ET-287 – Signal Processing using Neural Networks

3. Feedforward Neural Networks

Professor Sarah Negreiros de Carvalho Leite
Aeronautics Institute of Technology
Electronic Engineering Division
Department of Telecommunications

sarahnc@ita.br, sarah.leite@gp.ita.br
room: 221

Course Syllabus

Unit 1 – Introduction and brief review of linear algebra using Python.

Unit 2 - Introduction to Neural Networks.

a) What is an artificial neural network?

b) The human brain and models of a neuron.

c) Artificial intelligence and neural networks.

d) Neural network architecture.

e) Activation functions.

Unit 3 - Feedforward Neural Networks.

a) Perceptron.

b) Multilayer perceptron.

c) Neural networks with radial basis activation function.

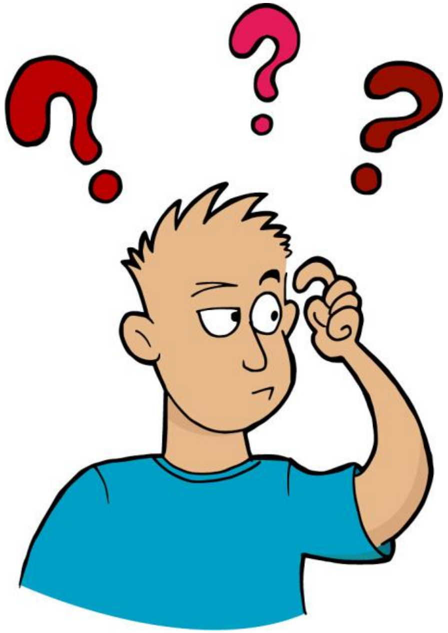
d) Extreme learning machines.

e) Convolutional neural networks.

Unit 4 - Recurrent Neural Networks.

Deep Learning

The accelerated growth in recent years of interest in deep neural networks and deep learning can be attributed to two factors:



1. Increase in the quantity and availability of data.
2. Development of hardware capable of enabling the training of complex structures that demand significant memory and processing power.

But… If a neural network with a nonlinear intermediate layer is already a universal approximator, why use a deep neural network?

Deep Learning

Although both neural network structures (shallow and deep) can solve the problem, sometimes increasing the number of layers is more beneficial than increasing the number of neurons.

The idea of employing **multiple layers** is particularly interesting when the solution to a problem can be **divided into stages**, typically explored in convolutional networks.

Deep Learning

In deep learning, one rule is to let the '***data speak for itself***'.

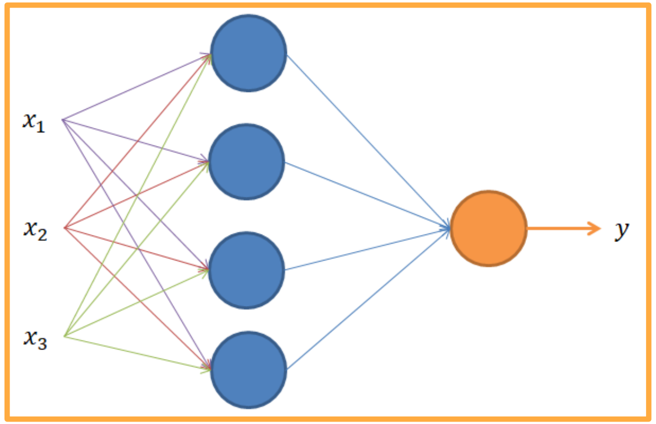
In other words, we want the neural network to learn to solve the problem from the raw information available.

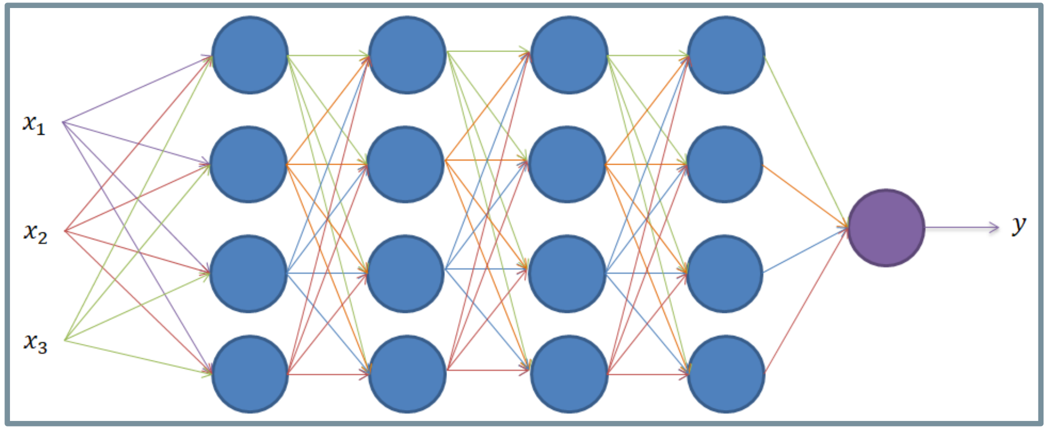
This contrasts with classical approaches where the first goal is to define a convenient way to represent the data, usually through preprocessing and feature extraction.

