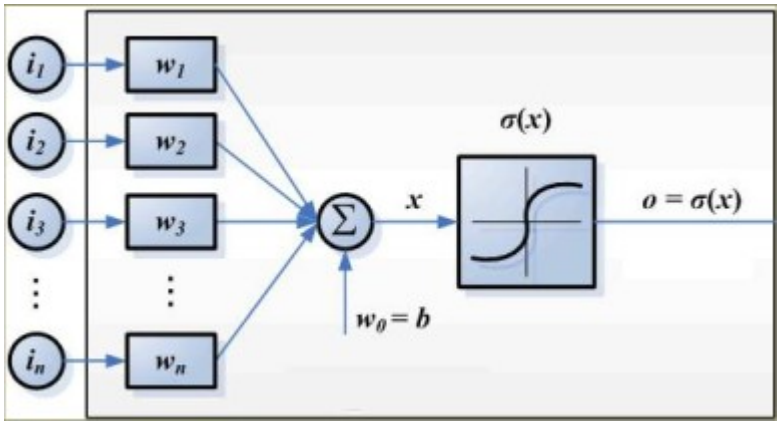
CNN

Convolutional Neural Networks
Extração de características

Prof. Frederico Coelho.

Introdução

Neurônio Artificial
- Vetor de Caracteristícas
- Pesos
- Bias
- Função de Ativação



Introdução

- Modelos abordados serão caracterizados pela equação

$$\mathbf{Xw = Y} \qquad \text{Ou de maneira mais geral} \qquad \mathbf{f(Xw) = Y}$$

Onde f(.) é a função de ativação do modelo:

- Função identidade f(u)=u;
- Função degrau;
- Função logística.

Modelos de única camada

Introdução

São modelos de única camada uma vez que as amostras de entrada, linhas de **X**, são multiplicadas pelo vetor **w** resultando em **Y**.

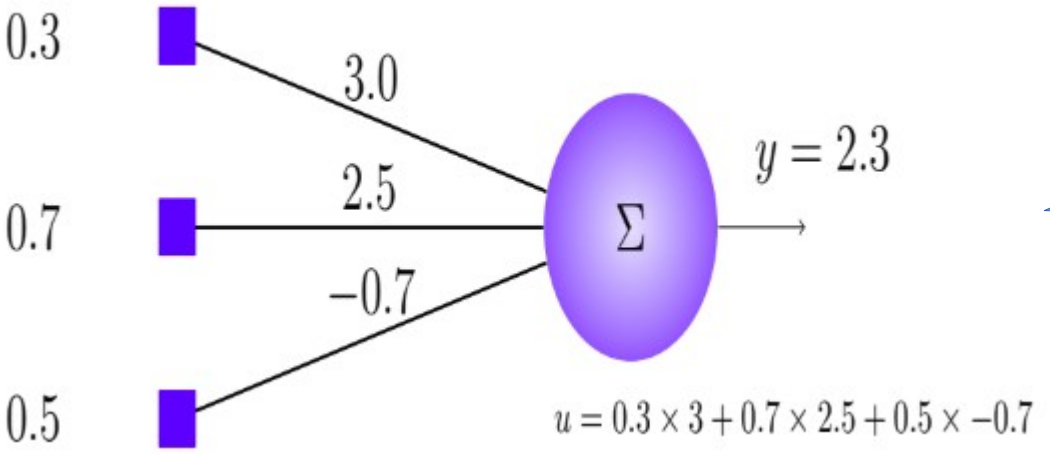
Para o caso de f(u) = u, o modelo de camada única será linear

$$\mathbf{Xw = Y}$$

Introdução

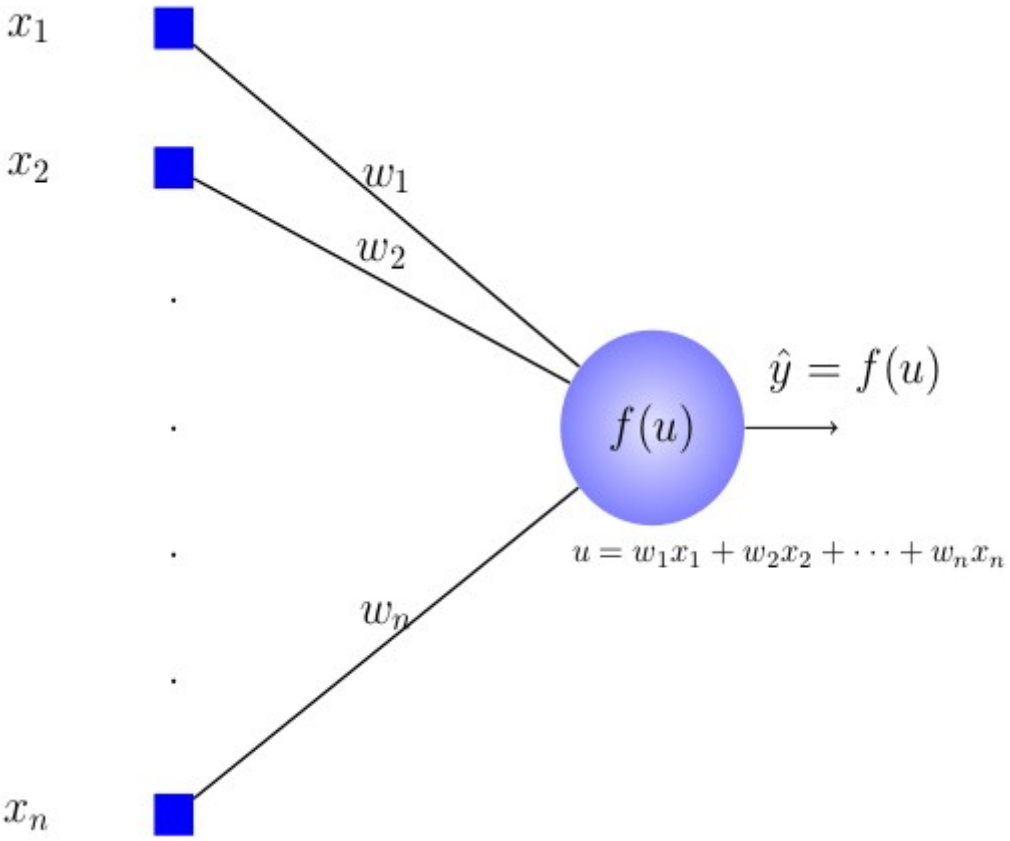
$$\underbrace{\begin{pmatrix} 0.3 & 0.7 & 0.5 \\ -1.2 & 0.5 & 3 \\ 0.4 & -1.7 & -2.1 \end{pmatrix}}_{\mathbf{X}} \underbrace{\begin{pmatrix} 3.0 \\ 2.5 \\ -0.7 \end{pmatrix}}_{\mathbf{w}} = \underbrace{\begin{pmatrix} 2.3 \\ -4.45 \\ -1.58 \end{pmatrix}}_{\mathbf{y}}$$

Representação Matricial



Representação Modelo linear de camada únical

Introdução



Modelo de McCulloch e Pitts:

- Aplicação de função de limiar à soma ponderada das entradas
- No modelo MCP a função de ativação é a degrau
- No Adaline a função de ativação é a função identidade:

$$f(u) = u$$

Então a saída corresponde exatamente à soma ponderada das entradas:

$$\hat{y} = \sum w_i x_i$$

Introdução

Treinar o neurônio minimizando uma função de custo:

$$J = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

É uma função quadrática – Facilita busca por mínimo global

Como $\hat{y}_i = \sum_{j=1}^{n} w_j x_{ij}$

A minimização de J é um problema quadrático nos pesos w_j já que a função de custo pode ser escrita como

$$J = \sum_{i=1}^{N} (y_i - \sum_{j=1}^{n} w_j x_{ij})^2$$

Introdução

Se considerarmos um problema de uma única variável o modelo Adaline será caracterizado por:

