of Marine Science and Engineering**, v. 9, n. 7, p. 753, 2021. Disponível em: https://doi.org/10.3390/jmse9070753. Acesso em: 10 out. 2022.

REDES Neurais Convolucionais Profundas (AlexNet). **Dive into Deep Learning**, 24 dez. 2021. Disponível em: https://pt.d2l.ai/chapter_convolutional-modern/alexnet.html. Acesso em: 13 out. 2022.

SAHA, S. A comprehensive guide to convolutional neural networks — the ELI5 way. **Towards Data Science**, 15 dez. 2018. Disponível em: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 . Acesso em: 14 out 2022.

SILVA, F. M. *et al*. **Inteligência artificial**. Porto Alegre: Sagah, 2019. E-book. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788595029392/ . Acesso em: 14 out. 2022.

TUTORIAL completo no LeNet-5 | Guia para começar com CNNs. **Quish**, 25 maio 2022. Disponível em: https://pt.quish.tv/complete-tutorial-lenet-5-guide-begin-with-cnns . Acesso em: 10 out. 2022.

Aula 4

BENGIO, Y. Deep learning of representations: looking forward. **Statistical Language and Speech Processing**, p. 1-37, 2013. Disponível em: https://link.springer.com/chapter/10.1007/978-3-642-39593-2_1. Acesso em: 3 out. 2022.

BUDUMA, N.; LOCASCIO, N. **Fundamentals of deep learning**: designing next-generation machine. Califórnia: O'Reilly, 2017.

DENG, L.; YU, D. Deep learning: methods and applications. **Foundations and Trends® in Signal Processing**, v. 7, n. 3-4, p. 197-387, 2013. Disponível em: https://www.nowpublishers.com/article/DownloadEBook/SIG-039. Acesso em: 3 out. 2022.

GATED Recurrent Units (GRU). **Dive into Deep Learning**, 24 dez. 2021. Disponível em: https://pt.d2l.ai/chapter_recurrent-modern/gru.html. Acesso em: 3 out. 2022.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural Computation**, v. 9, n. 8, p. 1735-1780, 1997.

IMPLEMENTAÇÃO de redes neurais recorrentes do zero. **Dive into Deep Learning**, 24 dez. 2021. Disponível em: https://pt.d2l.ai/chapter_recurrent-neural-networks/rnn-scratch.html. Acesso em: 3 out. 2022.

MARTENS, J. Deep learning with Hessian-free optimization. *In*: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 27., 2010, Haifa, Israel. **Anais...** Haifa, Israel: ICML, 2010. Disponível em: https://www.cs.toronto.edu/~jmartens/docs/Deep_HessianFree.pdf. Acesso em: 18 out. 2022.

MEMÓRIA Longa de Curto Prazo (LSTM). **Dive into Deep Learning**, 24 dez. 2021. Disponível em: https://pt.d2l.ai/chapter_recurrent-modern/lstm.html. Acesso em: 3 out. 2022.

ZHANG, A. *et al*. **Dive into deep learning**. [S. l.]: Dive Into Deep Learning, 2021. Disponível em: https://d2l.ai/index.html. Acesso em: 18 out. 2022.

Imagem de capa: Storyset e ShutterStock.