Mestrado em Engenharia e Gestão Industrial

Inteligência Artificial aplicada na Indústria

Artificial Neural Network (ANN)

Daniel Nogueira

dnogueira@ipca.pt

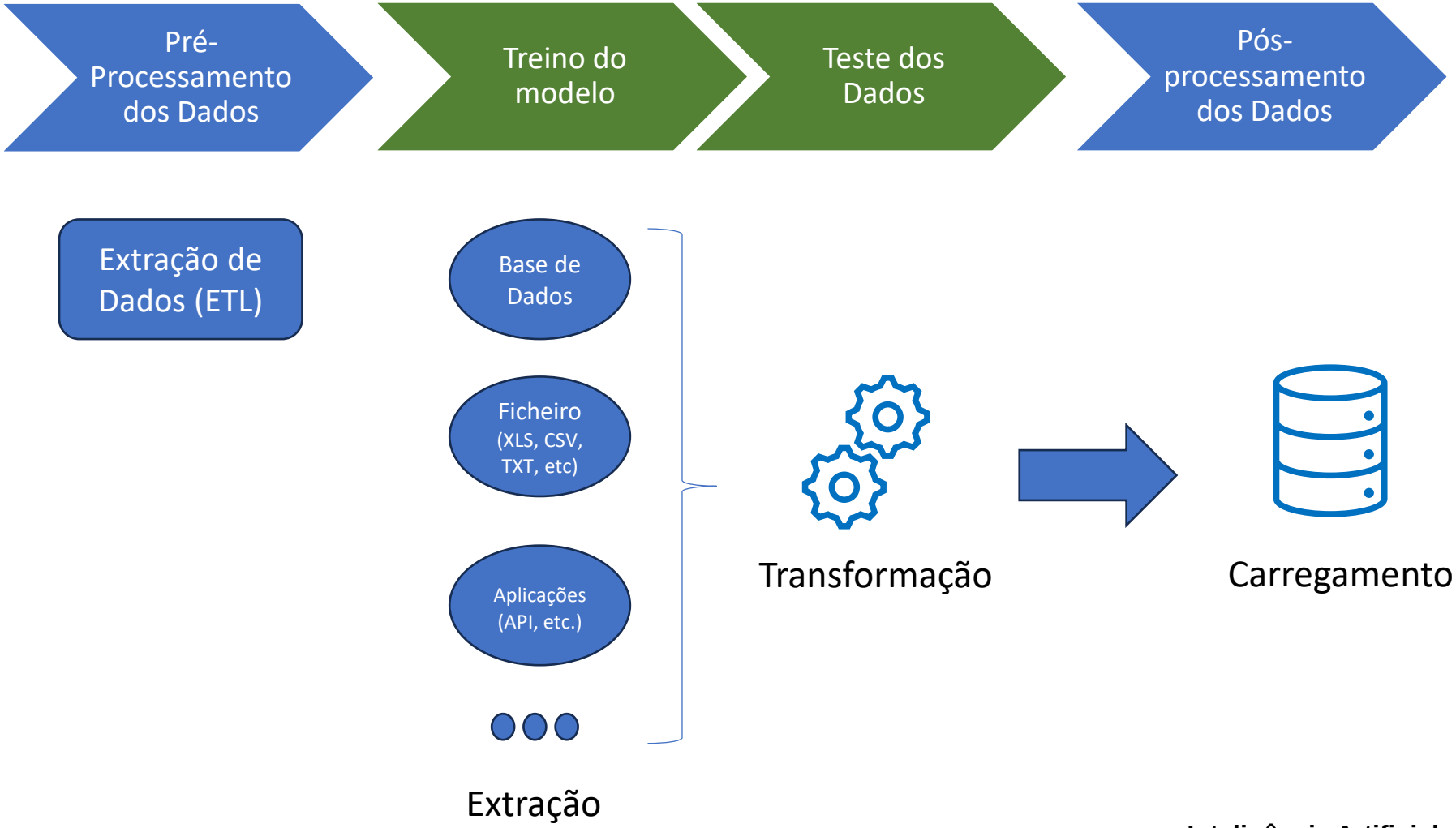
https://www.linkedin.com/in/danielfnogueira/

Concepts AI

Passos da Criação de um Modelo

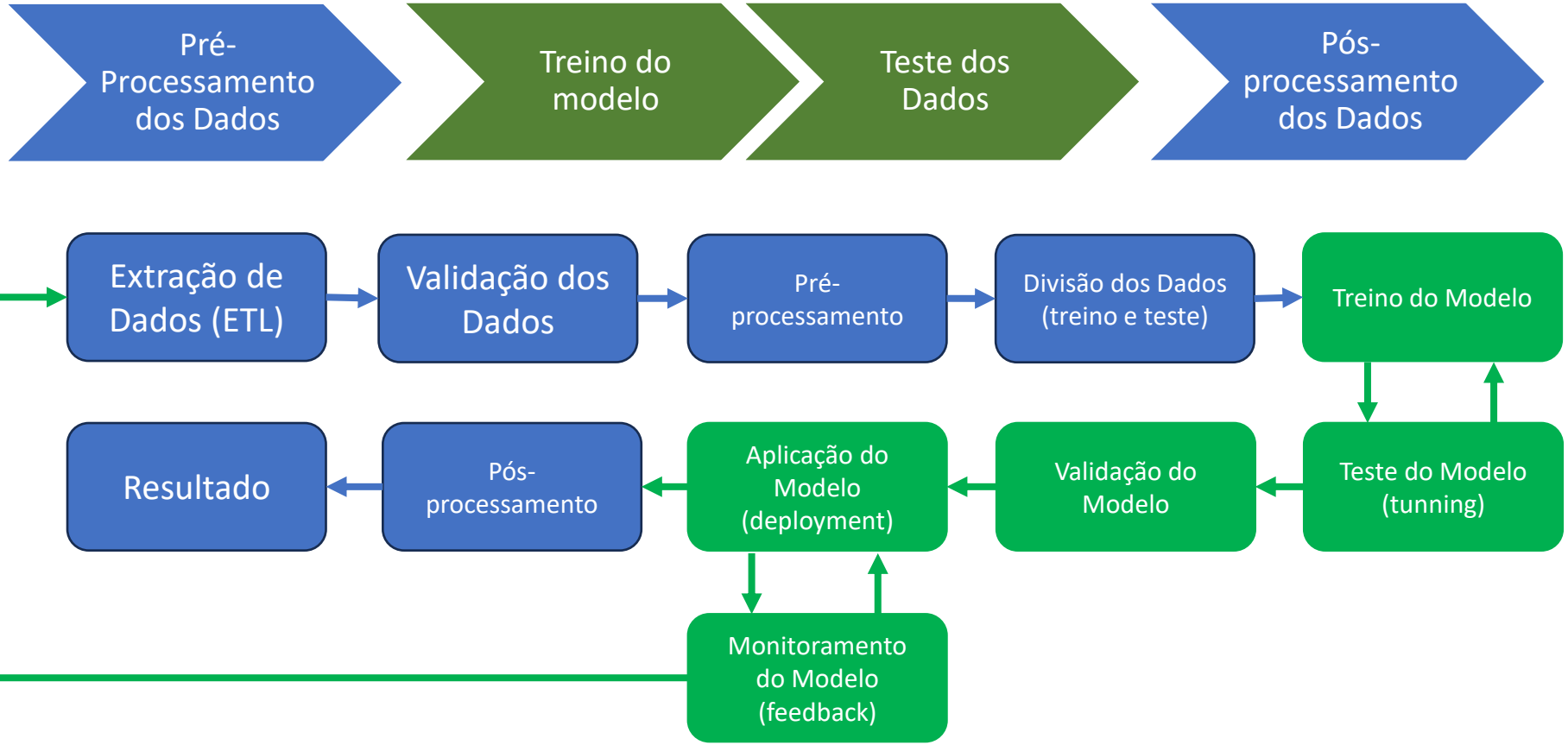
Concepts AI

Passos da Criação de um Modelo



Concepts AI

Passos da Criação de um Modelo



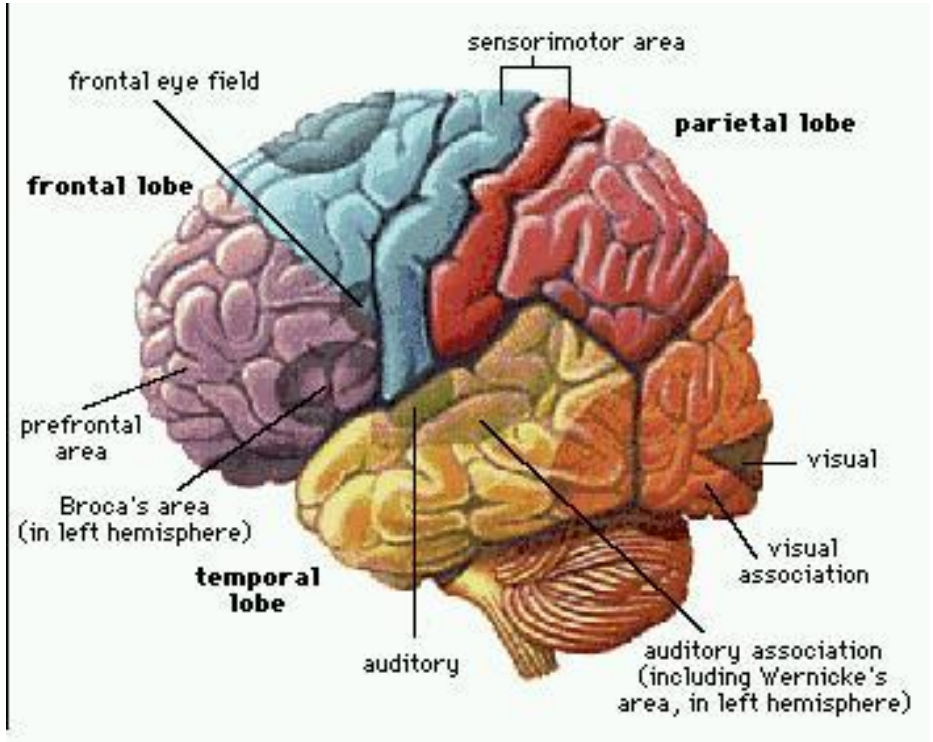
ANN

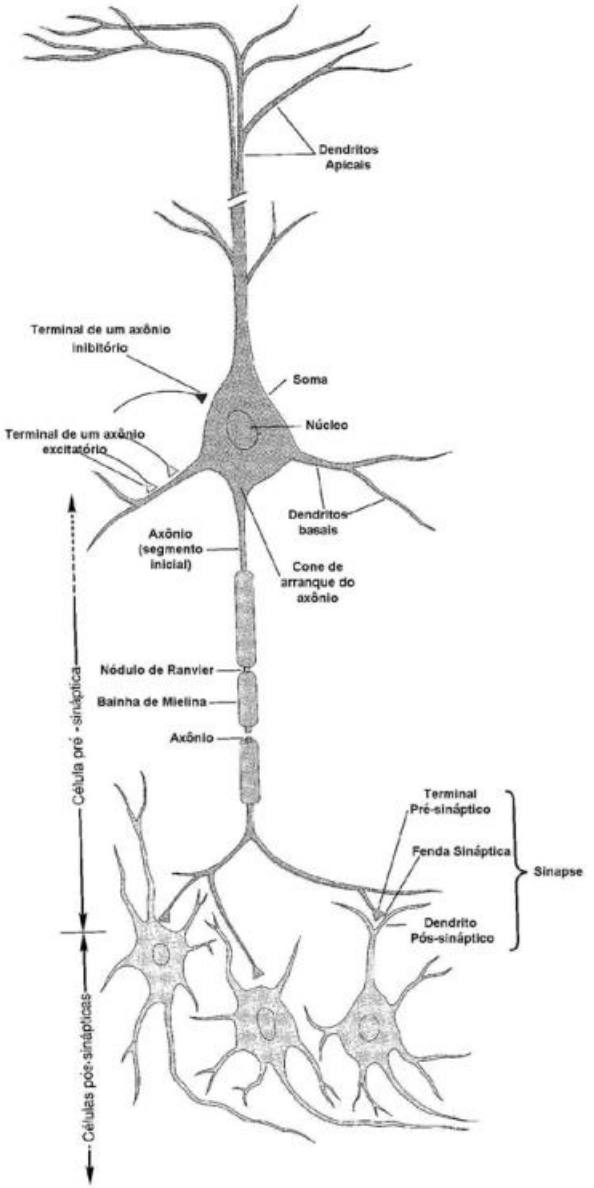
Introdução

- As redes neurais artificiais (ANN) modelam (*modelo matemático*) sistemas através de circuitos (*conexões*) que simulam o sistema nervoso humano.
- Procuram ter a capacidade que o sistema nervoso humano possui de aprender e agir perante as mais adversas situações apresentadas, bem como adquirir conhecimento através da experiência e da observação.
- Visa solucionar problemas de reconhecimento de padrões que geralmente são baseados em um conjunto de informações previamente conhecido.
- Uma grande rede neural artificial pode ter centenas ou milhares de unidades de processamento; já o cérebro de um mamífero pode ter muitos bilhões de neurônios.