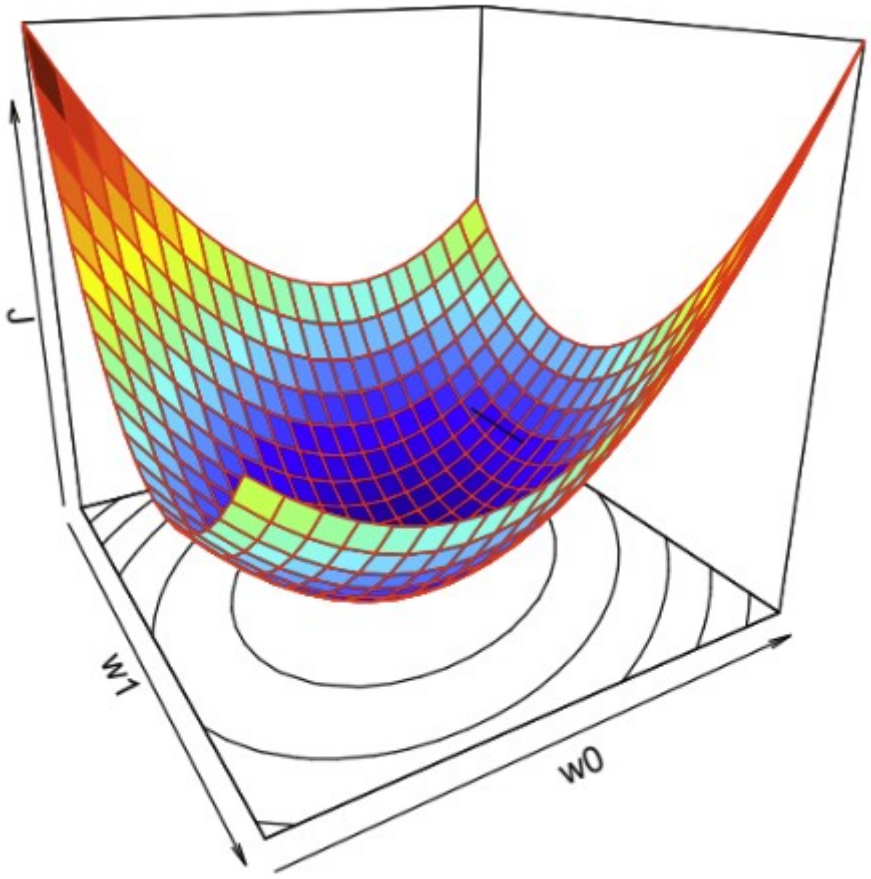
$$\hat{y}_i = w_1 x_i + w_0$$

E a função de custo será:

$$J = \sum_{i=1}^{N} (y_i - (w_1 x_i + w_0))^2$$

Que é quadrática em w_1 e w_0 que são os parâmetros que definem a função.

Introdução

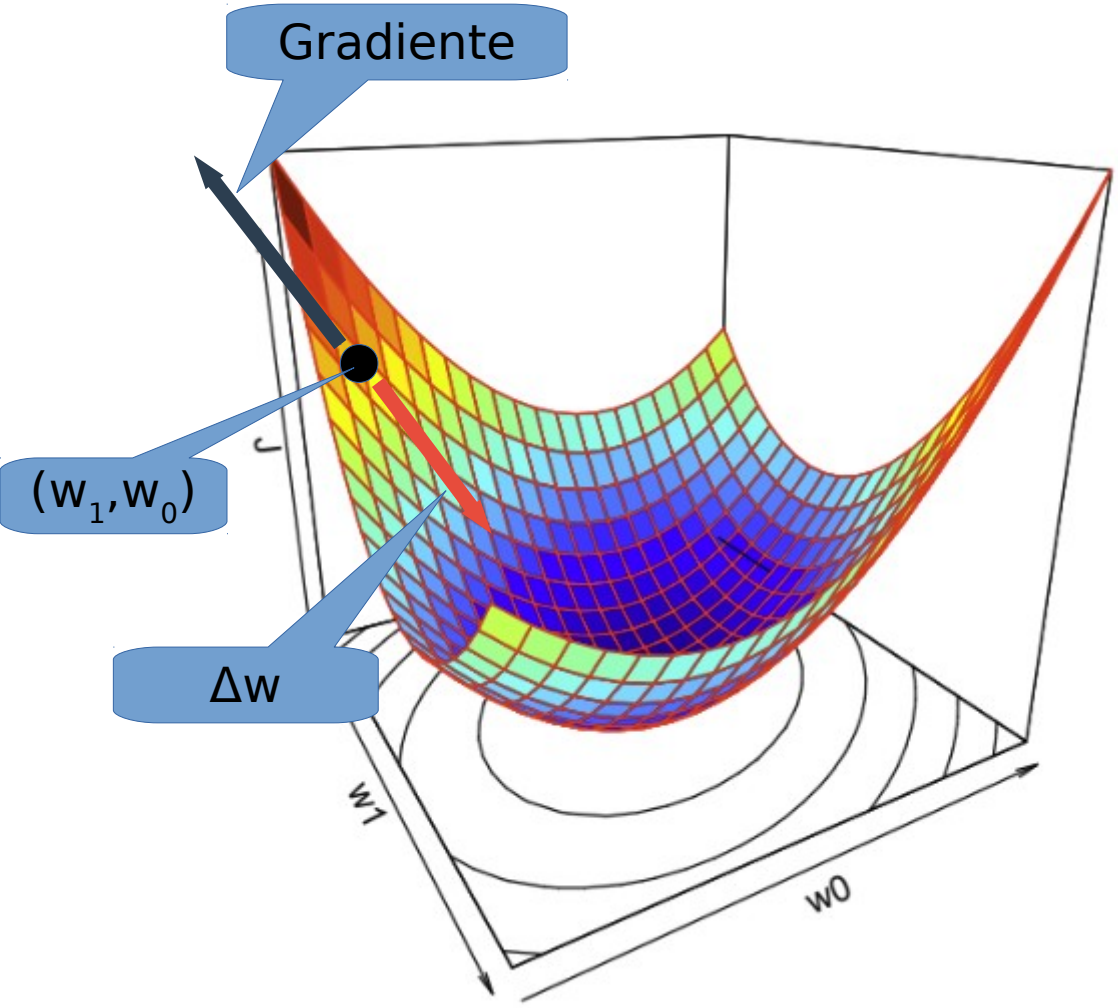


Superfície de erro quadrática correspondente à função de custo considerando:

$$\mathbf{x} = [0.1, 2, -1, 0.2]^T \text{ e } \mathbf{y} = [2.2, 6, 0, 2.4]^T$$

O mínimo da superfície ocorre para W = [2,2]

Introdução

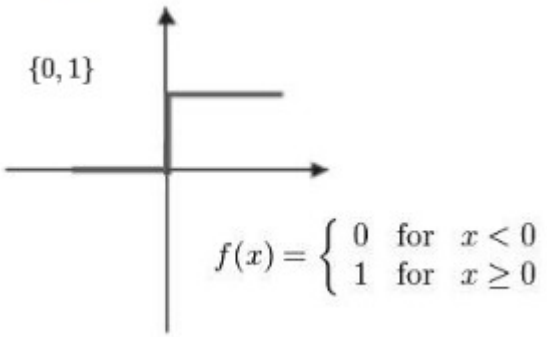


Deseja-se obter a direção de ajuste do vetor de pesos pelo gradiente descendente, ou seja, o ajuste deve ocorrer em direção contrária ao gradiente em cada ponto da supefície J.

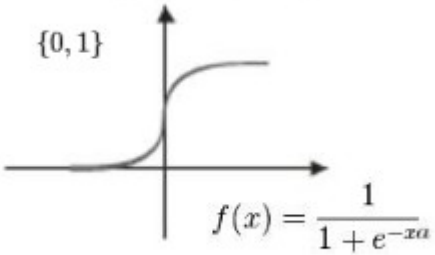
Introdução

Funções de ativação:

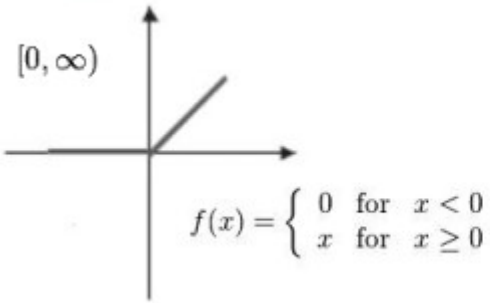
- Hard Limit



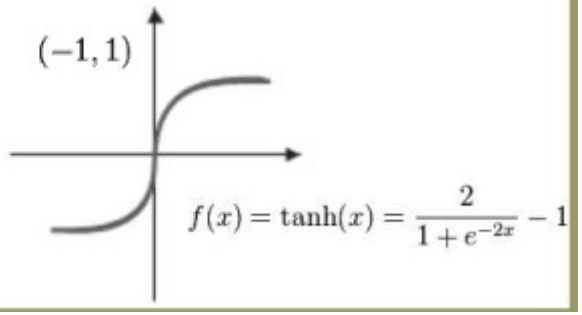
- Sigmoid (Log)



- RELU



- Tangente Hiperbólica

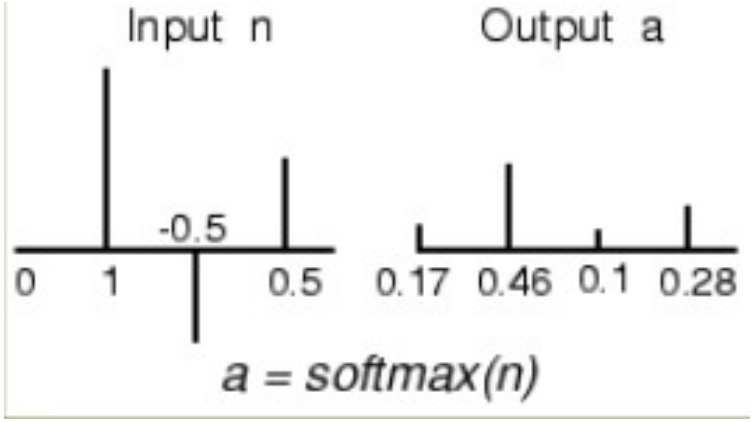


Introdução

Função de ativação Softmax:

- Normaliza entre 0 e 1 as saídas, em um problema multi-classes, obtidas por um classificador linear (Ex: Neurônios).
- Aplicada na camada de saída da rede (output layer).
- Objetivo: Definir a probabilidade de uma classe dentro de um problema multi-classes

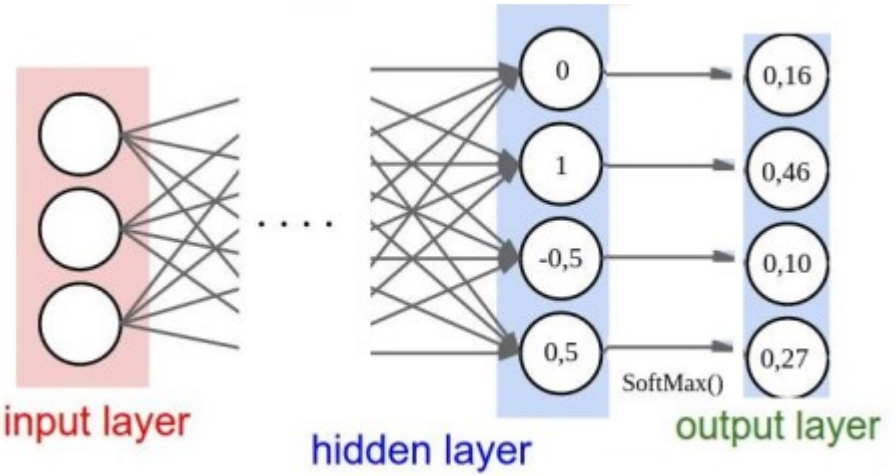
$$y_i = \frac{e^{z_i}}{\sum_{j \in group} e^{z_j}}$$



Introdução

Função de ativação Softmax (exemplo):

$$y_i = \frac{e^{z_i}}{\sum_{j \in group} e^{z_j}}$$



$$e^{0} = 1.0000$$
$$e^{1} = 2.7182$$
$$e^{-0.5} = 0.6065$$
$$e^{0.5} = 1.6487$$

$$\sum_{j \in group} e^{z_j} = \sim 5.9734$$

$$y_0 = \frac{1.0000}{\sim 5.9734} = 0.16$$

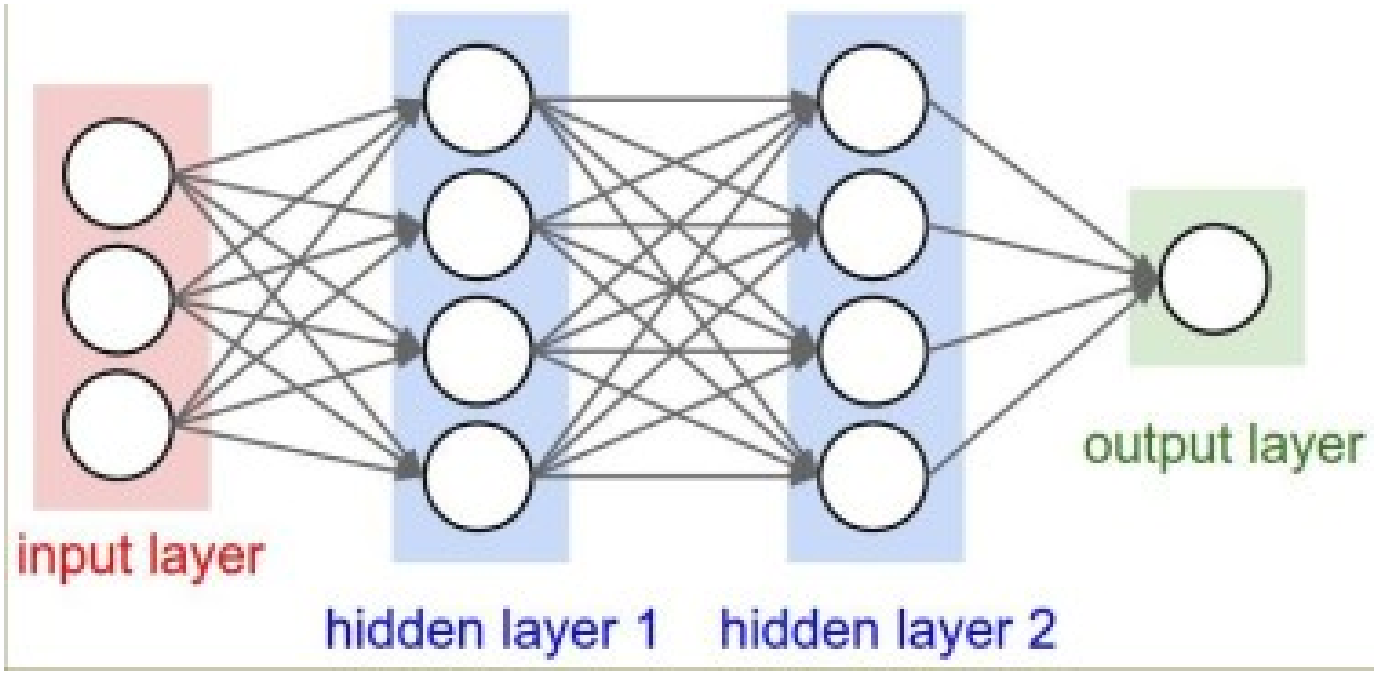
$$y_1 = \frac{2.7182}{\sim 5.9734} = 0.46$$

$$y_2 = \frac{0.6065}{\sim 5.9734} = 0.10$$

$$y_3 = \frac{1.6487}{\sim 5.9734} = 0.27$$

Introdução

Rede Neural Artificial (MLP)

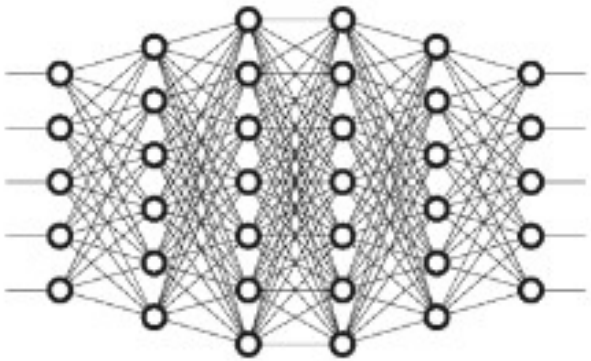


Conecta várias camadas.

Introdução

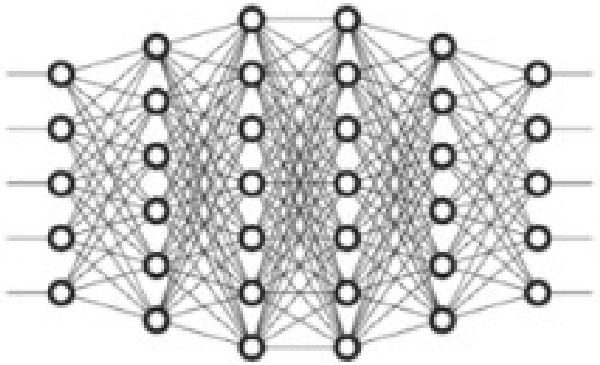
Porque não usar Rede Neural Artificial (MLP) neste tipo de problema?

Image with 28 x 28 x 3 pixels



Number of weights in the first hidden layer will be 2352

Image with 200 x 200 x 3 pixels



Number of weights in the first hidden layer will be 120,000

Impraticáveis e tendem ao overfitting.

Deep Learning

- Aprendizado baseado em Multi-Camadas (RN Densas)
- Representação dos dados em diferentes níveis de abstração
- Extração de Características é Implícita
- Elevada quantidade de parâmetros (Ajuste por BackProgation)
- Necessita de um grande número de exemplos para o aprendizado eficiente
- O aprendizado é voltado para compreensão pela máquina, ou seja, os dados são de díficil compreensão/visualização humana
- Vasta aplicação na área de Visão Computacional (processamento de áudio, imagens, vídeo, etc).