Deep Learning

The MLP can be considered a deep network when it has **three or more hidden layers**.

This architecture is widely used to solve real-world problems. There are also other neural network structures that exhibit very interesting properties in deep learning, such as convolutional neural networks and recurrent neural networks.





Convolutional Neural Networks

Yann LeCun

Convolutional Neural Networks (CNN or ConvNet) were proposed by the French scientist Yann LeCun in the late 1980s.

Yann LeCun works in the areas of machine learning, computer vision, mobile robotics, and computational neuroscience.

He is a professor at New York University and the Vice President and Chief Scientist of Artificial Intelligence at Facebook.

Introduction to CNN

The CNN is also an artificial neural network structure of the feedforward type that can be employed in both deep and shallow architectures.

However, generally, the CNN is explored in the context of deep learning, with many layers.

Due to their complexity and the need for large datasets and computational power, it was only in the early 21st century that they began to be effectively employed and produced surprising results.

Introduction to CNN

Convolutional networks were inspired by biological processes where the **pattern of connectivity between neurons** resembles the organization of the **visual cortex**.

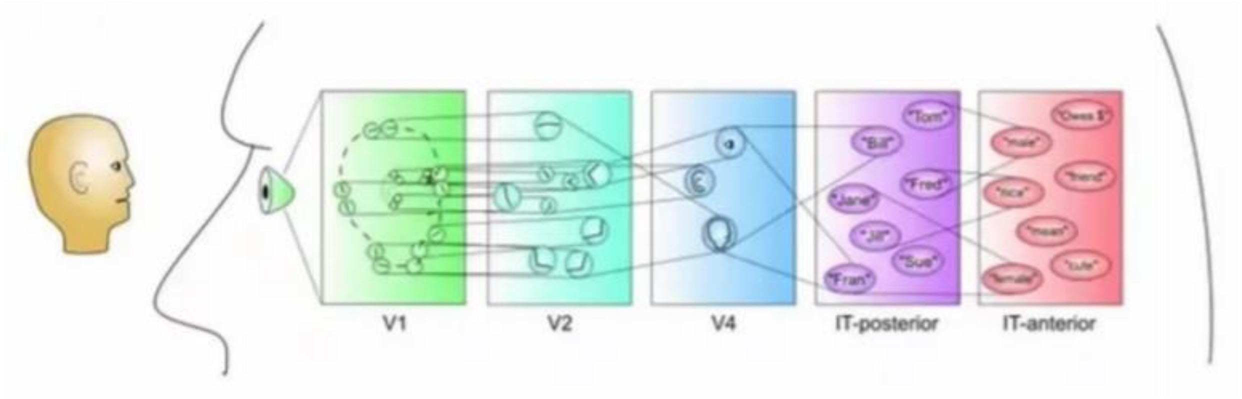
Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the **receptive field.**

The receptive fields of different neurons partially overlap, covering the entire visual field. Our vision is based on multiple levels of the cortex, each recognizing increasingly structured information. Initially, we see individual pixels, then recognize simple geometric shapes, and subsequently distinguish more sophisticated elements such as objects, faces, human bodies, animals, etc.

Introduction to CNN

Each layer of the CNN 'understands' a progressive level of the image.

In a hypothetical example: the first layer recognizes points or patches in the image, the second layer straight or curved lines, the third layer comprehends shapes like rectangles and circles, the fourth layer is capable of identifying, for example, human faces and objects, and the fifth layer can make associations with known people and objects.