ANN

Introdução





ANN

Introdução

- A primeira rede neural foi concebida por Warren McCulloch e Walter Pitts em 1943 (artigo sobre como os neurônios funcionavam e modelaram suas ideias criando uma rede neural simples com circuitos elétricos).
- Kunihiko Fukushima apresenta a primeira *rede neural multicamada* em 1975.
- O objetivo original da abordagem de rede neural era *criar um sistema computacional capaz de resolver problemas como um cérebro humano*
- Com o passar do tempo, os pesquisadores mudaram o foco e passaram a usar redes neurais para *resolver tarefas específicas*, desviando-se de uma abordagem estritamente biológica.
- ANNs têm oferecido suporte às mais diversas tarefas, incluindo *visão computacional, reconhecimento de fala, filtragem de redes sociais, jogos e diagnósticos médicos*.

ANN

“Nosso maior objetivo para redes neurais, ou para os modelos de redes neurais, é atingir a precisão máxima. Até você chegar nesse nível, você sabe que sempre pode fazer melhor.”

Gomez, Ivan (2022) – Zencos Consulting

ANN

Funcionamento

0

1

2

5

ANN

Funcionamento

0

1

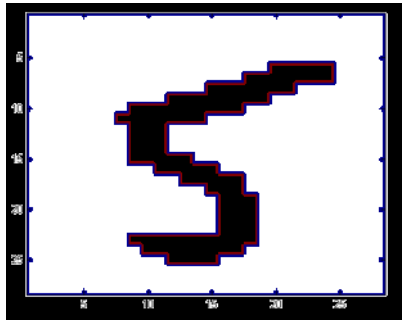
2

5

ANN

Funcionamento

Digit Recognition

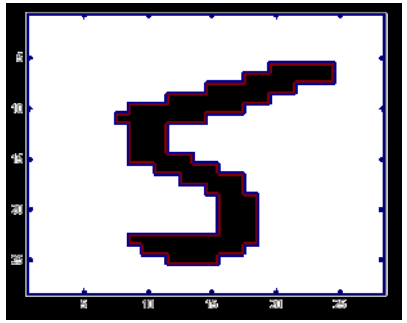


$X_1,\ldots,X_n \in \{0,1\}$ (Black vs. White pixels)

ANN

Funcionamento

Digit Recognition



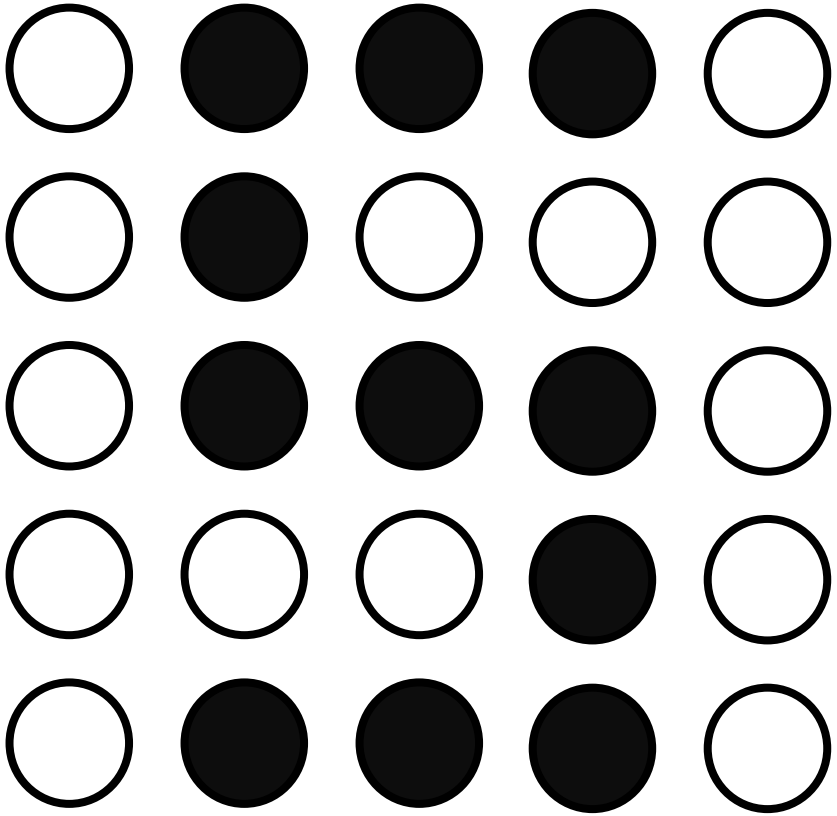
Classifier

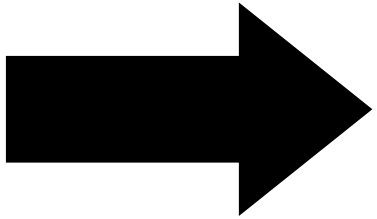
5

$X_1,\ldots,X_n \in \{0,1\}$ (Black vs. White pixels)

ANN

Funcionamento



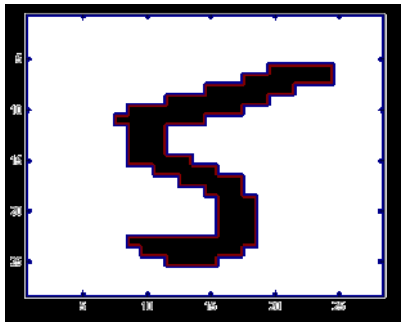


5

ANN

Funcionamento

Digit Recognition



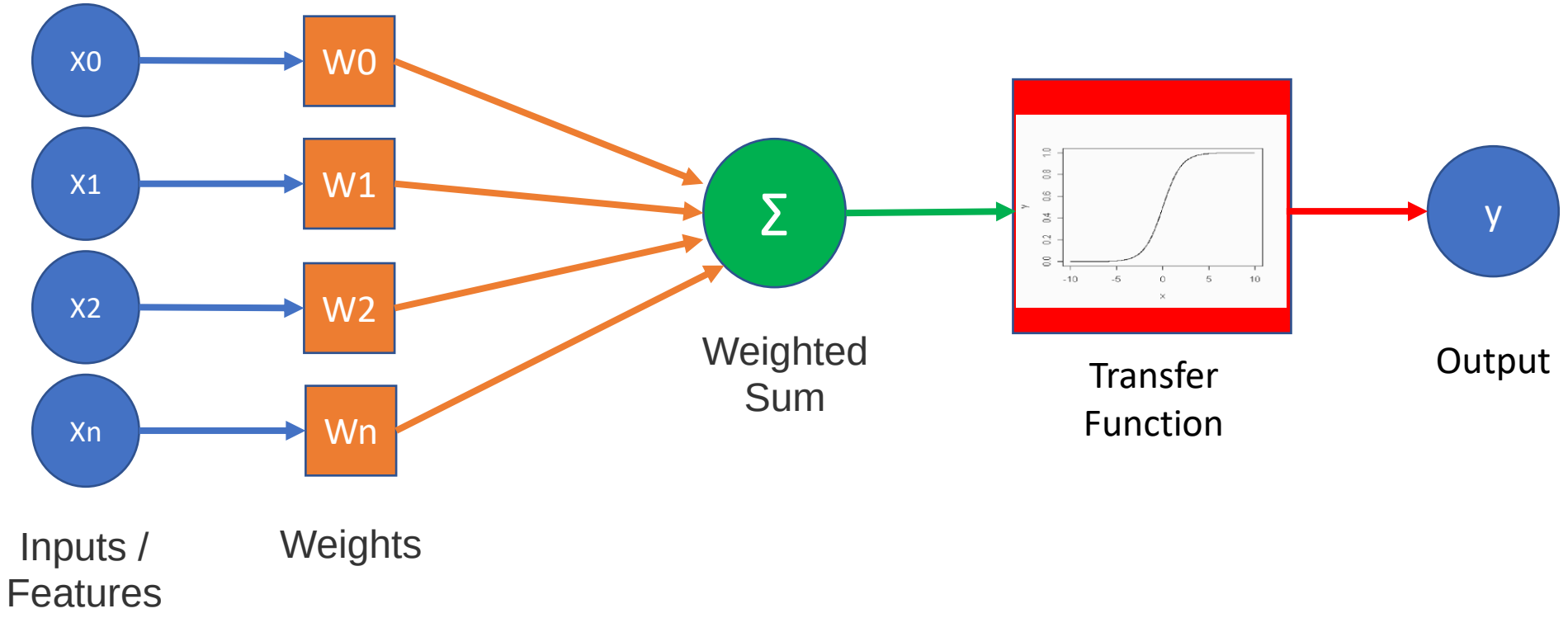
$X_1,\ldots,X_n \in [0,1]$

784 pixels

ANN

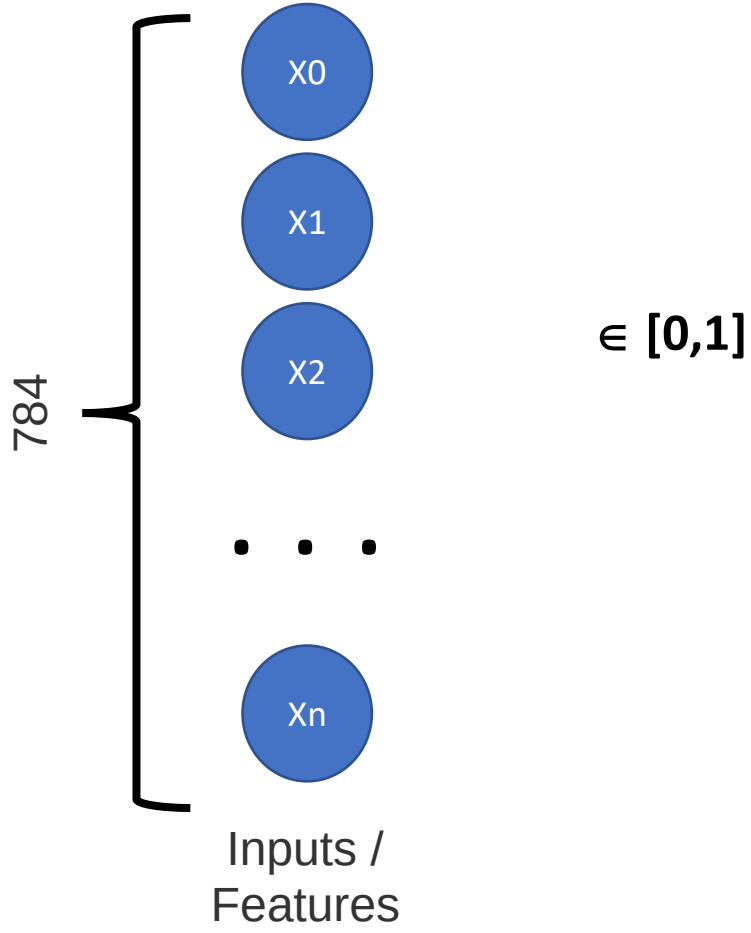
Funcionamento

Um neurônio de uma rede neural é um componente que *calcula a soma ponderada* de vários *inputs, aplica uma função* e *passa o resultado adiante*