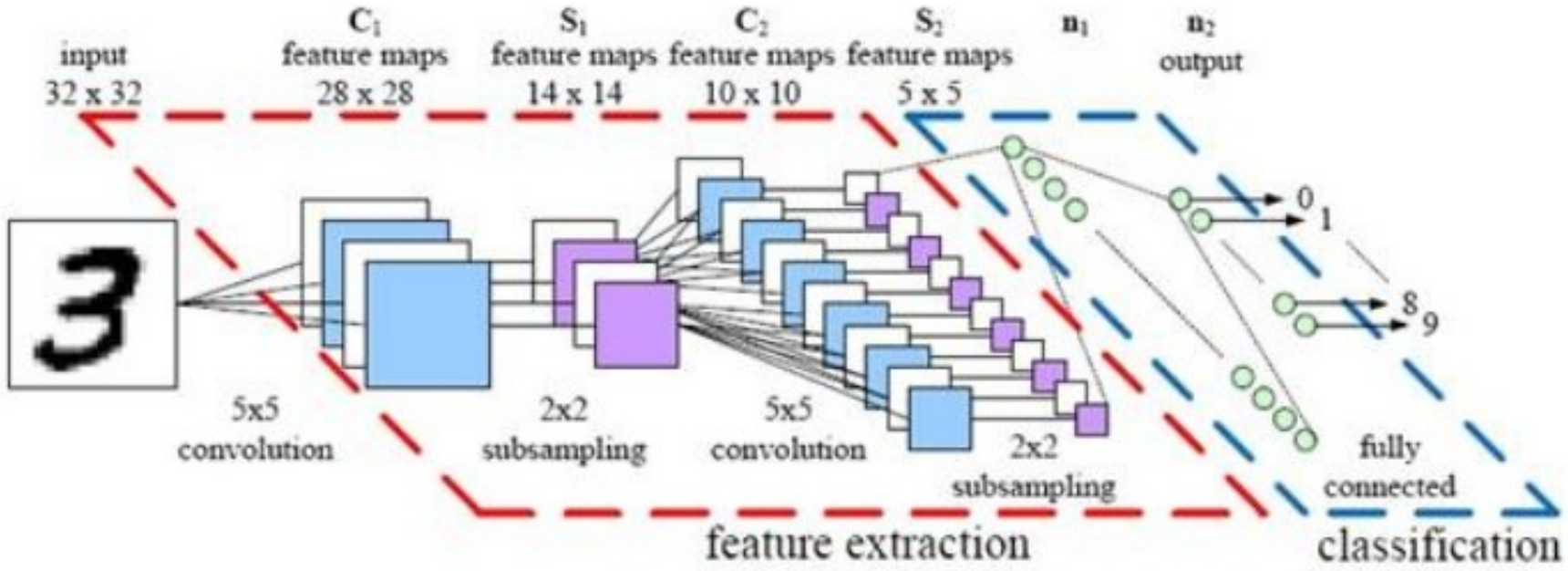
CNN

- Inspiradas no modelo biológico da visão
- Conceito de Deep Learning (Multi-Camadas)
- Idealizada no ínicio do anos 90 [Lecun], e vasta aplicação após 2006 devido a “popularização” de GPU's (Custo ~$ 3000,00)
- Treinamento requer alto custo computacional e numerosa base de dados

CNN

- Compostas de duas grandes etapas:

→Extração de Características pelas Camadas Convolucionais
→Classificação



CNN

Tipos de camadas

● Convolucional : Definem os filtros (Aprendizado / BackPropagation)

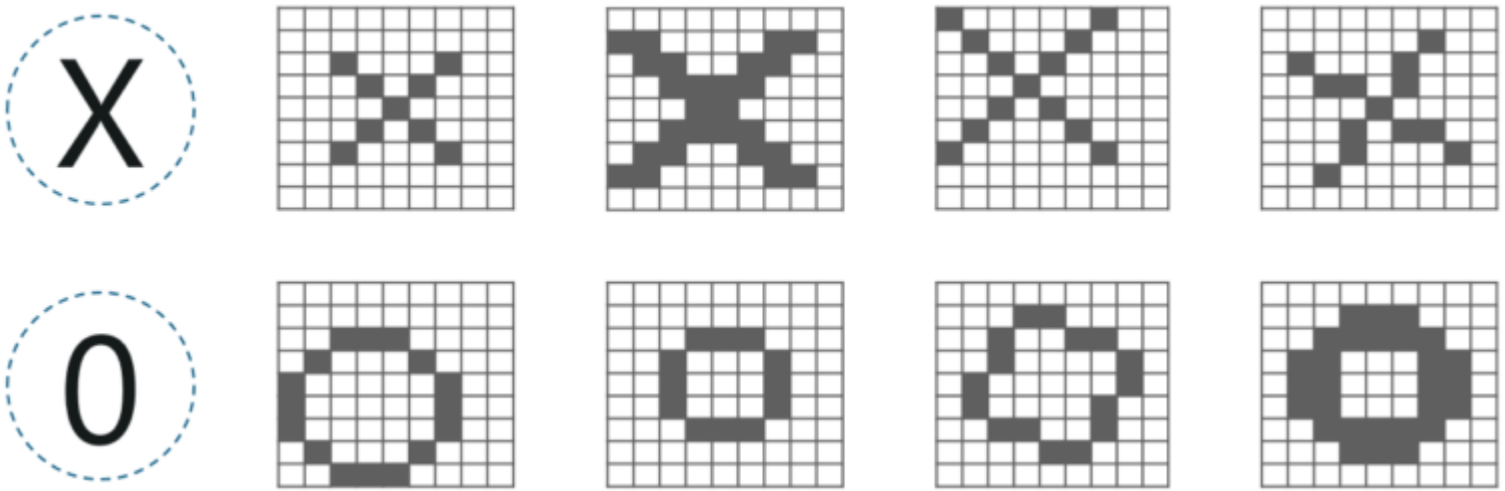
● Ativação: Neurônios (Relu / Sigmoid / TangH)

● Pooling : Reduzem as escalas (Max, Median, etc..)

● Fully-Connected (FC): Camada que determina as classes (Classificador)

CNN

Como funcionam?
Considere as seguintes imagens



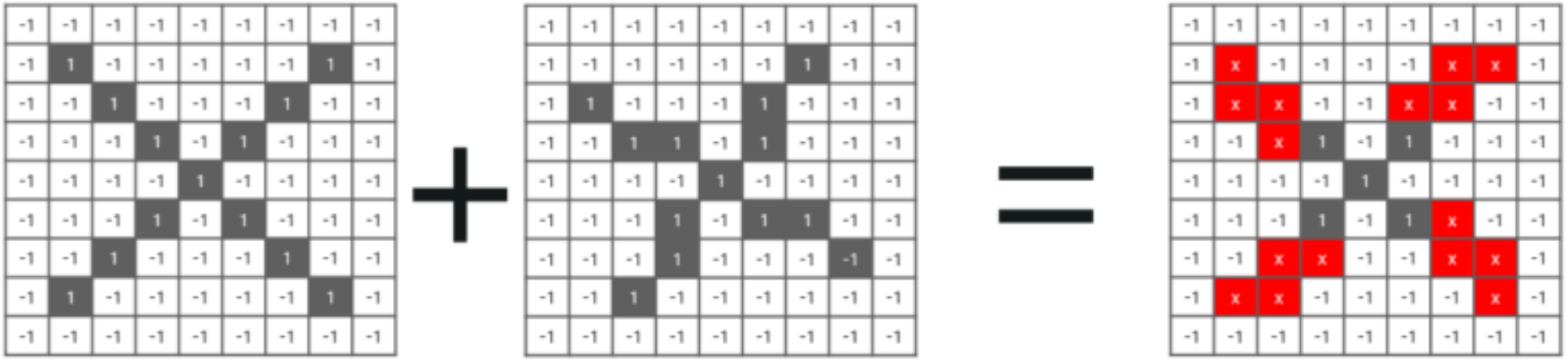
CNN

O computador enxerga cada imagem como pixels (números)

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

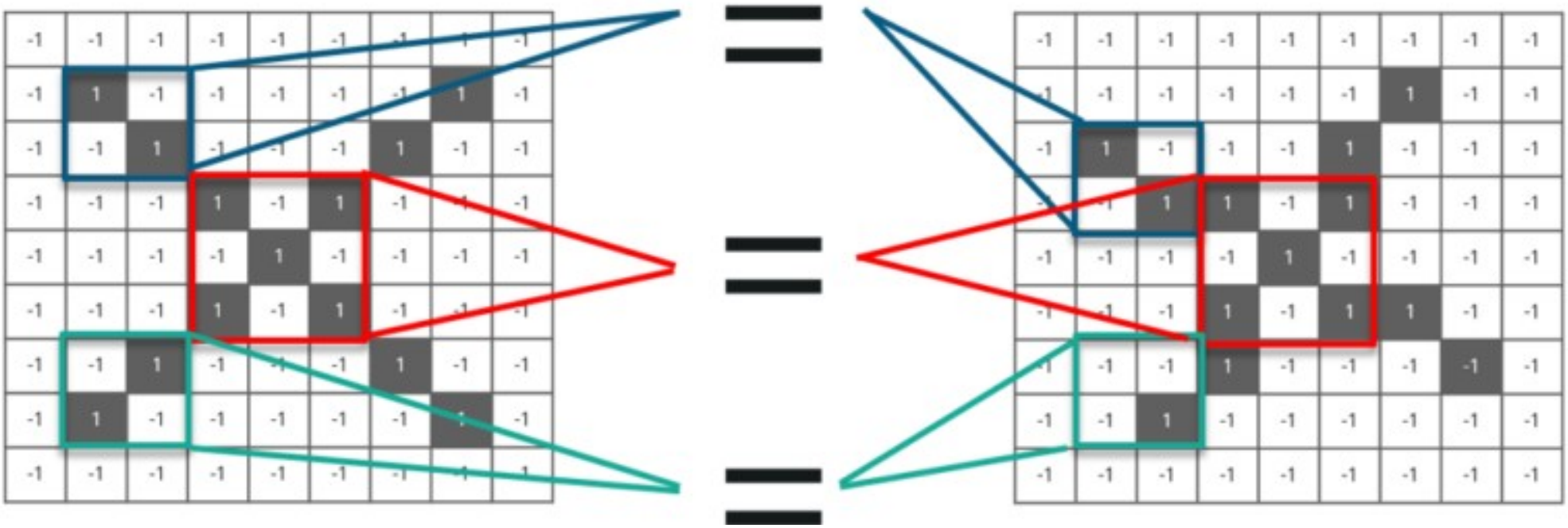
CNN

Se queremos classificar a seguinte imagem, podemos compará-la com o padrão da classe X