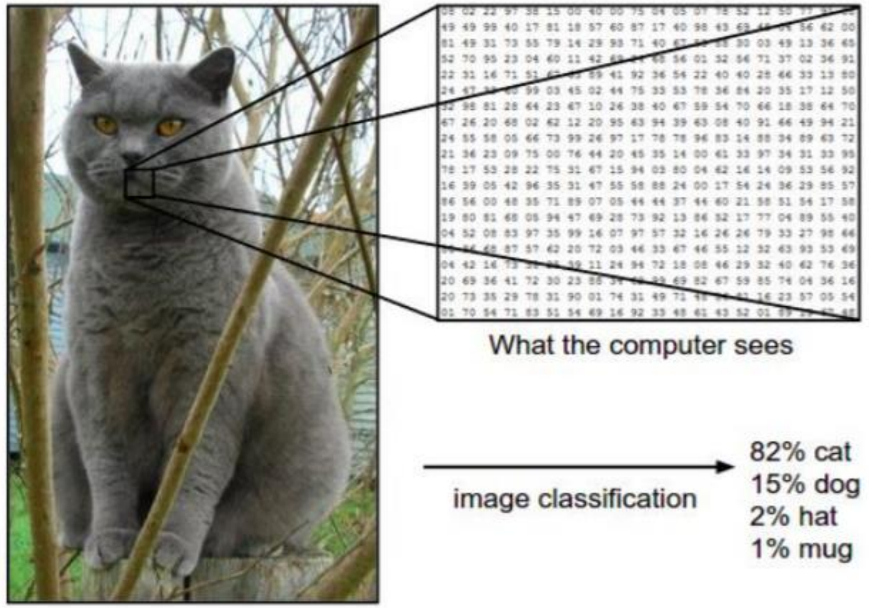


Sourcce: https://www.dca.fee.unicamp.br/~vonzuben/courses/ia353_1s20.html

Introduction to CNN

CNNs enhance **spatial information** and are very suitable for working with **images**.

Our brain can detect any image in fractions of a second, but for the computer, the image is just a sequence of numbers organized in a matrix.



Soruce: http://cs231n.github.io/classification/

Convolution

Convolutional networks are named because they use the convolution operation instead of matrix multiplication in at least one of their layers.

Convolution allows exploring information in organized structures in time (such as, time series) or in space (such as, images).

But, strictly speaking, what is applied is the **cross-correlation operation**, which is taken as synonymous with convolution in machine learning. Cross-correlation provides the similarity between two time series.

Cross-correlation vs. Convolution

Cross-correlation:

$$G = h \times F$$

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} h[u,v]F[i+u, j+v]$$

Convolution:

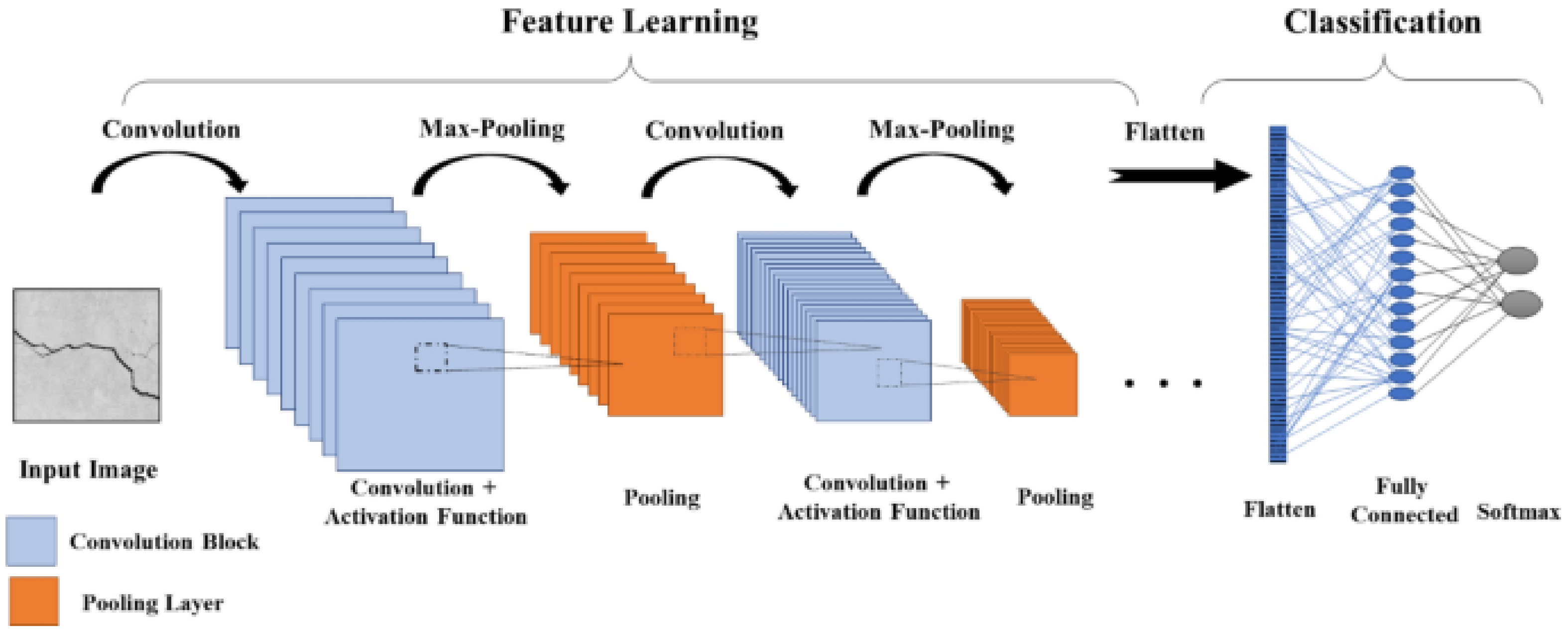
$$G = h * F$$

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} h[u,v]F[i-u, j-v]$$

In a CNN, we typically apply the **cross-correlation** operation between the input data and the filter.
However, by convention, we use the term **convolution**, knowing that it refers to the cross-correlation operation.

Convolution

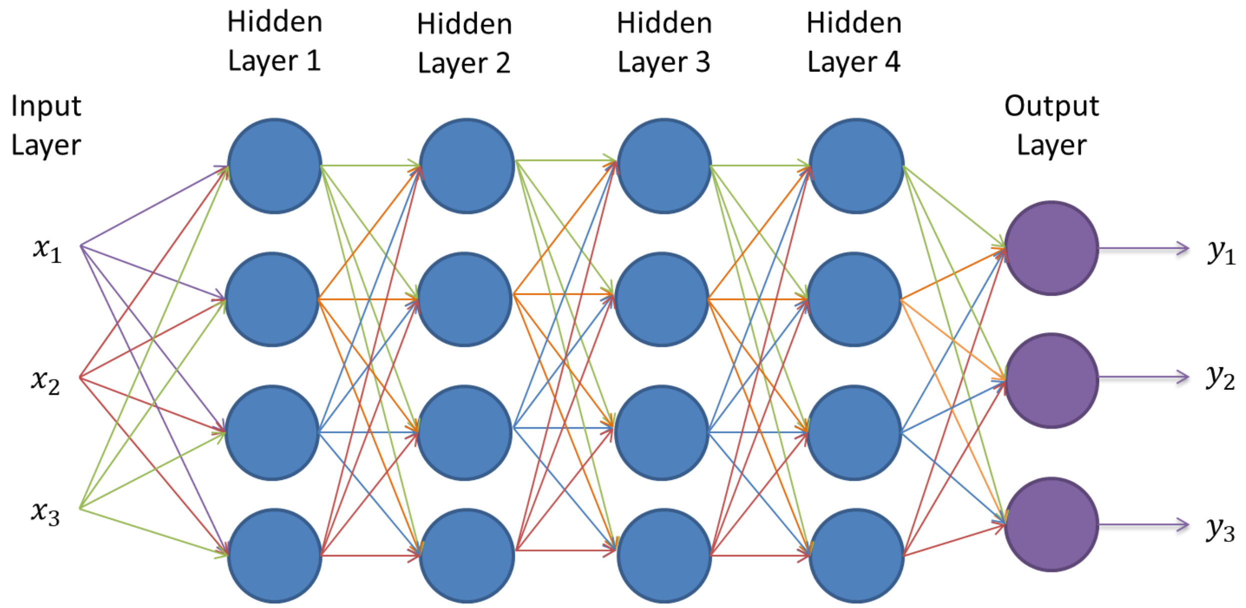
A deep CNN typically consists of an alternation between two different types of layers: **convolutional** and **pooling**, and finally one or more **dense or fully connected** layers.



Source: Bubryur Kim et al. “Surface crack detection using deep learning with shallow CNN architecture for enhanced computation”. Em: Neural Computing and Applications (2021).