ANN

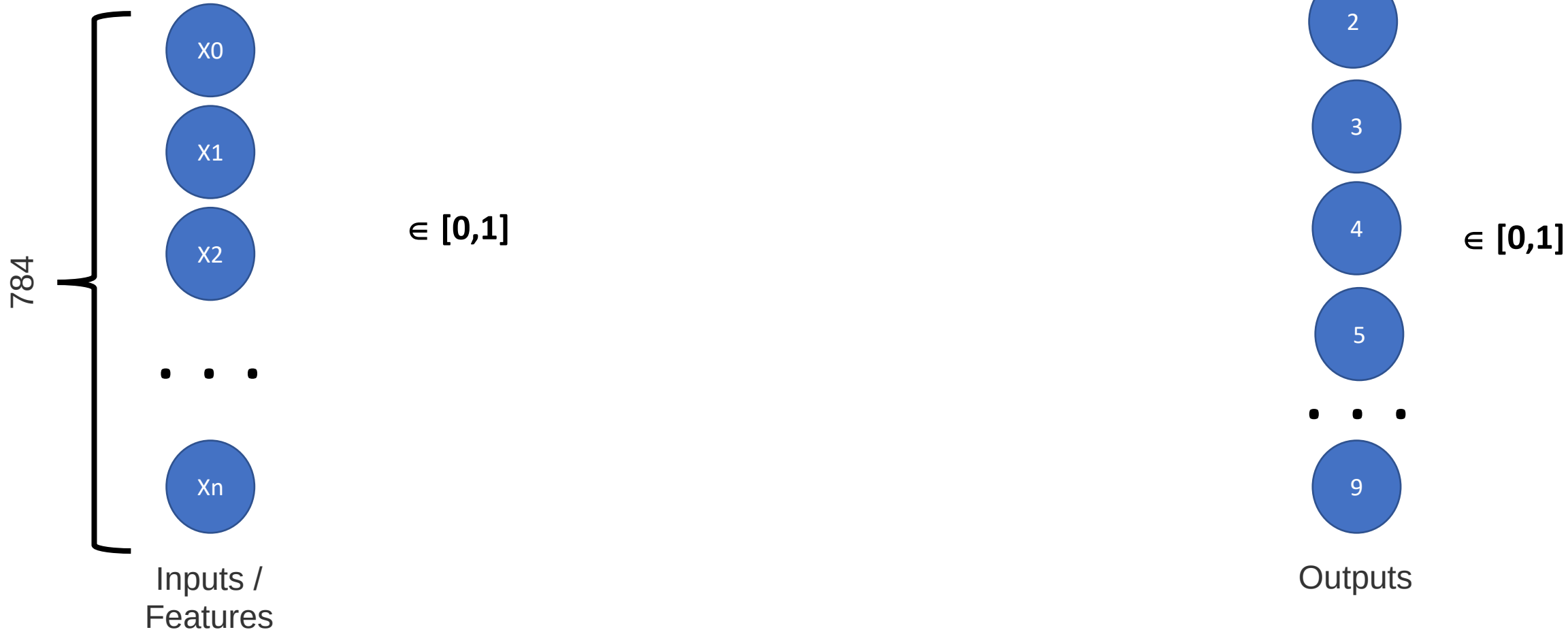
Funcionamento



$\in [0,1]$

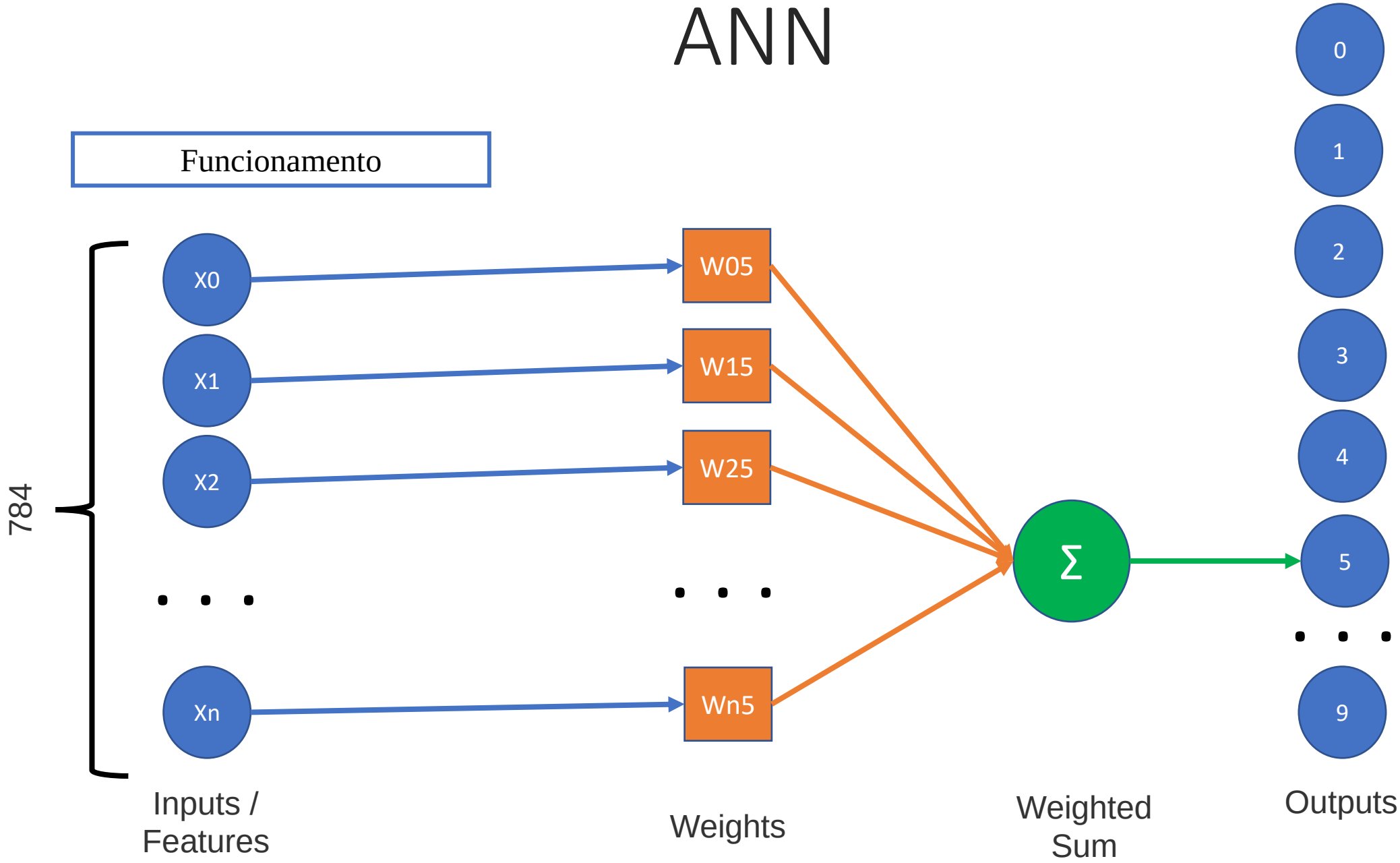
ANN

Funcionamento



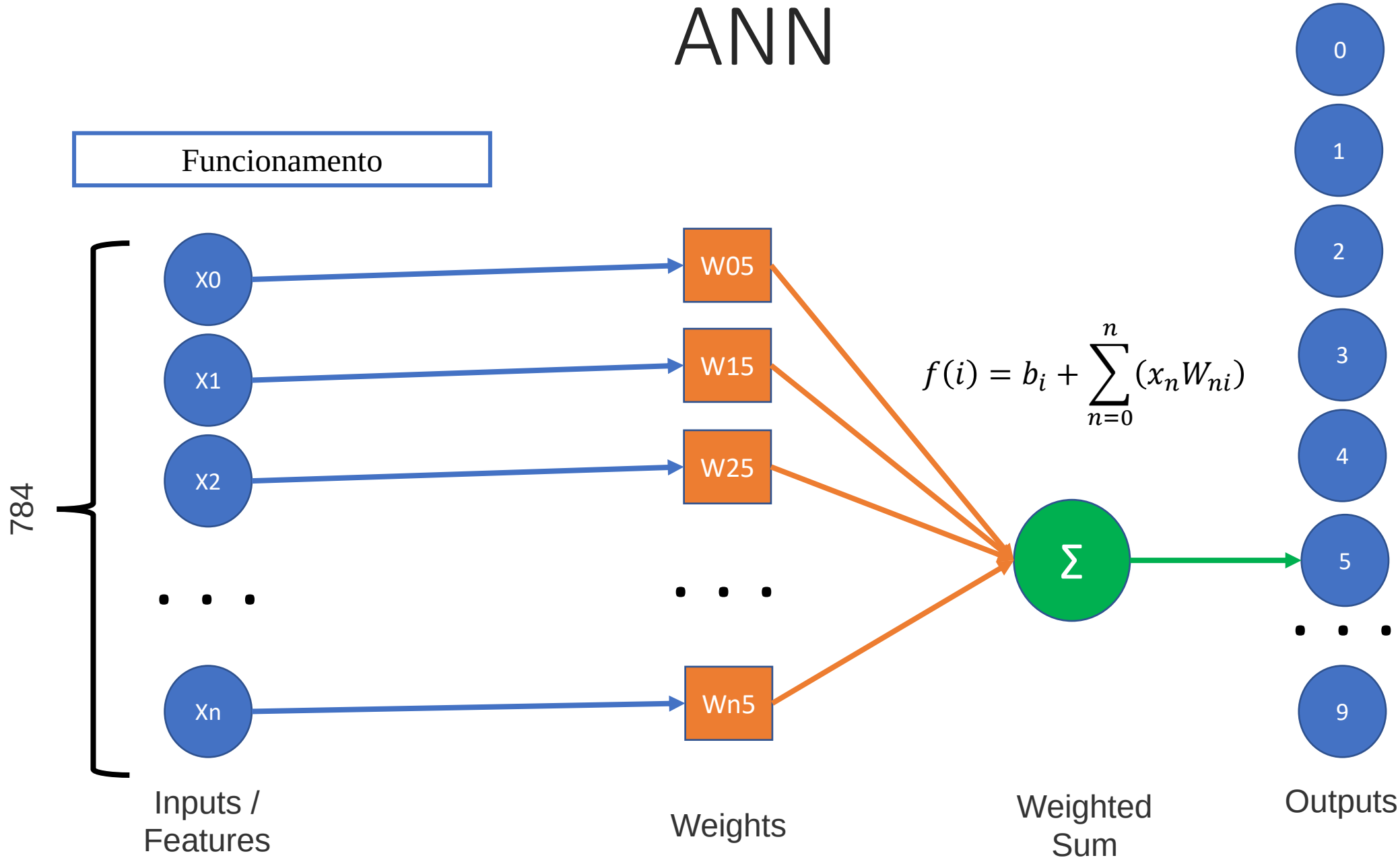
ANN

Funcionamento



ANN

Funcionamento

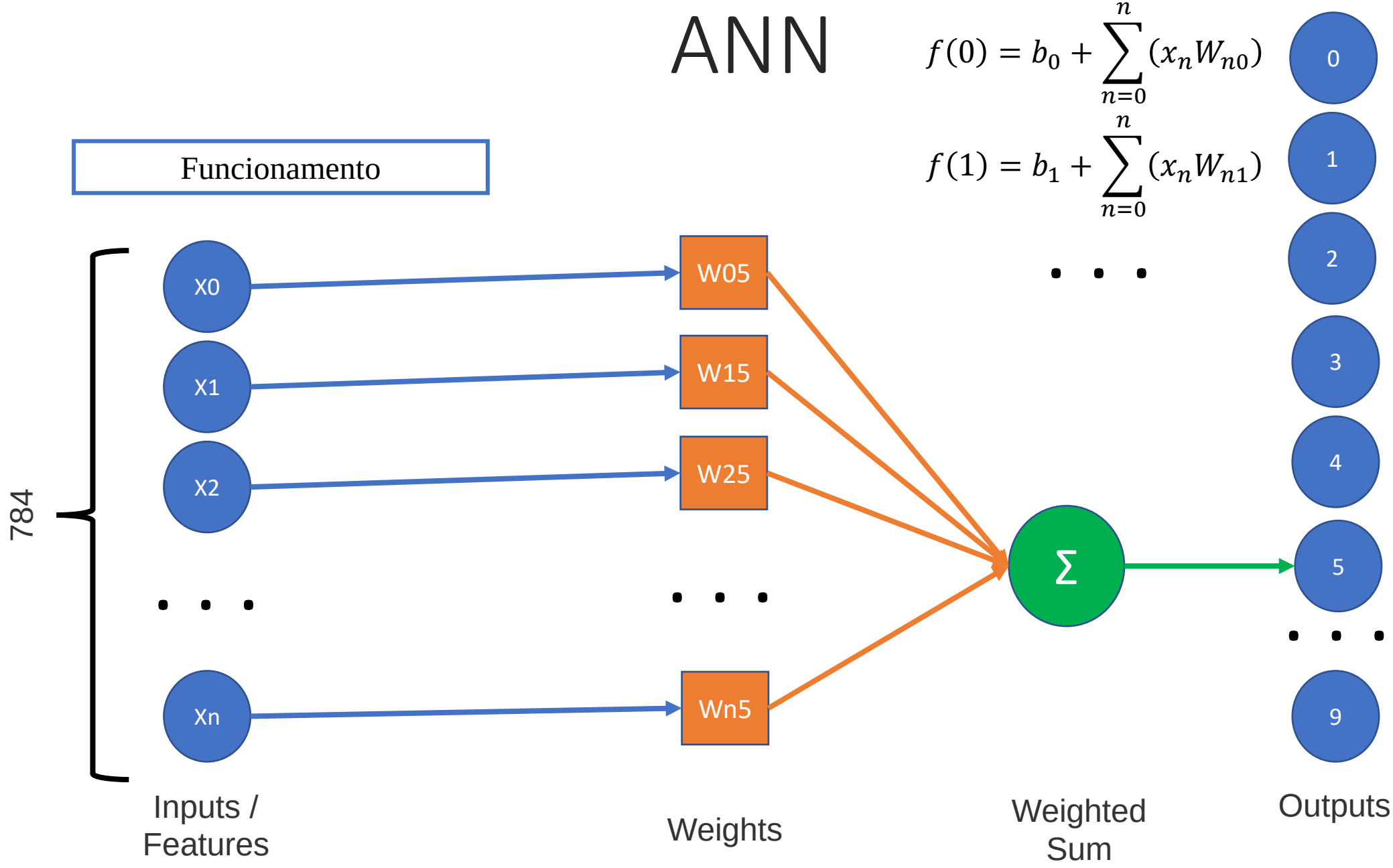


ANN

$$f(0) = b_0 + \sum_{n=0}^{n} (x_n W_{n0})$$

$$f(1) = b_1 + \sum_{n=0}^{n} (x_n W_{n1})$$

Funcionamento

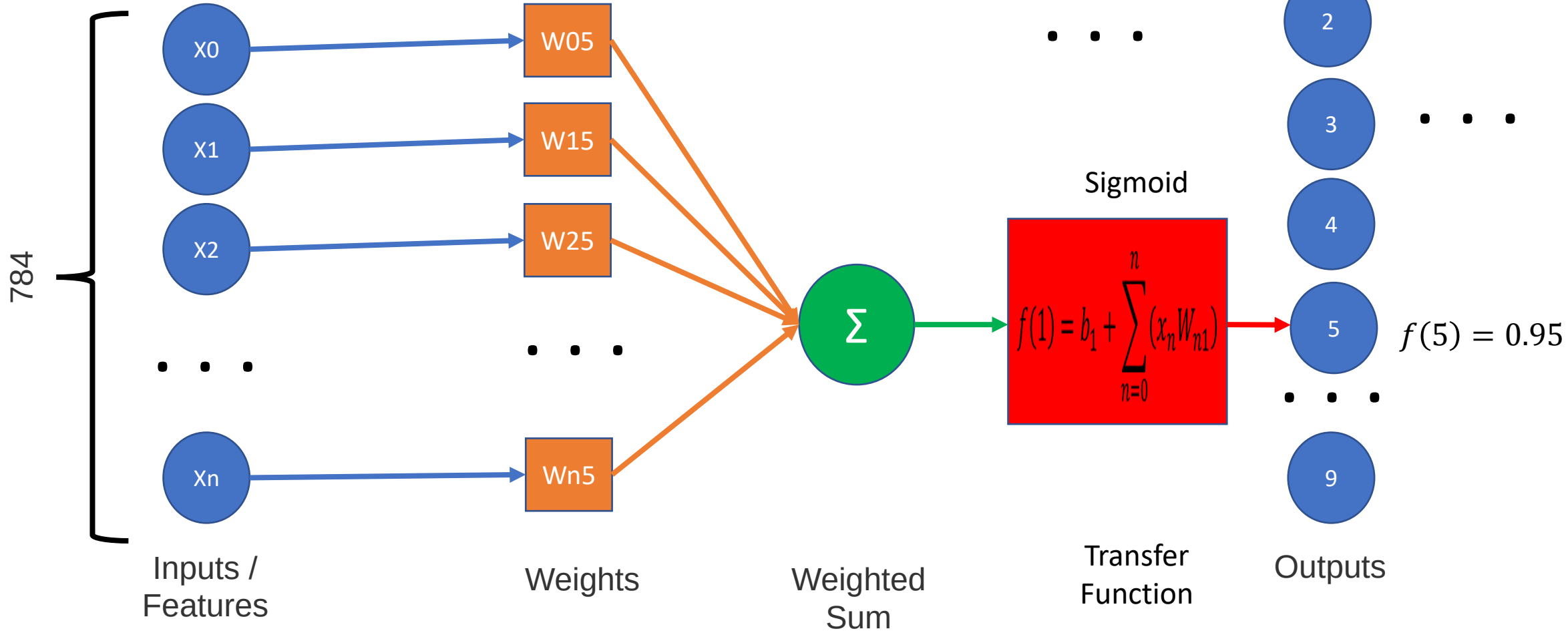


ANN

$$f(0) = b_0 + \sum_{n=0}^{n} (x_n W_{n0})$$

$$f(1) = b_1 + \sum_{n=0}^{n} (x_n W_{n1})$$

Funcionamento



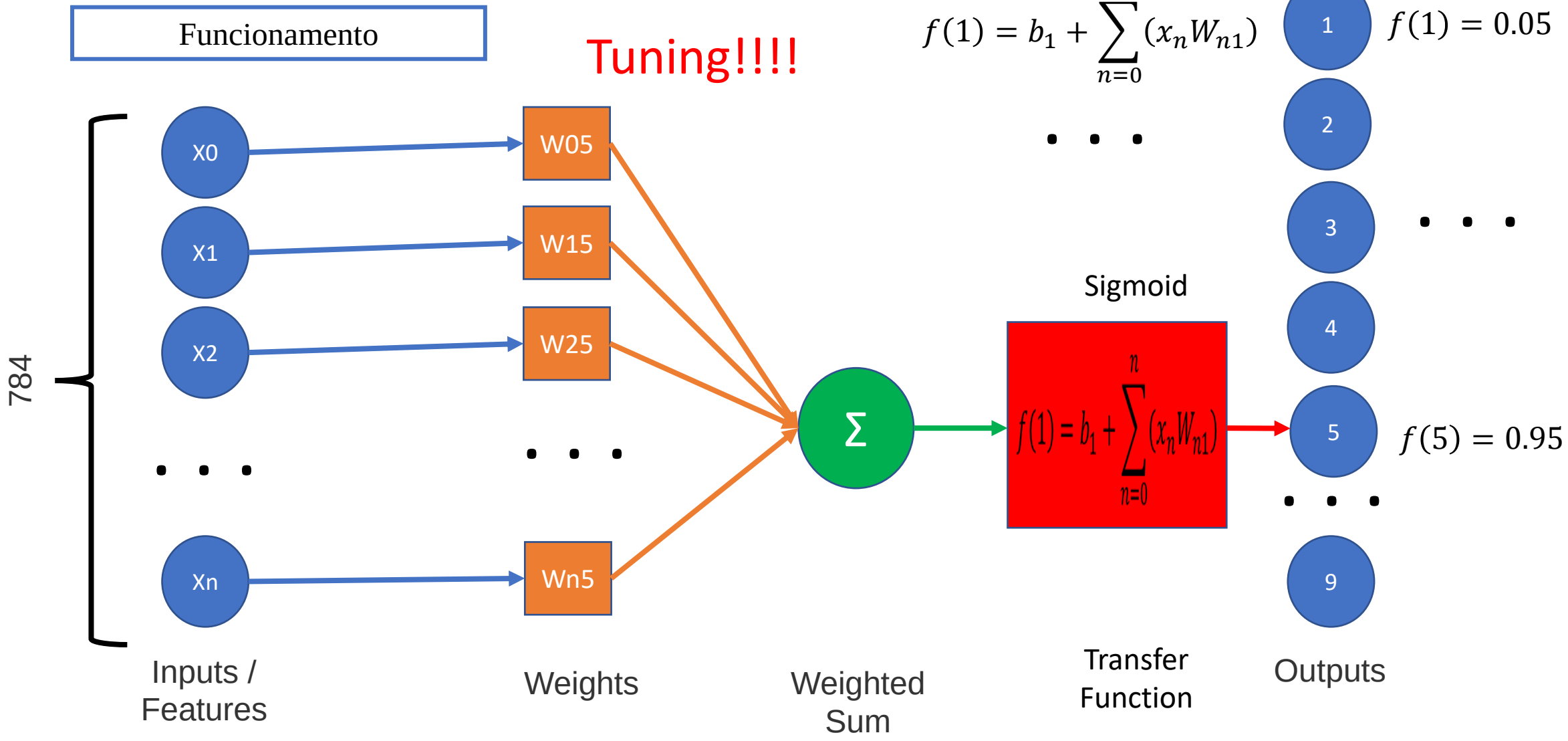
ANN

$$f(0) = b_0 + \sum_{n=0}^{n} (x_n W_{n0})$$