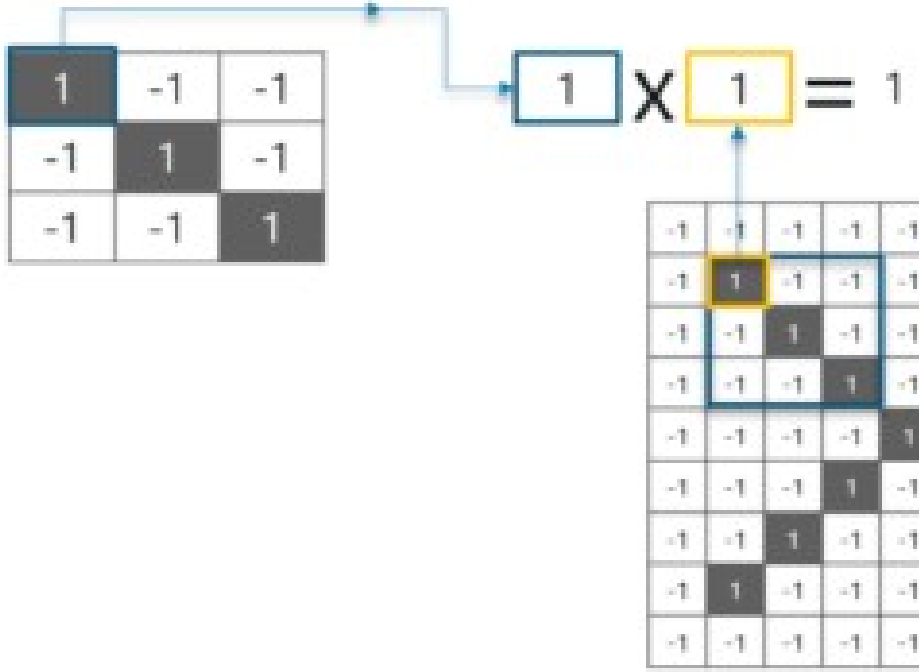
CNN

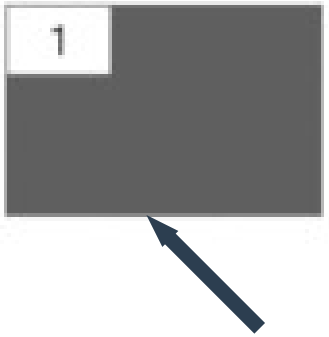
Mas erraríamos em vários pontos e não saberíamos como definir se ela pertence a eesta classe