Convolution

One of the motivations for the use of convolutional layers is the savings in synaptic connections when compared to the demand for synaptic connections in a fully connected neural network.



Convolution

Multiple fully connected layers are also capable of extracting relevant features from images, but they require a very high number of synaptic connections and neurons, leading to a significant computational load.

It's important to note that each neuron is a non-linear filter that performs the inner product calculation with the entire image produced by the previous layer.

Convolution

The convolutional layers include the following hyperparameters:

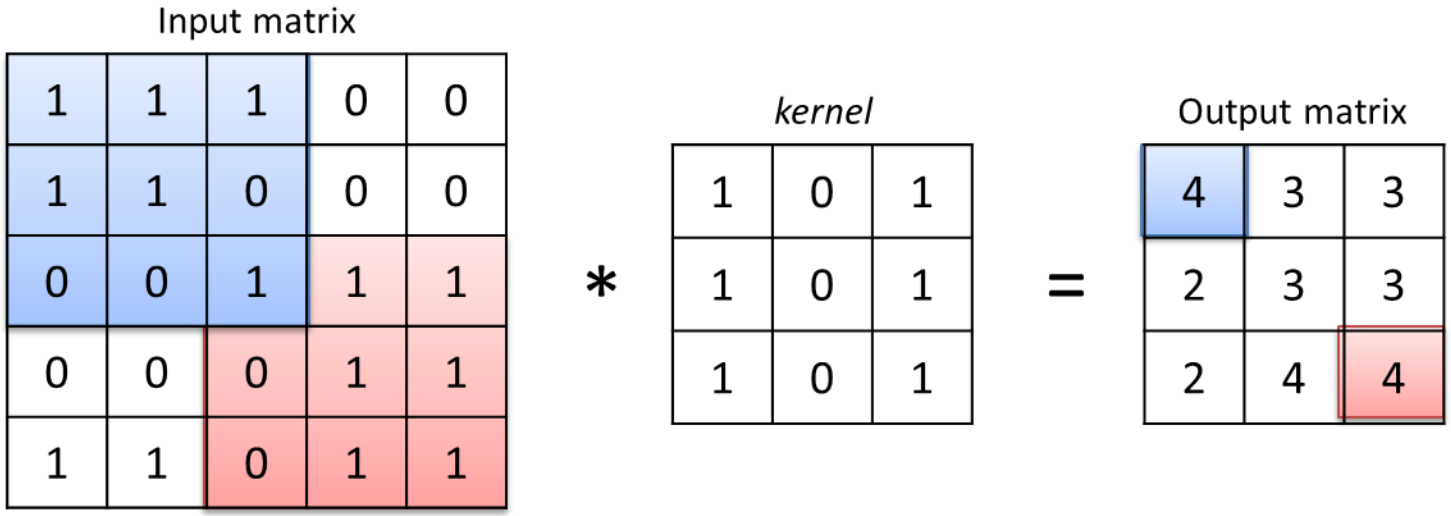
- **Kernel size:** the size of the window used for convolution.
- **Stride:** the size of the step the kernel takes when sliding the window.
- **Padding:** defines how the sample's edge is treated.