$$f(1) = b_1 + \sum_{n=0}^{n} (x_n W_{n1})$$

Funcionamento

Tuning!!!!



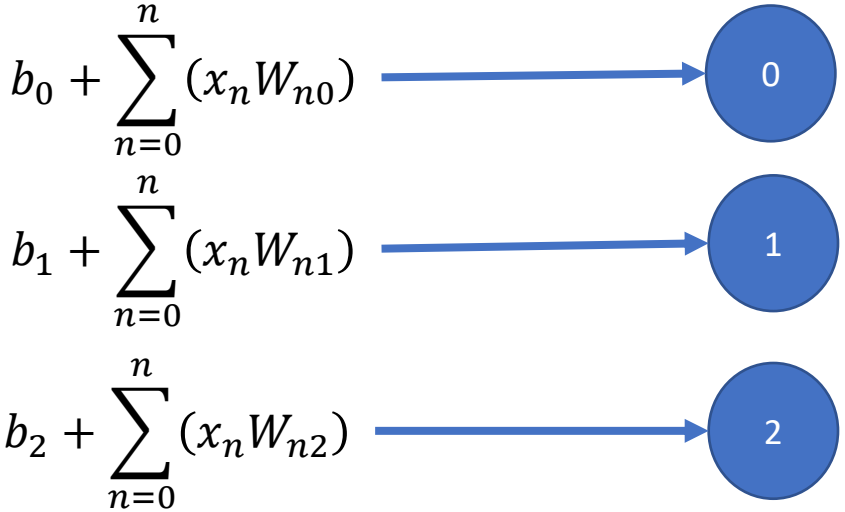
ANN

Funcionamento

$$f(0) = b_0 + \sum_{n=0}^{n} (x_n W_{n0})$$

$$f(1) = b_1 + \sum_{n=0}^{n} (x_n W_{n1})$$

$$f(2) = b_2 + \sum_{n=0}^{n} (x_n W_{n2})$$



Outputs

784 * 10 pesos + 10 bias = 7850 variáveis

ANN

Funcionamento

$$f(0) = b_0 + \sum_{n=0}^{n} (x_n W_{n0})$$

$$f(1) = b_1 + \sum_{n=0}^{n} (x_n W_{n1})$$

$$f(2) = b_2 + \sum_{n=0}^{n} (x_n W_{n2})$$

. . .

0

1

2

. . .

Outputs

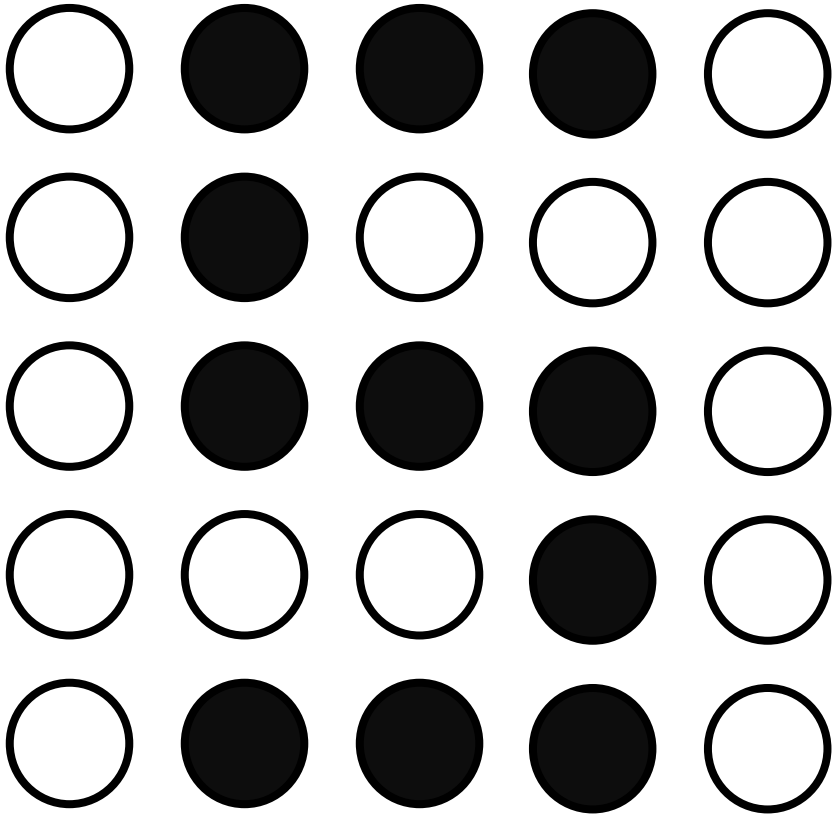
784 * 10 pesos + 10 bias = 7850 variáveis

Pesos podem ser valores NEGATIVOS!!!!