Solução: comparar pedaços da imagem, ou seja, fazer convoluções

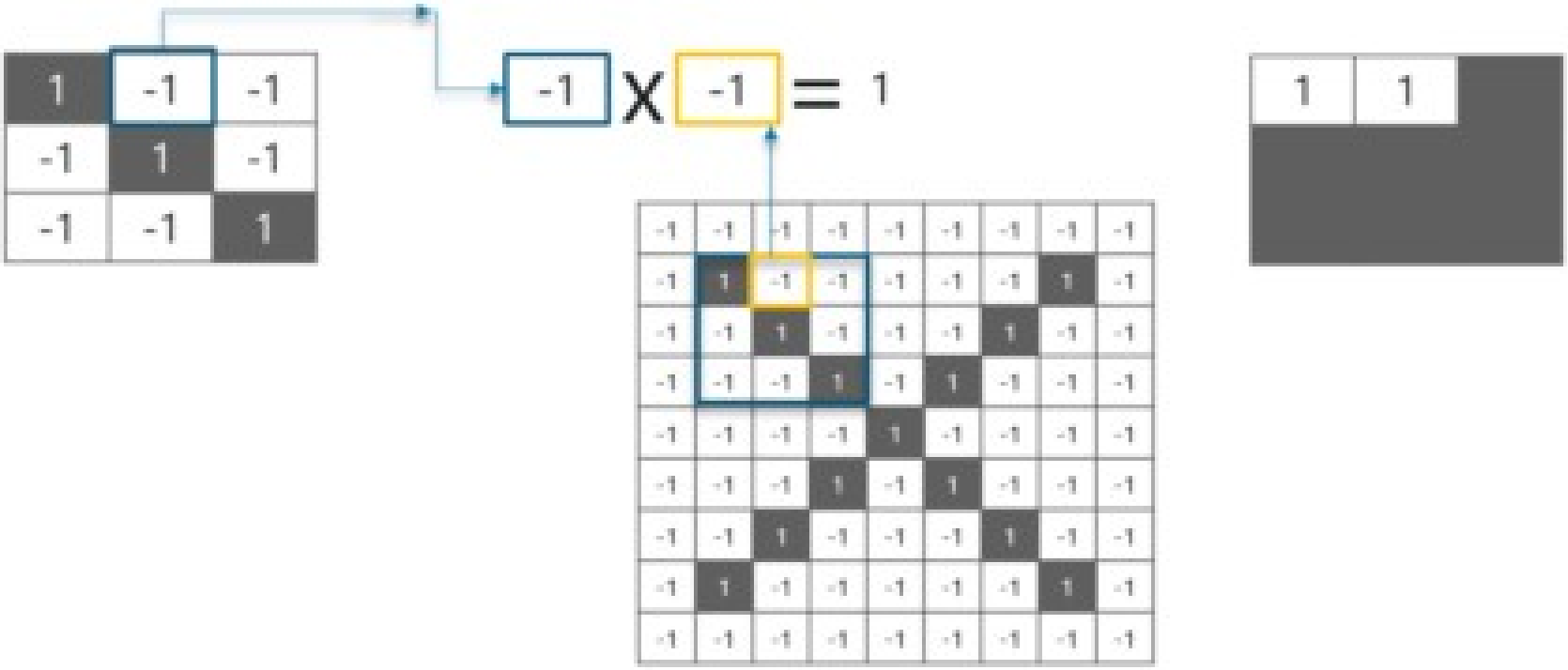
CNN - convolução





Produz outra feature

CNN - convolução



CNN - convolução

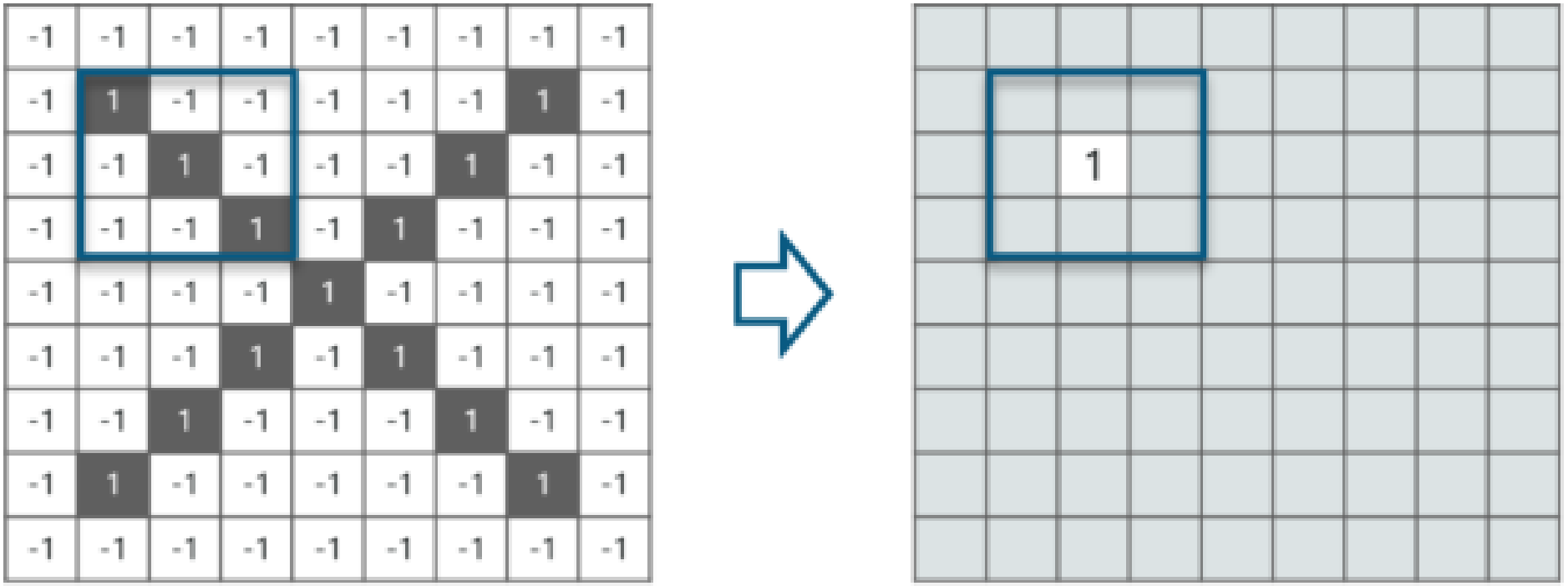
1	-1	-1
-1	1	-1
-1	-1	1

$$\frac{1+1+1+1+1+1+1+1+1}{9} = 1$$

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

1	1	1
1	1	1
1	1	1

CNN - convolução



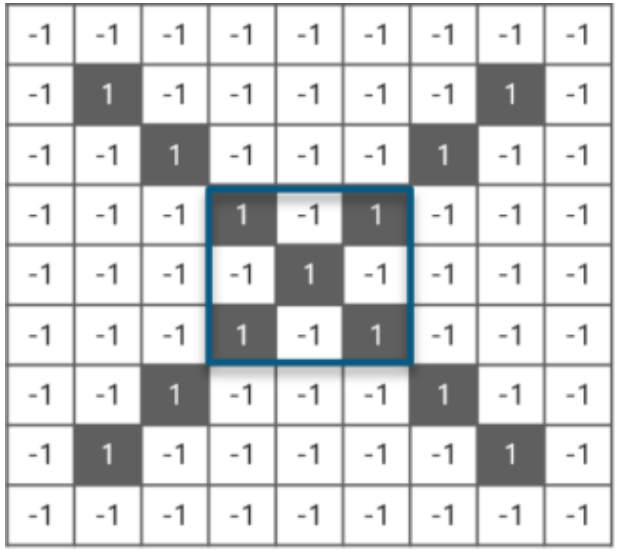
CNN - convolução

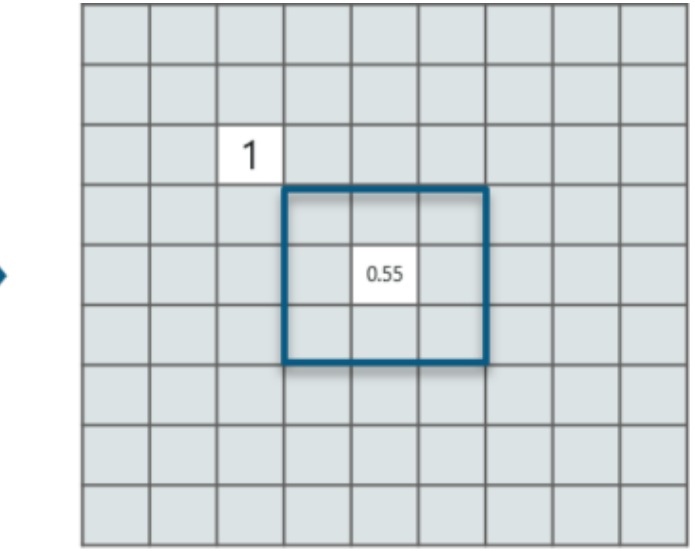
1	-1	-1
-1	1	-1
-1	-1	1

$$\frac{1+1-1+1+1+1-1+1+1}{9} = .55$$

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

1	1	-1
1	1	1
-1	1	1





CNN - convolução

Ao final da convolução teremos uma matriz/imagem como esta:

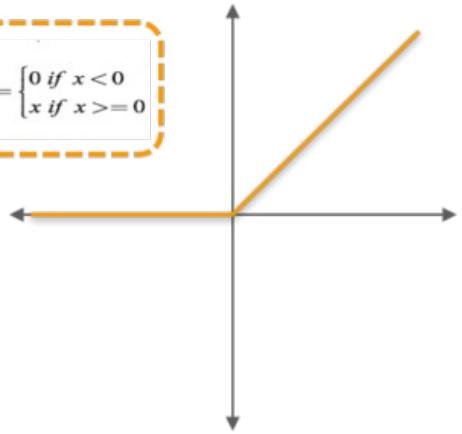
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.0	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.0	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

A intensidade do sinal de saída não depende de onde os recursos estão localizados, mas simplesmente se os recursos estão presentes. Portanto, um alfabeto poderia estar em posições diferentes e o algoritmo da Rede Neural Convolucional ainda seria capaz de reconhecê-lo.

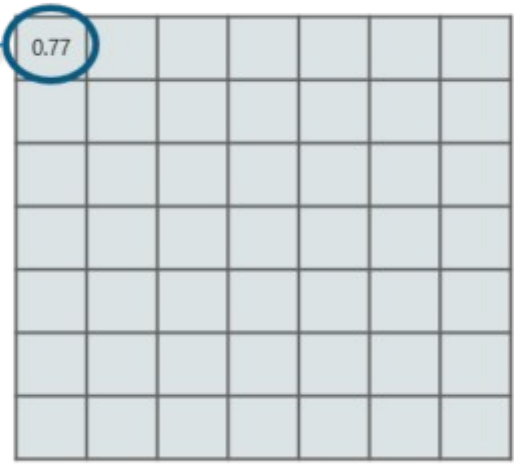
CNN - ReLU

Principal objetivo: retirar os números negativos

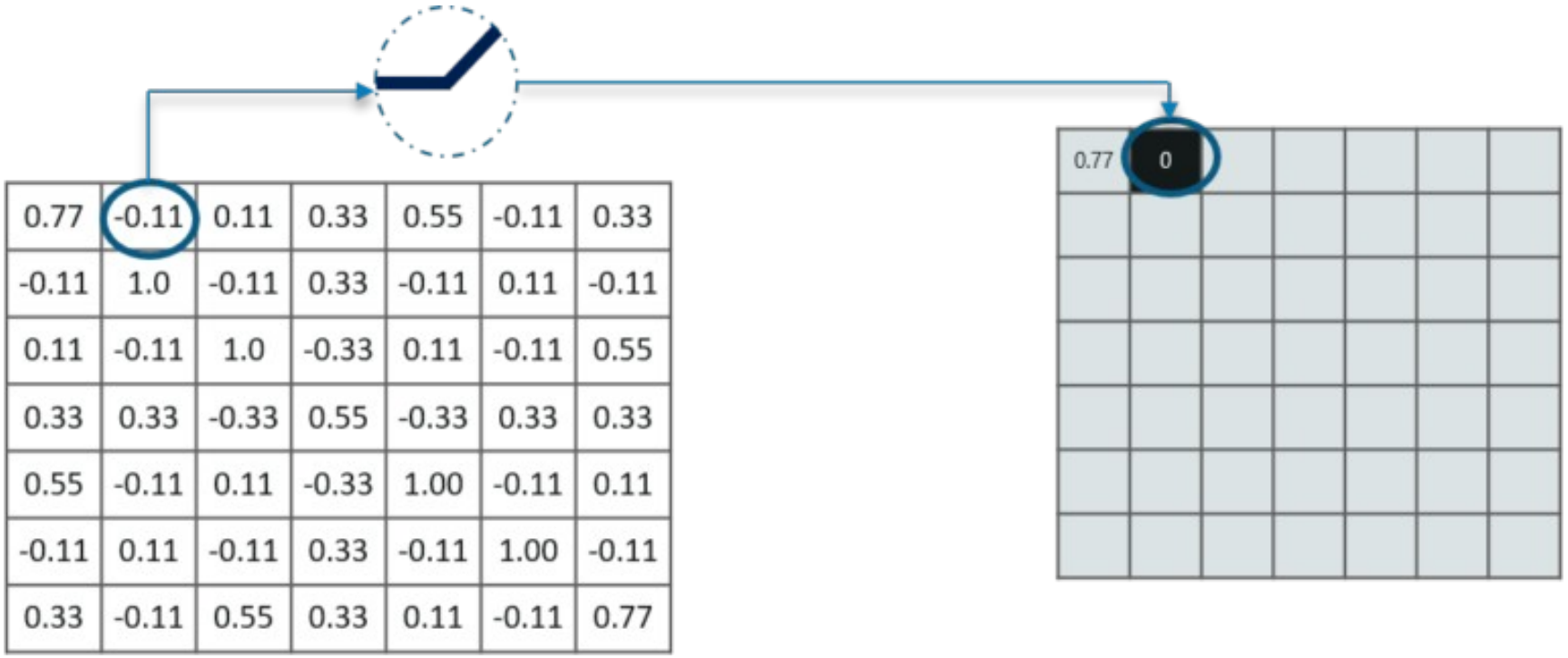
$$f(x) = \begin{cases} 0 \text{ if } x < 0 \\ x \text{ if } x >= 0 \end{cases}$$



0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.0	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.0	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77



CNN - ReLU



0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.0	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.0	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

CNN - ReLU

Ao final teremos:

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.0	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.0	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