Understanding the hyperparameters

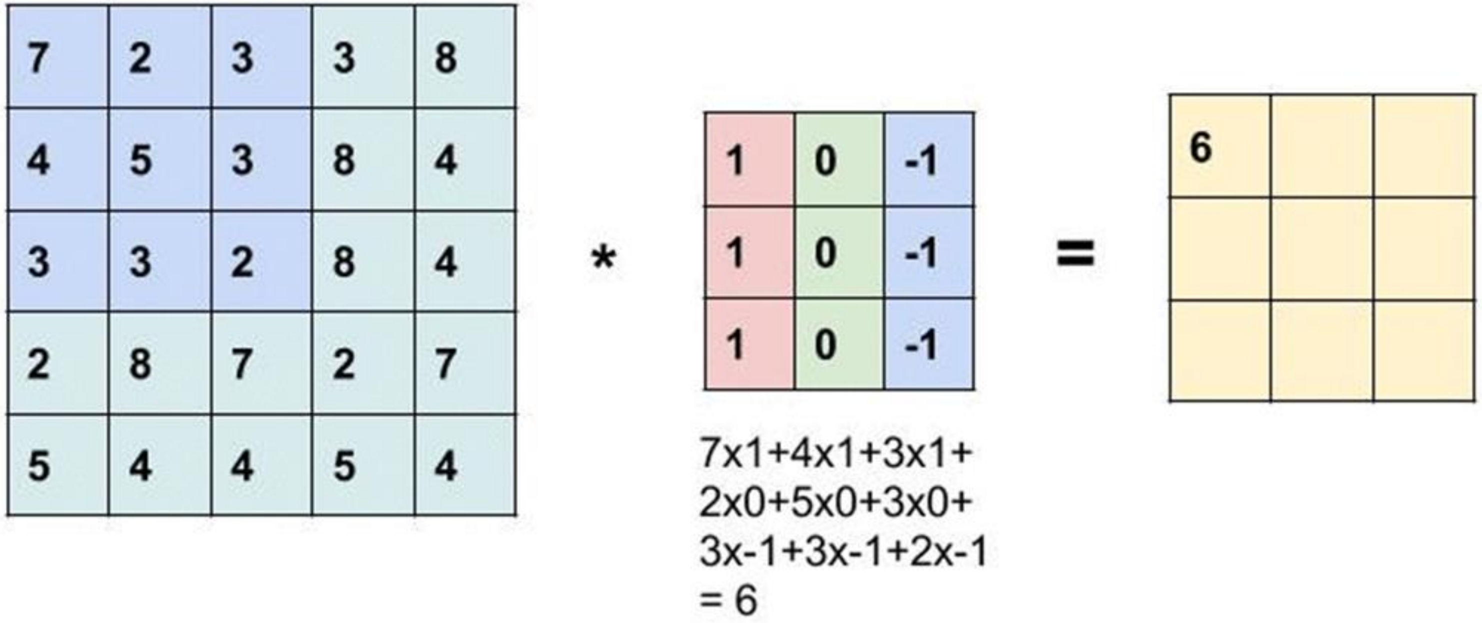
- Example of convolution using a 3 x 3 kernel:

1.1+1.1+0.1+0.1+0.1+0.0+1.1+0.1+1.1 = 4

1.1+0.1+0.1+1.0+1.0+1.0+1.1+1.1+1.1 = 4



Understanding the hyperparameters

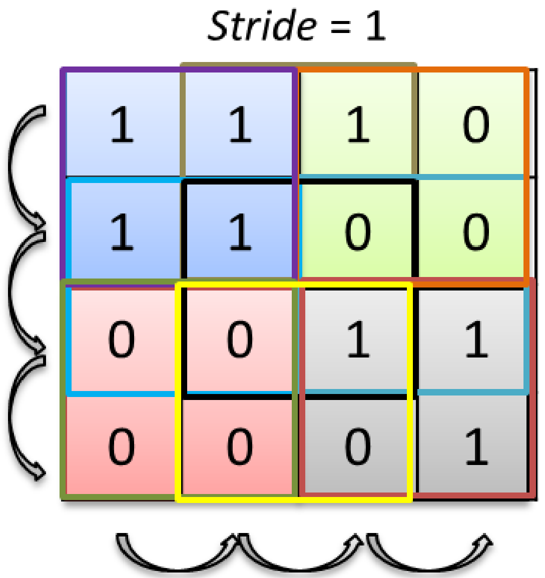


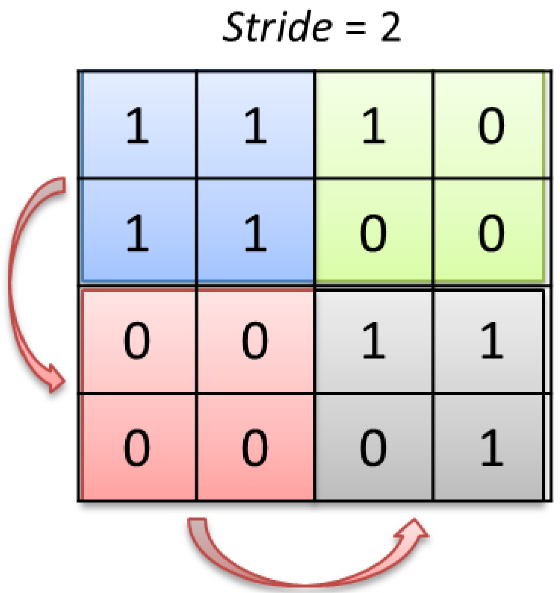
Fonte: https://medium.datadriveninvestor.com/convolutional-neural-networks-3b241a5da51e

Understanding the hyperparameters

- Example of Stride:

By skipping some positions in the multiplication, we reduce computational cost, aware that the feature extraction will be less detailed.