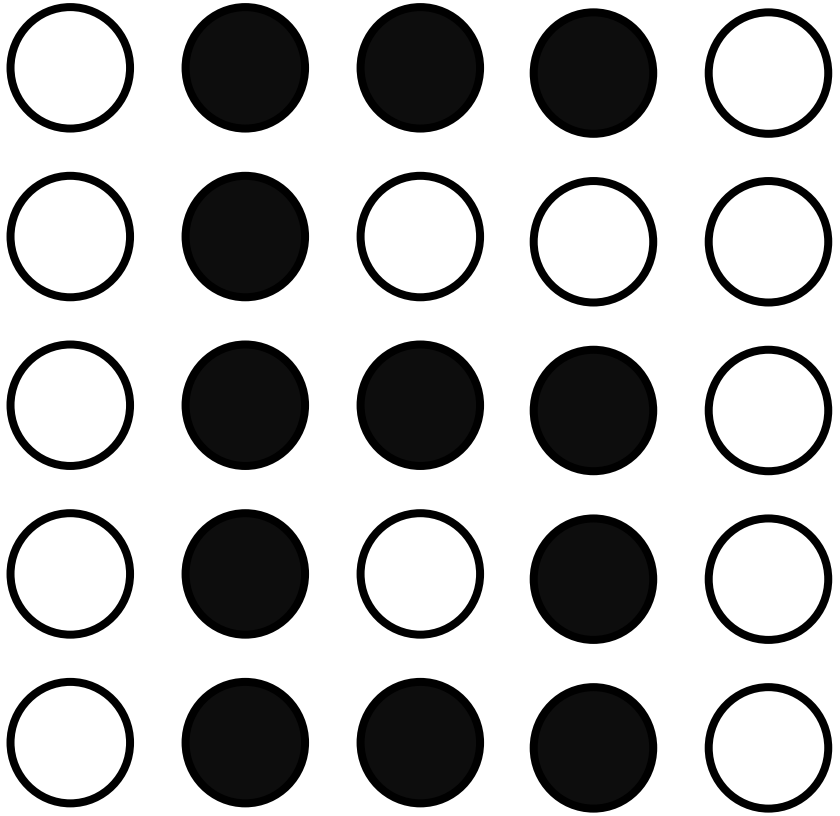
ANN

Funcionamento

5



6



ANN

Outputs

2

0 $f(0) = 0{,}26$

1 $f(1) = 0{,}05$

2 $f(2) = 0{,}18$

3 $f(3) = 0{,}39$

4 $f(4) = 0{,}23$

5 $f(5) = 0{,}08$

6 $f(6) = 0{,}74$

7 $f(7) = 0{,}52$

8 $f(8) = 0{,}41$

9 $f(9) = 0{,}33$

ANN

Outputs

2

0	$f(0) = 0{,}26$	$f(0) = 0.00$
1	$f(1) = 0{,}05$	$f(1) = 0.00$
2	$f(2) = 0{,}18$	$\boldsymbol{f(2) = 1{,}00}$
3	$f(3) = 0{,}39$	$f(3) = 0.00$
4	$f(4) = 0{,}23$	$f(4) = 0.00$
5	$f(5) = 0{,}08$	$f(5) = 0.00$
6	$\boldsymbol{f(6) = 0{,}74}$	$f(6) = 0.00$
7	$f(7) = 0{,}52$	$f(7) = 0.00$
8	$f(8) = 0{,}41$	$f(8) = 0.00$
9	$f(9) = 0{,}33$	$f(9) = 0.00$

$$f(2) = b_2 + (x_0 W_{02}) + (x_1 W_{12}) + (x_2 W_{22}) + \ \ldots$$

ANN

2

Outputs

0	$f(0) = 0{,}26$	$f(0) = 0.00$
1	$f(1) = 0{,}05$	$f(1) = 0.00$
2	$f(2) = 0{,}18$	$\boldsymbol{f(2) = 1{,}00}$
3	$f(3) = 0{,}39$	$f(3) = 0.00$
4	$f(4) = 0{,}23$	$f(4) = 0.00$
5	$f(5) = 0{,}08$	$f(5) = 0.00$
6	$\boldsymbol{f(6) = 0{,}74}$	$f(6) = 0.00$
7	$f(7) = 0{,}52$	$f(7) = 0.00$
8	$f(8) = 0{,}41$	$f(8) = 0.00$
9	$f(9) = 0{,}33$	$f(9) = 0.00$

$$f(2) = b_2 + (x_0 W_{02}) + (x_1 W_{12}) + (x_2 W_{22}) + \ \ldots$$

Função de Custos

$$CUSTO_2 = \Sigma Error^2$$

$$CUSTO_2 = (0.26 - 0)^2 + (0.05 - 0)^2 + (0.18 - 1)^2 + \ldots = 2{,}05$$

ANN

2

Outputs

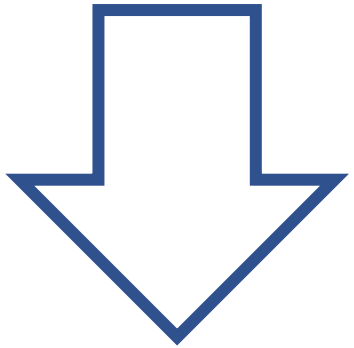
0	$f(0) = 0{,}26$	$f(0) = 0.00$
1	$f(1) = 0{,}05$	$f(1) = 0.00$
2	$f(2) = 0{,}18$	$\boldsymbol{f(2) = 1{,}00}$
3	$f(3) = 0{,}39$	$f(3) = 0.00$
4	$f(4) = 0{,}23$	$f(4) = 0.00$
5	$f(5) = 0{,}08$	$f(5) = 0.00$
6	$\boldsymbol{f(6) = 0{,}74}$	$f(6) = 0.00$
7	$f(7) = 0{,}52$	$f(7) = 0.00$
8	$f(8) = 0{,}41$	$f(8) = 0.00$
9	$f(9) = 0{,}33$	$f(9) = 0.00$

$$f(2) = b_2 + (x_0 W_{02}) + (x_1 W_{12}) + (x_2 W_{22}) + \ \ldots$$

Função de Custos

$$CUSTO_2 = \Sigma Error^2$$

$$CUSTO_2 = (0.26 - 0)^2 + (0.05 - 0)^2 + (0.18 - 1)^2 + \ldots = 2{,}05$$



ANN

2

Outputs

0	$f(0) = 0{,}26$	$f(0) = 0.00$
1	$f(1) = 0{,}05$	$f(1) = 0.00$
2	$f(2) = 0{,}18$	$\boldsymbol{f(2) = 1{,}00}$
3	$f(3) = 0{,}39$	$f(3) = 0.00$
4	$f(4) = 0{,}23$	$f(4) = 0.00$
5	$f(5) = 0{,}08$	$f(5) = 0.00$
6	$\boldsymbol{f(6) = 0{,}74}$	$f(6) = 0.00$
7	$f(7) = 0{,}52$	$f(7) = 0.00$
8	$f(8) = 0{,}41$	$f(8) = 0.00$
9	$f(9) = 0{,}33$	$f(9) = 0.00$

$$f(2) = b_2 + (x_0 W_{02}) + (x_1 W_{12}) + (x_2 W_{22}) + \ \ldots$$

Função de Custos

$$CUSTO_2 = \Sigma Error^2$$

$$CUSTO_2 = (0.26 - 0)^2 + (0.05 - 0)^2 + (0.18 - 1)^2 + \ldots = 2{,}05$$

$$CUSTO_5 = 4{,}15$$

. . .

ANN

2

Outputs

0	$f(0) = 0{,}26$	$f(0) = 0.00$
1	$f(1) = 0{,}05$	$f(1) = 0.00$
2	$f(2) = 0{,}18$	$\boldsymbol{f(2) = 1{,}00}$
3	$f(3) = 0{,}39$	$f(3) = 0.00$
4	$f(4) = 0{,}23$	$f(4) = 0.00$
5	$f(5) = 0{,}08$	$f(5) = 0.00$
6	$\boldsymbol{f(6) = 0{,}74}$	$f(6) = 0.00$
7	$f(7) = 0{,}52$	$f(7) = 0.00$
8	$f(8) = 0{,}41$	$f(8) = 0.00$
9	$f(9) = 0{,}33$	$f(9) = 0.00$

$$f(2) = b_2 + (x_0 W_{02}) + (x_1 W_{12}) + (x_2 W_{22}) + \ \ldots$$

Função de Custos

$$CUSTO_2 = \Sigma Error^2$$

$$CUSTO_2 = (0.26 - 0)^2 + (0.05 - 0)^2 + (0.18 - 1)^2 + \ldots = 2{,}05$$

$$CUSTO_5 = 4{,}15$$

. . .

$$CUSTO_{TOTAL} = \Sigma\, CUSTO_n$$

ANN

2

Outputs

0	$f(0) = 0{,}26$	$f(0) = 0.00$
1	$f(1) = 0{,}05$	$f(1) = 0.00$
2	$f(2) = 0{,}18$	$\boldsymbol{f(2) = 1{,}00}$
3	$f(3) = 0{,}39$	$f(3) = 0.00$
4	$f(4) = 0{,}23$	$f(4) = 0.00$
5	$f(5) = 0{,}08$	$f(5) = 0.00$
6	$\boldsymbol{f(6) = 0{,}74}$	$f(6) = 0.00$
7	$f(7) = 0{,}52$	$f(7) = 0.00$
8	$f(8) = 0{,}41$	$f(8) = 0.00$
9	$f(9) = 0{,}33$	$f(9) = 0.00$

$$f(2) = b_2 + (x_0 W_{02}) + (x_1 W_{12}) + (x_2 W_{22}) + \ \ldots$$

Função de Custos

$$CUSTO_2 = \Sigma Error^2$$

$$CUSTO_2 = (0.26 - 0)^2 + (0.05 - 0)^2 + (0.18 - 1)^2 + \ldots = 2{,}05$$

$$CUSTO_5 = 4{,}15$$

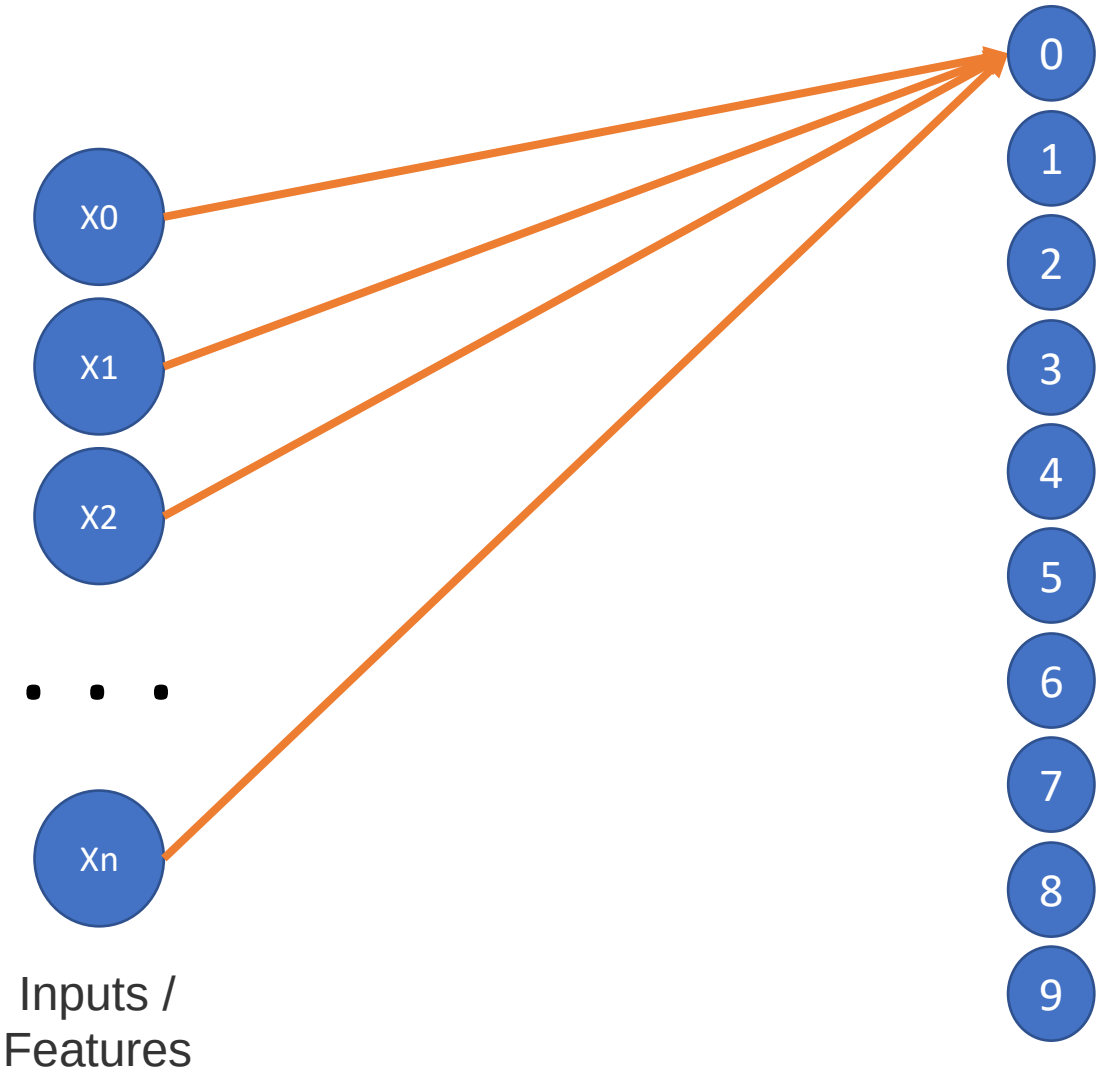
. . .