0.77	0	0.11	0.33	0.55	0	0.33
0	1.00	0	0.33	0	0.11	0
0.11	0	1.00	0	0.11	0	0.55
0.33	0.33	0	0.55	0	0.33	0.33
0.55	0	0.11	0	1.00	0	0.11
0	0.11	0	0.33	0	1.00	0
0.33	0	0.55	0.33	0.11	0	1.77

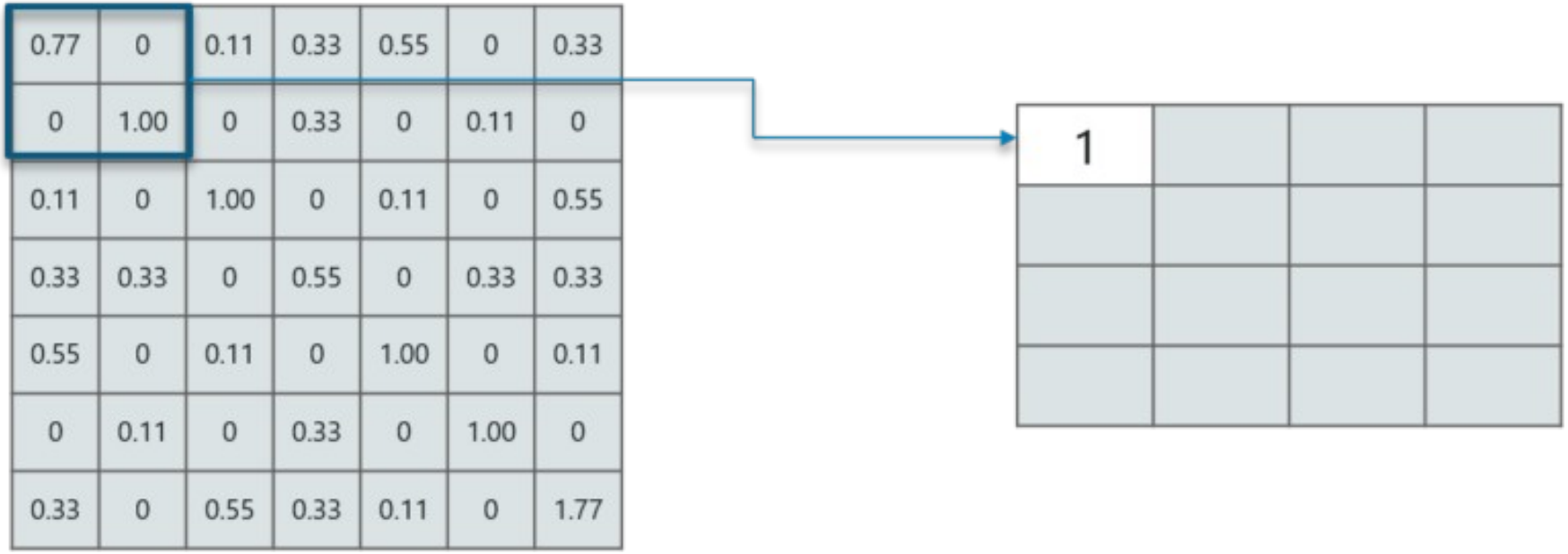
Isso é feito para todos os filtros

CNN - Pooling

As entradas vindas da camada de convolução podem ser "suavizadas" para reduzir a sensibilidade dos filtros a ruídos e variações.

Esse processo de suavização é chamado de subamostragem e pode ser alcançado através da média ou do máximo de uma amostra do sinal.

CNN - Pooling


0.77 0 0.11 0.33 0.55 0 0.33
0 1.00 0 0.33 0 0.11 0
0.11 0 1.00 0 0.11 0 0.55
0.33 0.33 0 0.55 0 0.33 0.33
0.55 0 0.11 0 1.00 0 0.11
0 0.11 0 0.33 0 1.00 0
0.33 0 0.55 0.33 0.11 0 1.77
1

CNN – Max Pooling

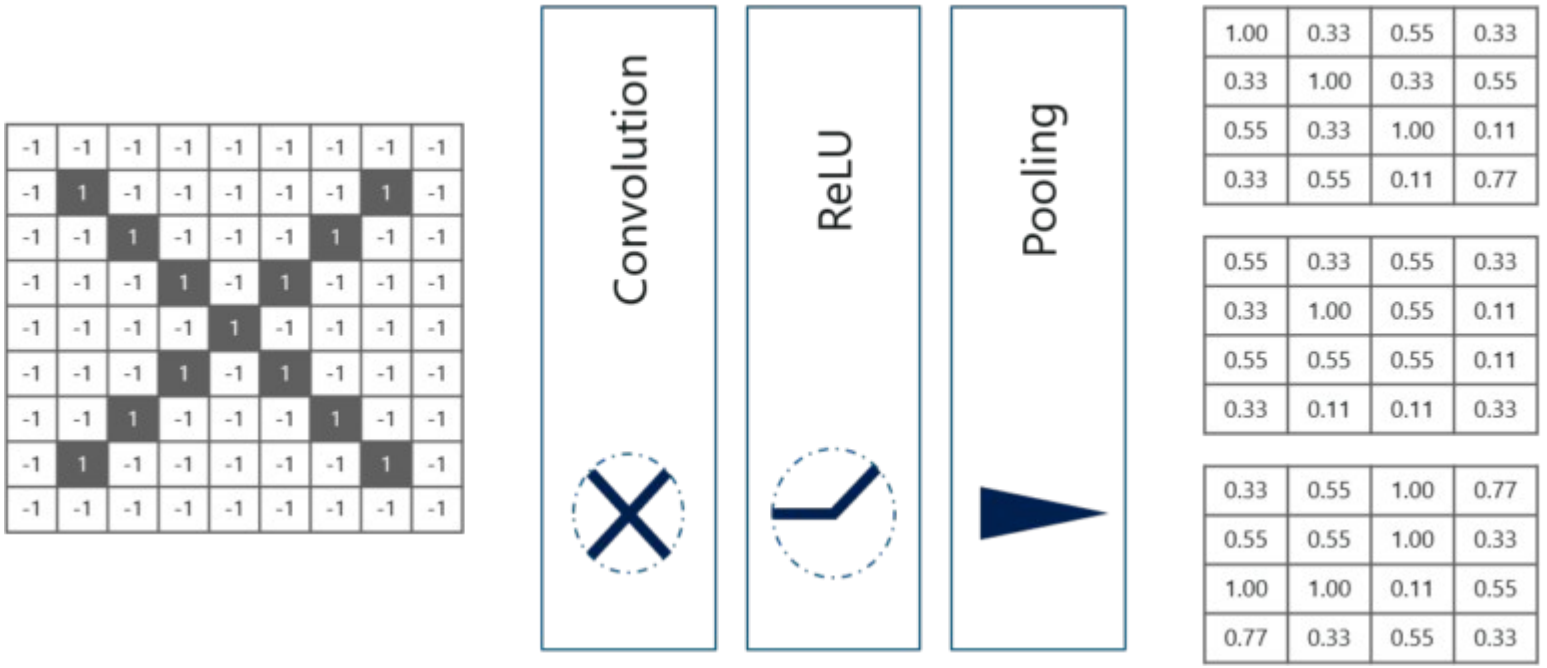
0.77	0	0.11	0.33	0.55	0	0.33
0	1.00	0	0.33	0	0.11	0
0.11	0	1.00	0	0.11	0	0.55
0.33	0.33	0	0.55	0	0.33	0.33
0.55	0	0.11	0	1.00	0	0.11
0	0.11	0	0.33	0	1.00	0
0.33	0	0.55	0.33	0.11	0	1.77