10.000 images

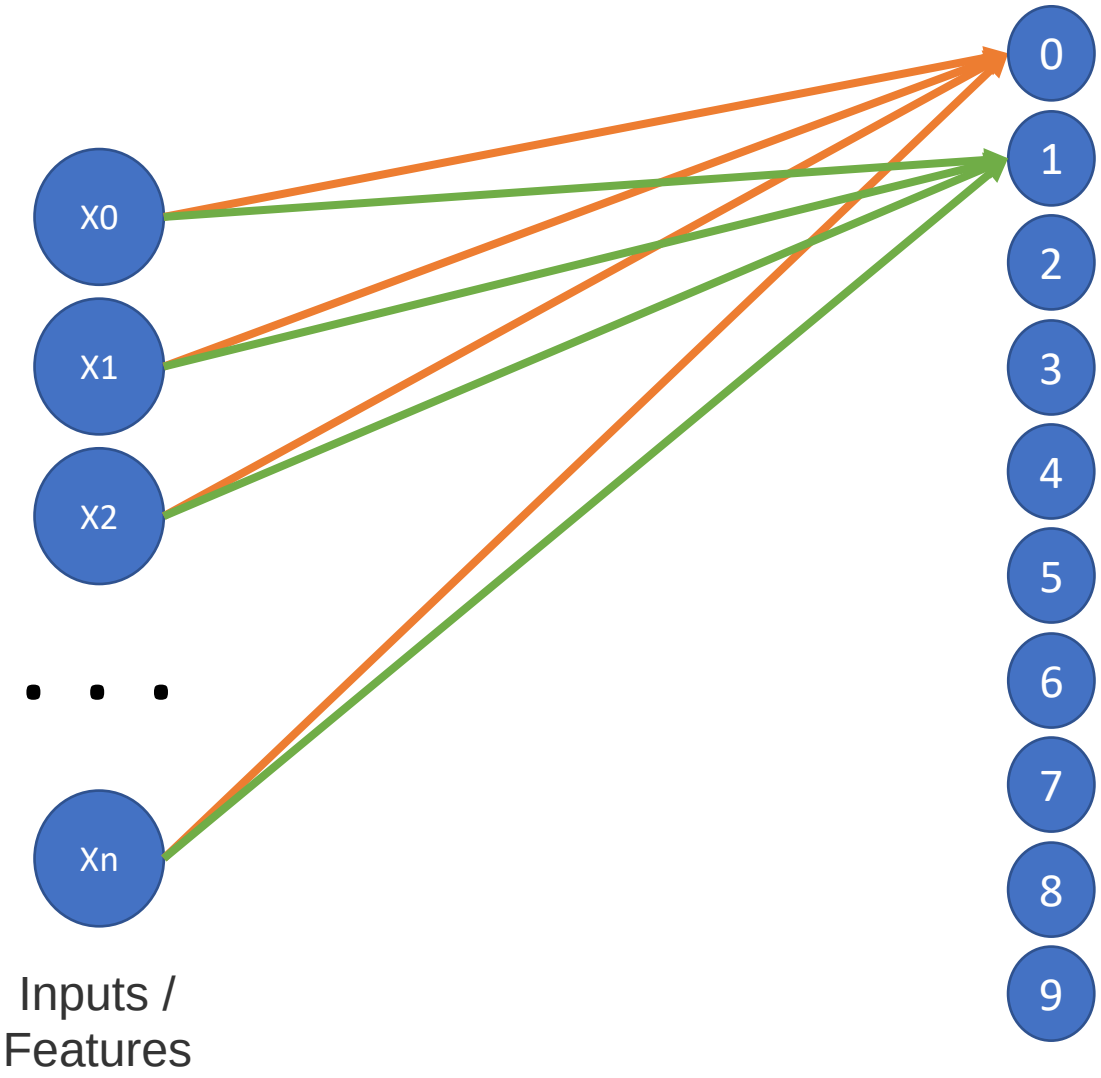
$$CUSTO_{TOTAL} = \Sigma\, CUSTO_n$$

$$CUSTO_{TOTAL} = 30.548$$

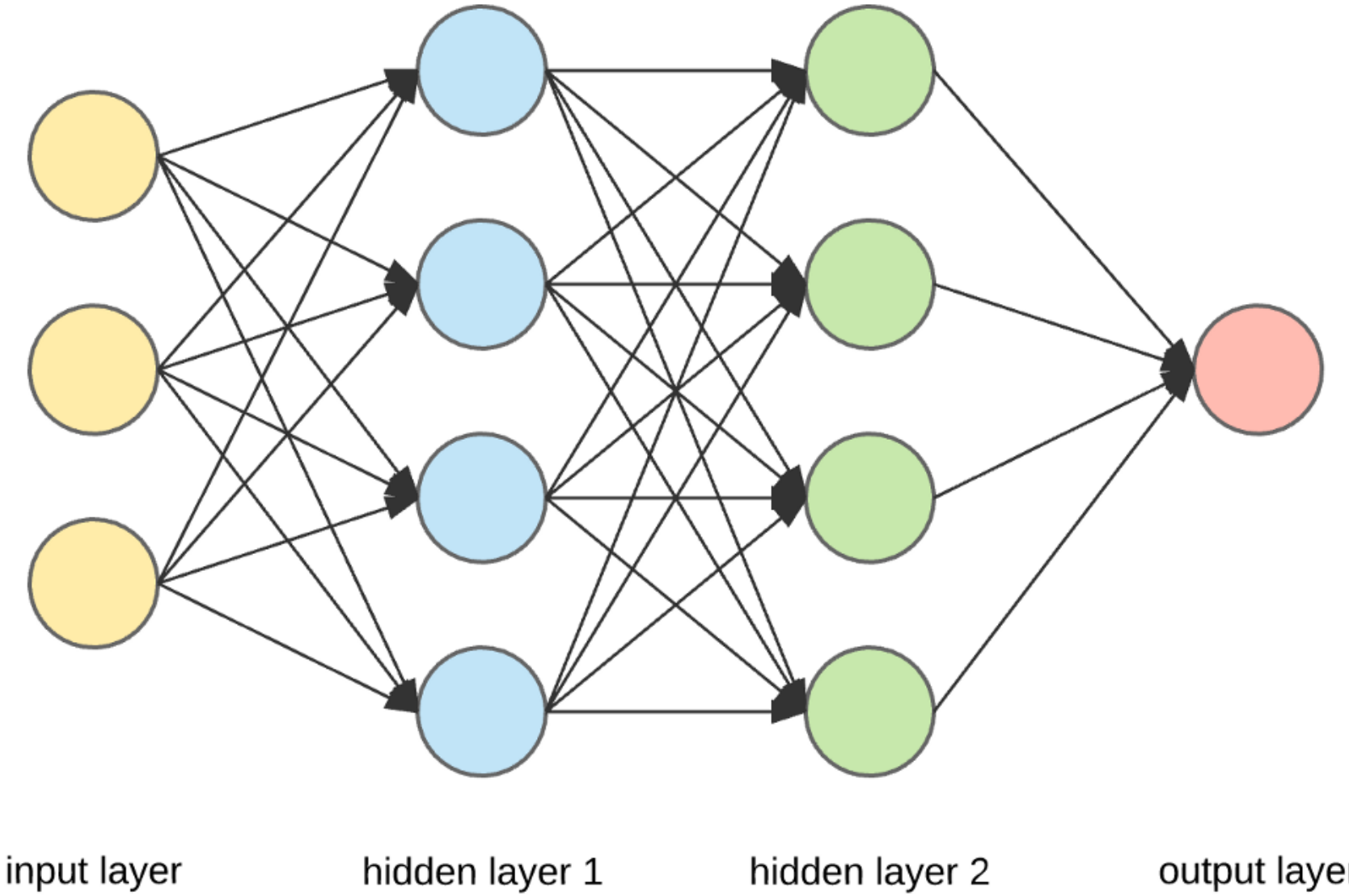
ANN



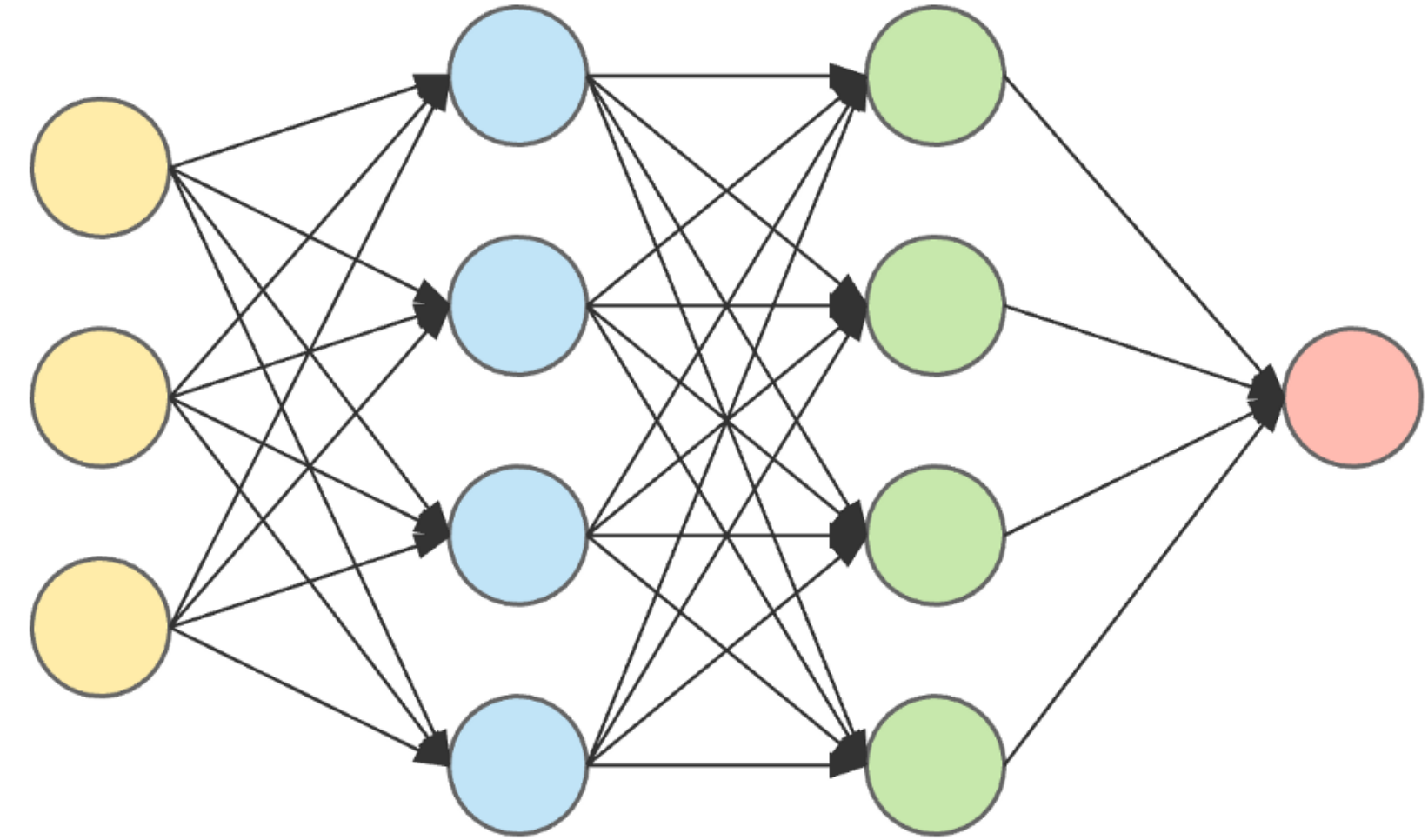
ANN



ANN

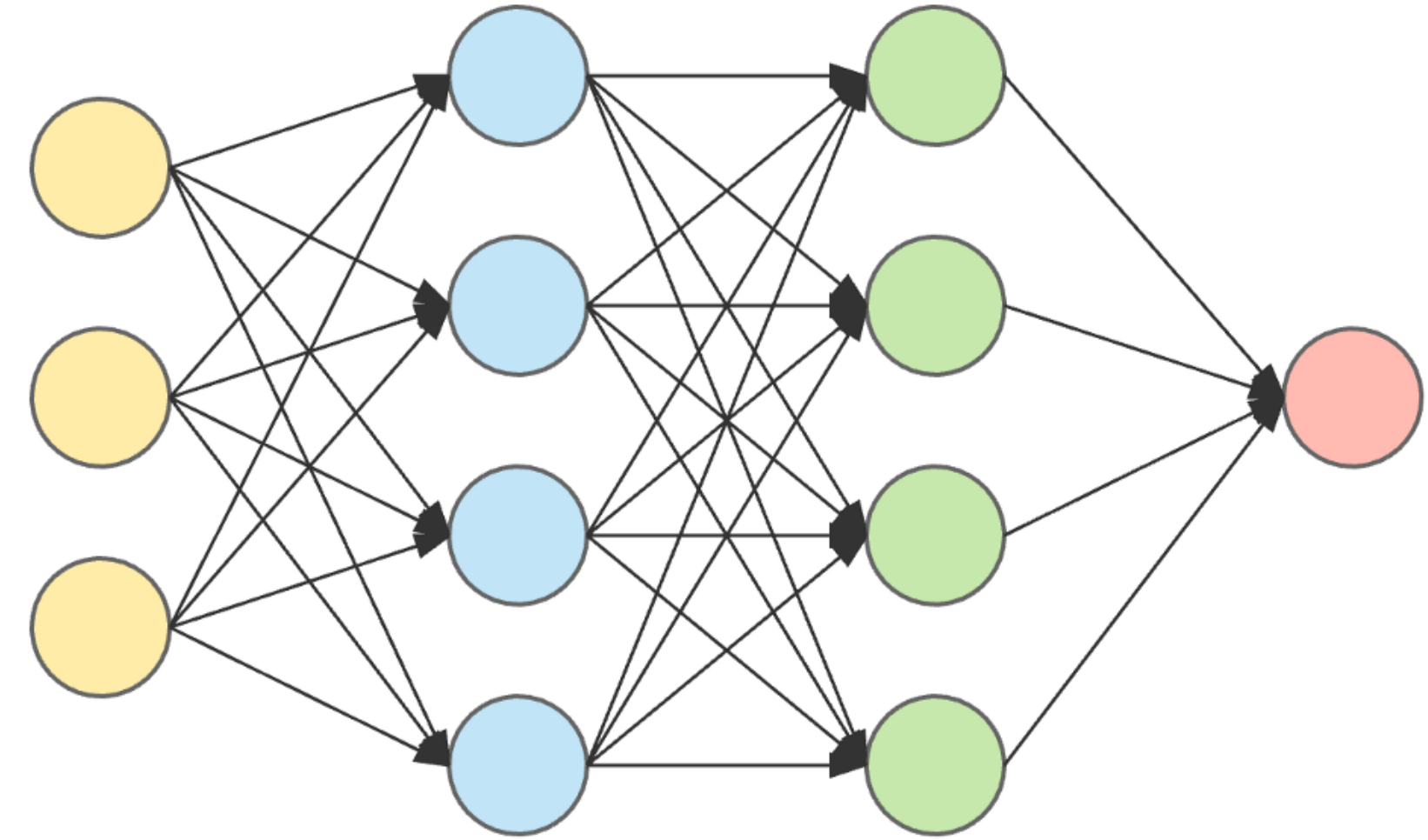


ANN


784
input layer
30
hidden layer 1
30
hidden layer 2
10
output layer

ANN

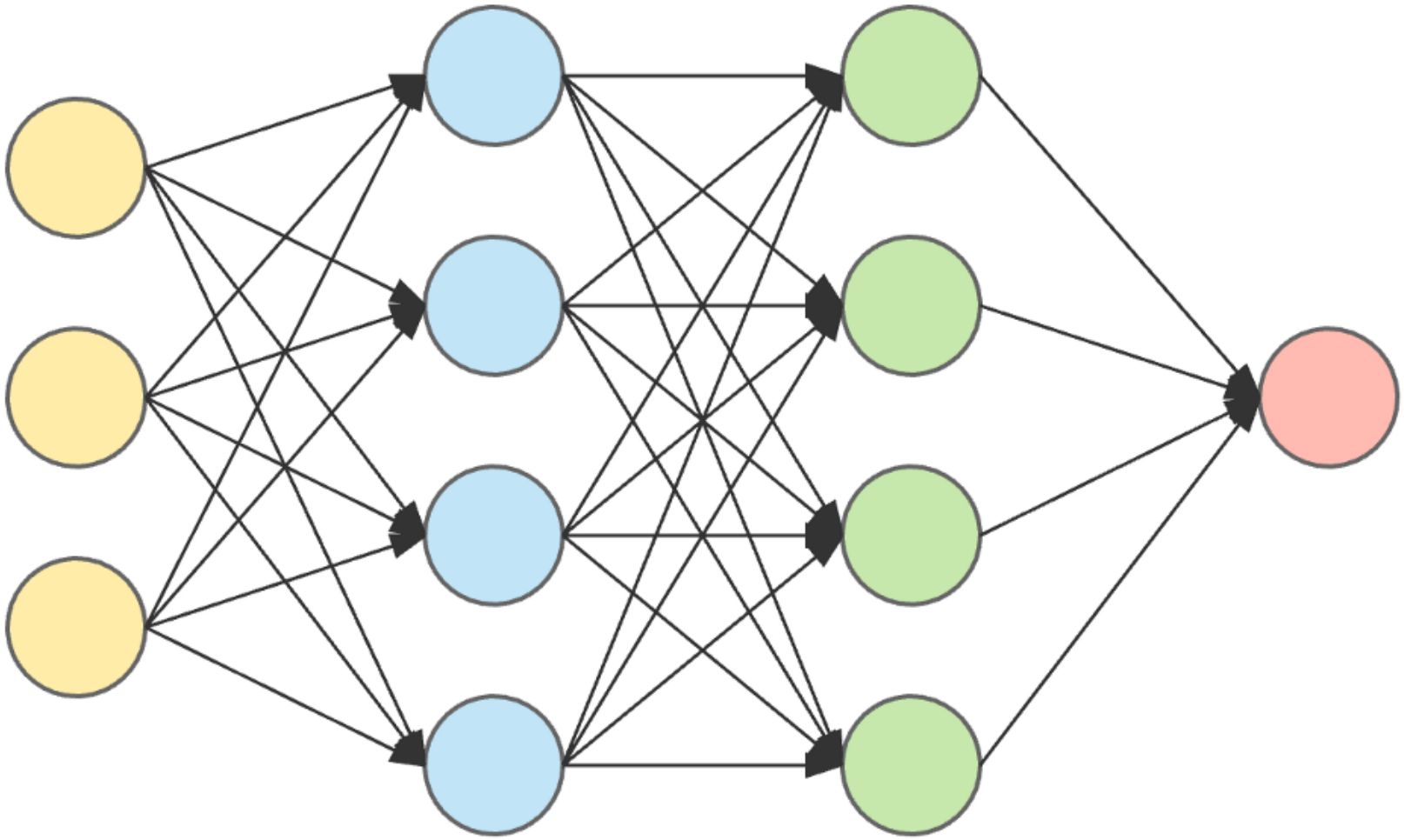
784 * 10 pesos + 10 bias = 7850 variáveis



ANN

784 * 10 pesos + 10 bias = 7850 variáveis

(784 * 30 pesos + 30 bias) +
(30 * 30 pesos + 30 bias) +
(30 * 10 pesos + 10 bias) +
= 24780 variáveis



784
input layer

30
hidden layer 1

30
hidden layer 2

10
output layer

CNN

As redes neurais convolucionais (CNNs) têm várias características distintas que as tornam especialmente adequadas para o processamento de dados de entrada com estrutura espacial, como imagens. Aqui estão algumas características principais das CNNs:

1. ***Convolução:*** As CNNs utilizam a operação de convolução para aprender e extrair características dos dados de entrada. Essa operação envolve a aplicação de filtros convolucionais a pequenas regiões locais dos dados, calculando a multiplicação entre os valores dos filtros e os valores de entrada correspondentes. A convolução é capaz de capturar informações locais e padrões espaciais nas imagens.

2. ***Camadas convolucionais:*** As camadas convolucionais são responsáveis por aplicar filtros convolucionais aos dados de entrada. Essas camadas aprendem automaticamente os pesos dos filtros durante o processo de treinamento, permitindo a extração de características relevantes, como bordas, texturas ou formas, em diferentes níveis de abstração.