1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

Também é feito para todos os filtros

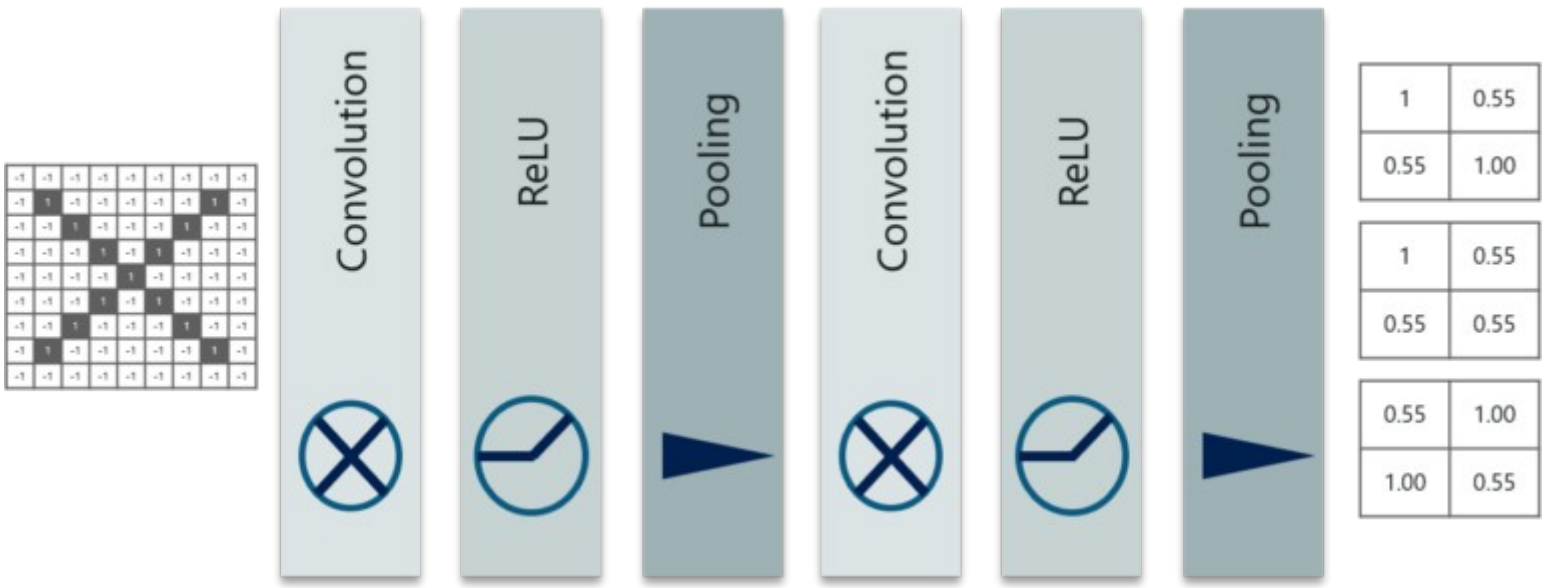
CNN – multiplas camadas

As primeiras três camadas podem então ser representadas assim:



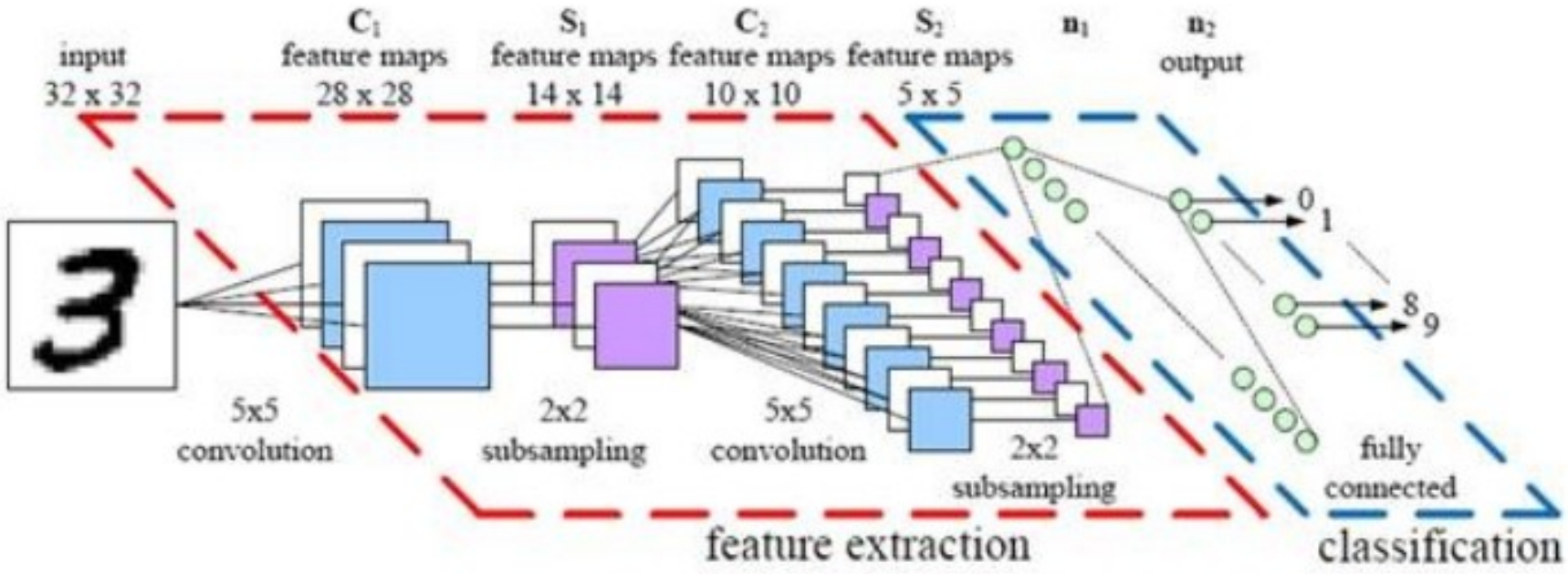
Mas podemos reduzir ainda mais a saída da rede?

CNN – multiplas camadas



A maneira como compomos e conectamos as diferentes camadas produzem diferentes arquiteturas.

CNN – Classificação



CNN – Classificação

Organizamos a saida em um único vetor que contém as features que serão consideradas pelo classificador

1	0.55
0.55	1.00

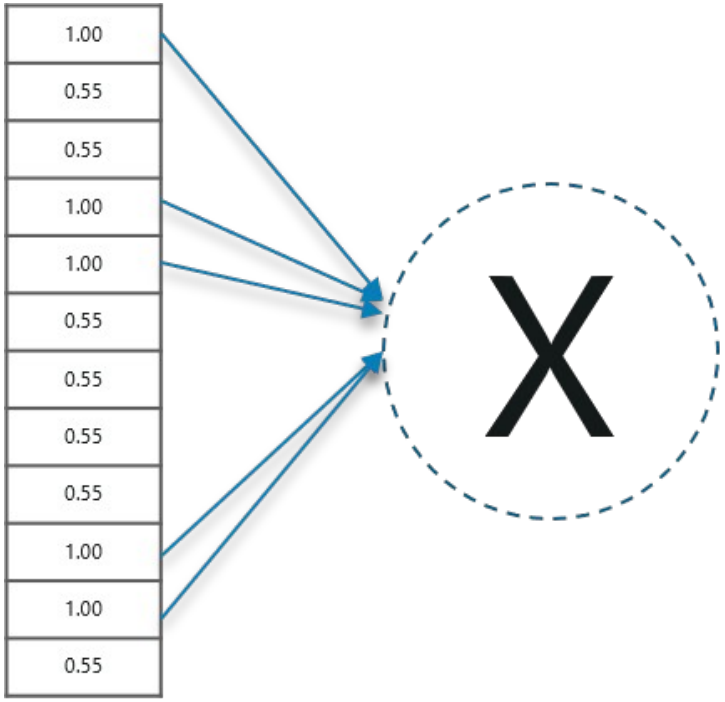
1	0.55
0.55	0.55

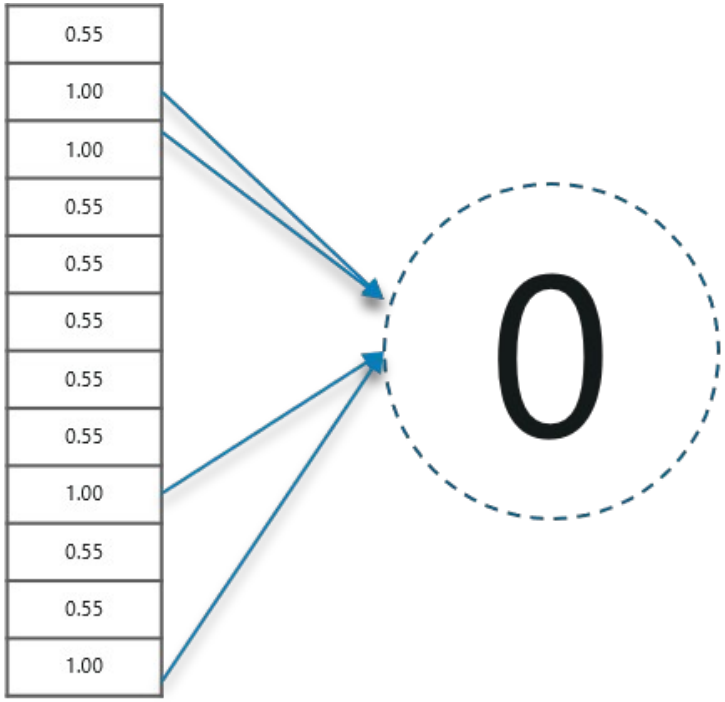
0.55	1.00
1.00	0.55

1.00
0.55
0.55
1.00
1.00
0.55
0.55
0.55
0.55
1.00
1.00
0.55

CNN – Classificação

Organizamos a saida em um único vetor que contém as features que serão consideradas pelo classificador





Daí pode-se aplicar qualquer classificador.

CNN – Exercício

1) O aluno deverá implementar as seguintes funções como explicado em sala:

- ReLU
- Max Pooling

2) O aluno deverá aplicar uma operação de convolução de um filtro detector de bordas (visto na última aula) e em seguida aplicar as operações ReLU e Max Pooling à qualquer imagem do banco de dados Olivetti (usado na última aula).

Deverá ser entregue um pdf com as imagens resultantes de cada uma das três operações.

Referências

- Notas de aula Prof. Antônio Braga (UFMG)
- Material de André Gustavo Hochuli (UFPR)
- https://www.edureka.co/blog/convolutional-neural-network/
- Material Prof. Eduardo