3. ***Pooling:*** As camadas de pooling (como o MaxPooling) são usadas para reduzir a dimensionalidade espacial dos mapas de características. Elas dividem a entrada em regiões e mantêm apenas um valor representativo (como o máximo) para cada região. O pooling ajuda a reduzir a quantidade de parâmetros e torna a rede mais tolerante a pequenas variações nas características encontradas nas imagens.

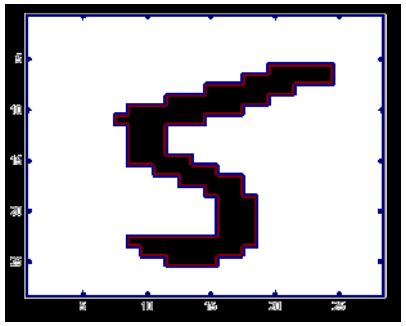
4. ***Camadas totalmente conectadas:*** Após as camadas convolucionais e de pooling, as CNNs geralmente têm camadas totalmente conectadas (Dense). Essas camadas são responsáveis por realizar a classificação ou regressão final, mapeando as características extraídas para as classes ou valores de saída desejados.

5. ***Funções de ativação:*** As CNNs utilizam funções de ativação não lineares, como a função ReLU (Rectified Linear Unit), após cada camada convolucional e camada totalmente conectada. Essas funções introduzem a não linearidade nas redes e ajudam na aprendizagem de relações complexas entre as características.

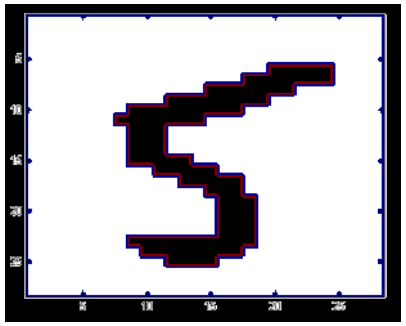
6. ***Aprendizado de pesos:*** Durante o treinamento, as CNNs aprendem automaticamente os pesos dos filtros convolucionais e das camadas totalmente conectadas por meio de algoritmos de otimização, como o Gradiente Descendente. Os pesos são ajustados para minimizar a diferença entre as previsões da rede e os rótulos corretos durante o treinamento.

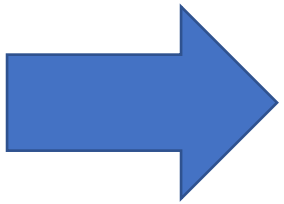
7. ***Conexões locais e compartilhamento de pesos:*** Nas CNNs, as conexões são locais, o que significa que cada neurônio está conectado apenas a uma região local da entrada. Além disso, os pesos dos filtros convolucionais são compartilhados em todas as regiões do mapa de características, o que ajuda a reduzir o número de parâmetros e torna as CNNs mais eficientes em termos de uso de memória e capacidade de generalização.

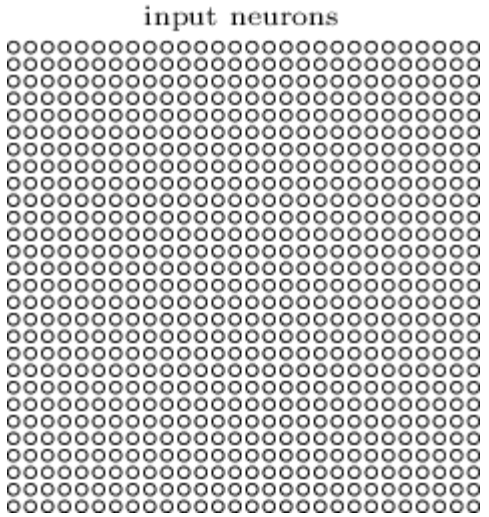
CNN



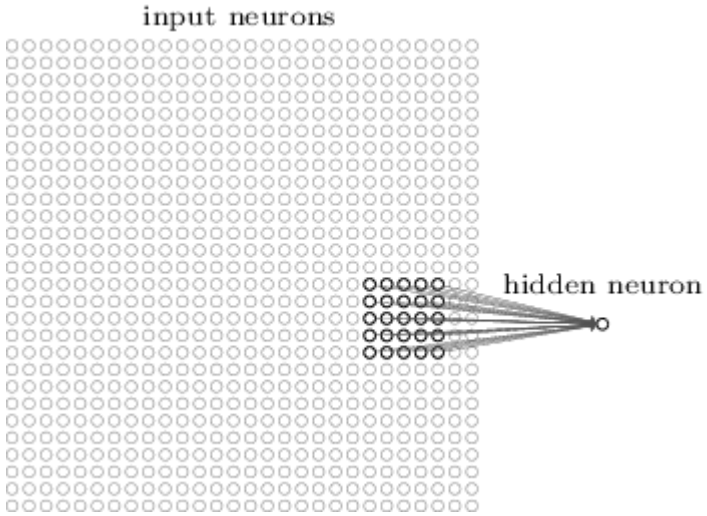
CNN



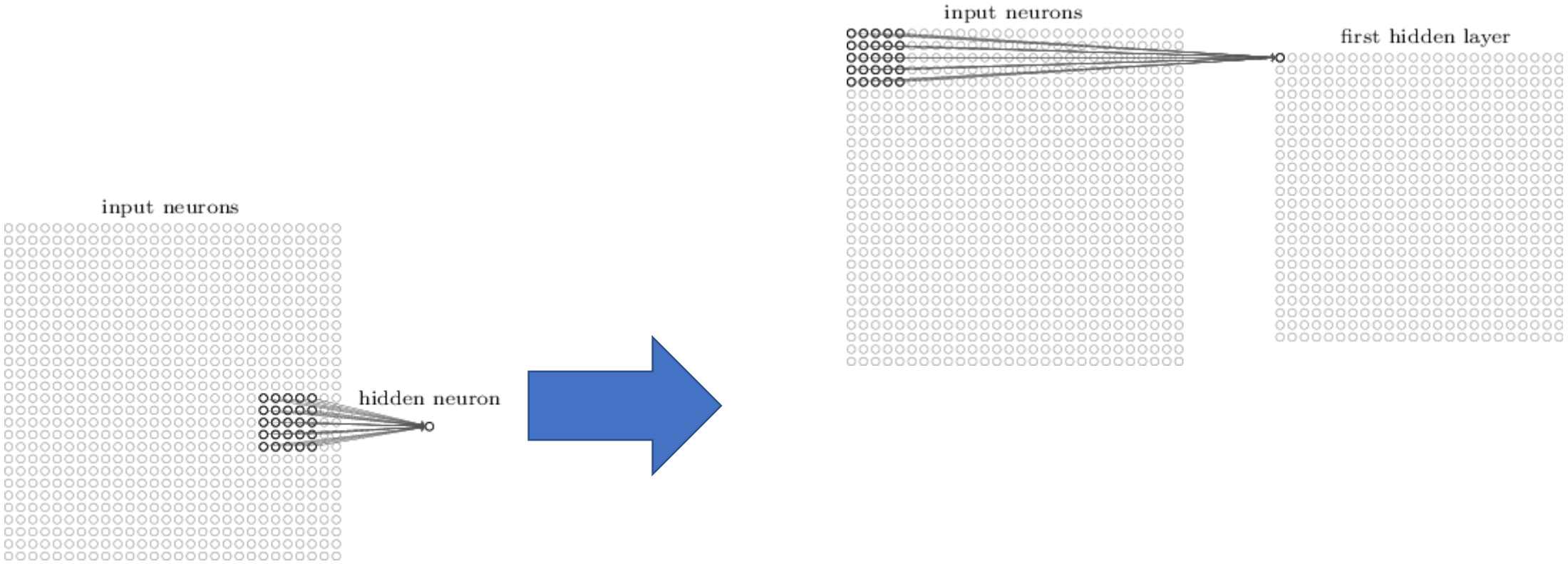




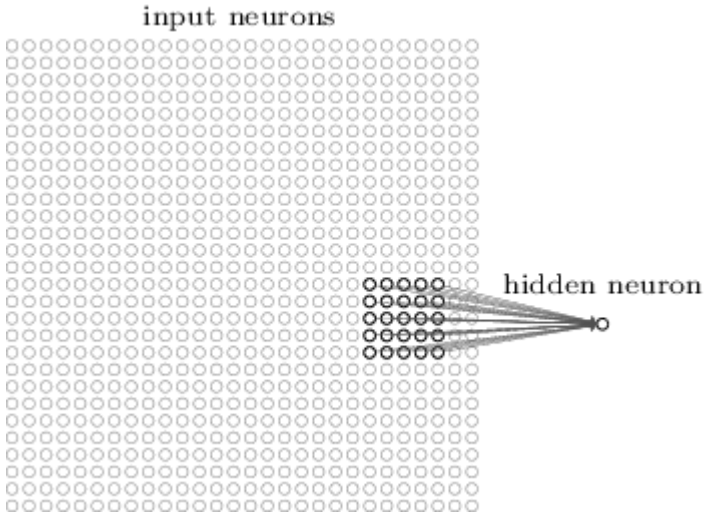
CNN

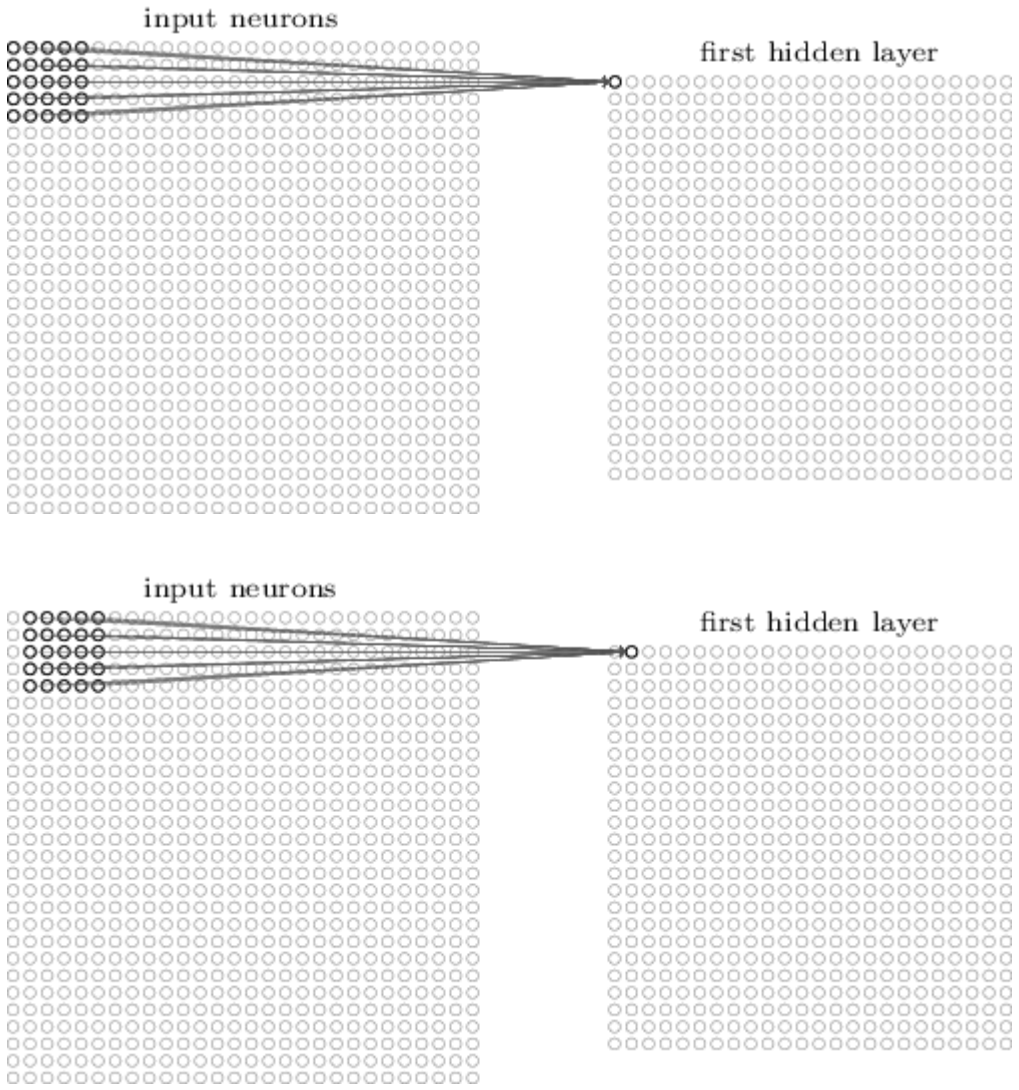


CNN

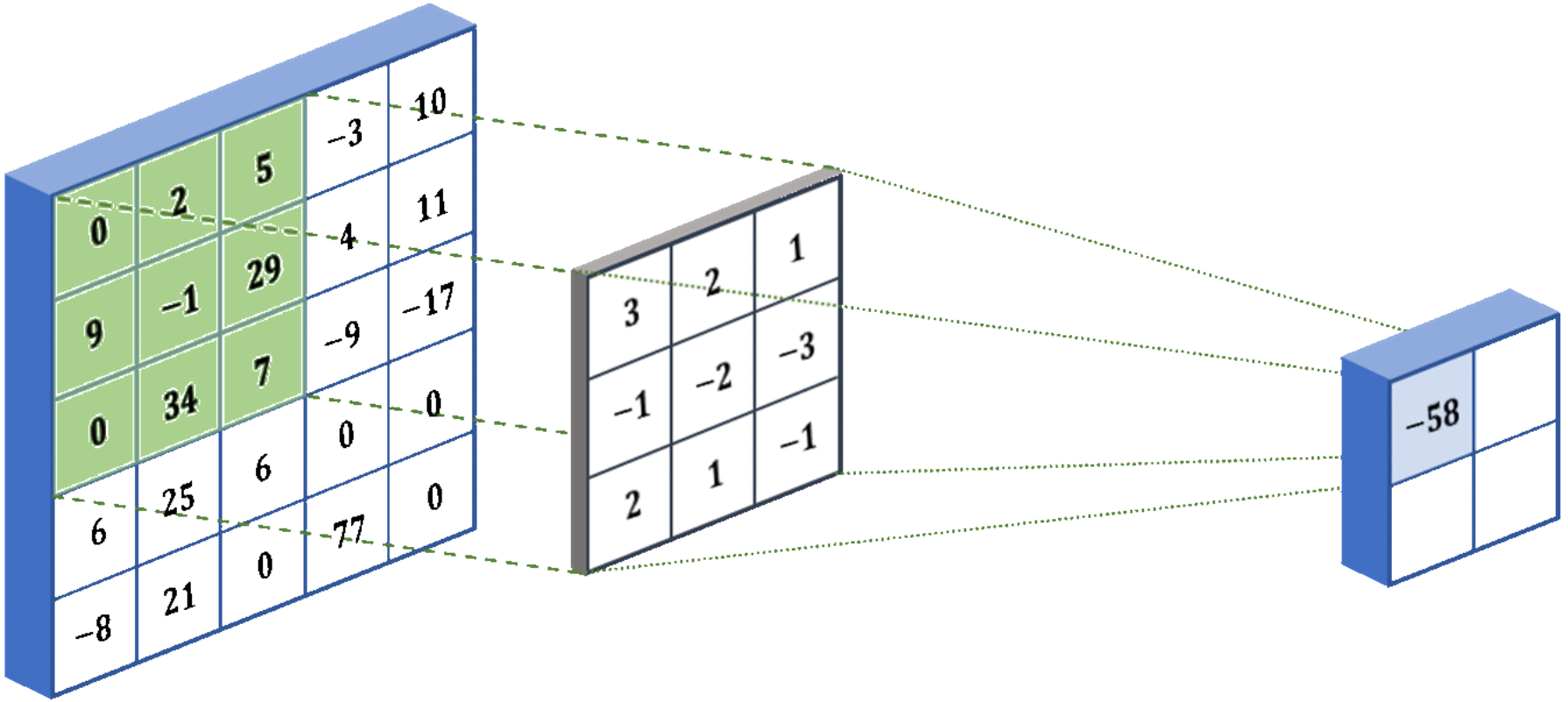

input neurons
hidden neuron
input neurons
first hidden layer

CNN


input neurons
hidden neuron


input neurons
first hidden layer
input neurons
first hidden layer

CNN



CNN

CAMADAS:

1. ***Dense:*** A camada Dense (totalmente conectada) é uma camada clássica de redes neurais. Ela conecta cada neurônio da camada anterior a cada neurônio da camada atual. Essa camada é usada para mapear as características extraídas das camadas anteriores para as classes ou valores de saída finais. É comumente usada nas últimas camadas de uma rede neural para classificação ou regressão.

2. ***Dropout:*** A camada Dropout é usada para reduzir o overfitting (sobreajuste) em uma rede neural. Durante o treinamento, ela aleatoriamente "desativa" um determinado percentual de neurônios, forçando a rede a aprender características mais robustas e reduzindo a dependência de neurônios específicos. Isso ajuda a melhorar a capacidade de generalização da rede.

3. ***Flatten:*** A camada Flatten é usada para converter um tensor multidimensional em um vetor unidimensional. Ela é comumente usada para conectar camadas convolucionais a camadas totalmente conectadas. Por exemplo, após a aplicação de camadas convolucionais em uma imagem, a camada Flatten é usada para transformar os recursos espaciais em um vetor antes de alimentá-los para camadas Dense.

4. ***Conv2D:*** A camada Conv2D é uma camada de convolução 2D usada em redes neurais convolucionais para extrair características de uma imagem. Ela aplica um conjunto de filtros convolucionais à entrada e realiza operações de convolução para calcular mapas de características. Cada filtro aprende a detectar características específicas, como bordas, texturas ou padrões relevantes para a tarefa em questão.

5. ***MaxPooling2D:*** A camada MaxPooling2D é usada para reduzir a dimensionalidade espacial dos mapas de características gerados pelas camadas convolucionais. Ela divide a entrada em regiões retangulares e retorna o valor máximo de cada região, reduzindo assim a resolução espacial. Isso ajuda a reduzir a quantidade de parâmetros e a tornar a rede mais tolerante a pequenas variações nas características encontradas nas imagens de entrada.

Essas camadas são blocos fundamentais para a construção de redes neurais convolucionais no Keras, permitindo que as redes aprendam características importantes dos dados de entrada e realizem tarefas de classificação, regressão ou outras tarefas relacionadas a imagens.

Mestrado em Engenharia e Gestão Industrial

Inteligência Artificial aplicada na Indústria

Daniel Nogueira

dnogueira@ipca.pt

https://www.linkedin.com/in/danielfnogueira/