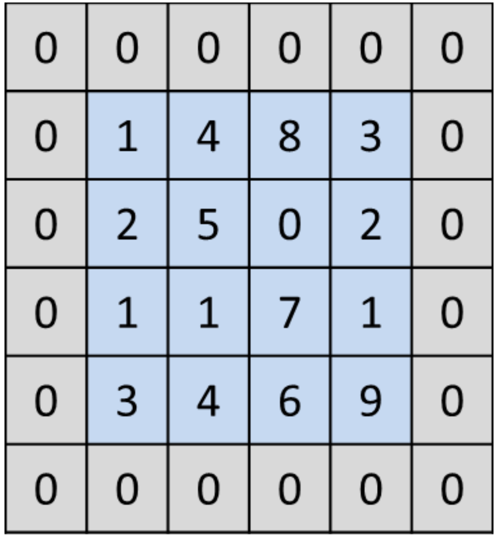




Understanding the hyperparameters

- Example of Padding:

Padding allows for better handling of edge data in the data matrix.



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

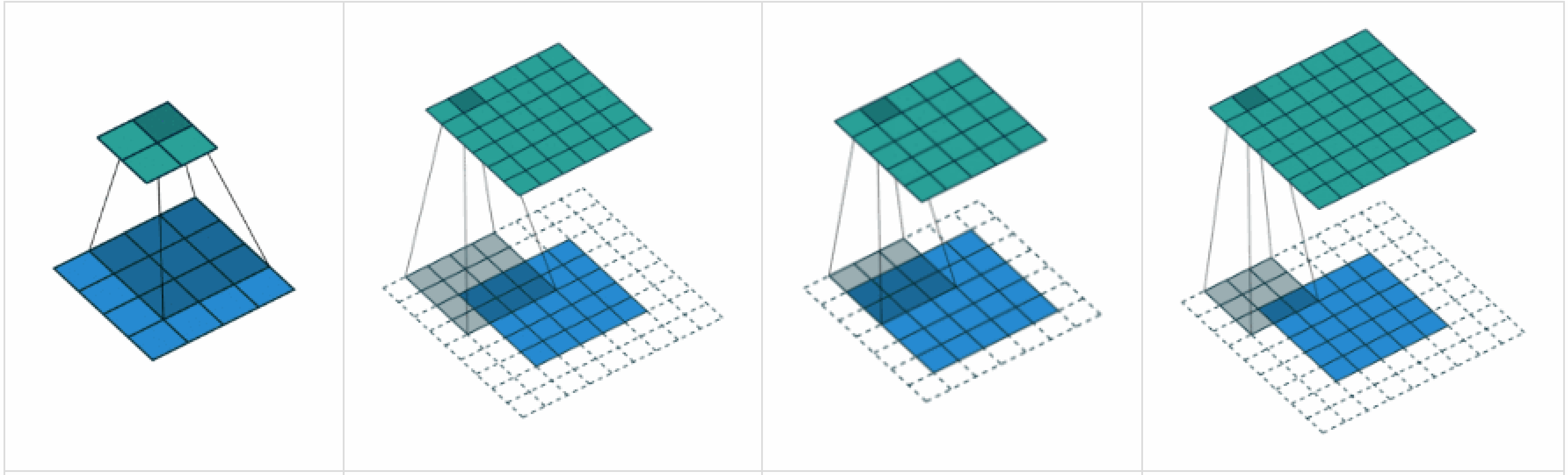
Understanding the hyperparameters

In Keras the padding can be performed in two ways defined by the parameters 'VALID' or 'SAME'.

- 'VALID': The input image is not padded. Only valid and original elements of the input image are considered. There may be loss of information, as depending on the kernel size and stride, elements to the right and at the bottom of the image may be ignored. The output image always has the same size or is smaller than the input image.

- 'SAME‘: The input is padded with zeros, and the size of the output image is equal to or larger than the size of the input image. This padding option ensures that the filter is applied to all elements of the input image.

Padding ‘VALID’ vs. ‘SAME’



The first image illustrates a padding = 'VALID', with the filter constrained by the input image.

The other cases exemplify padding = 'SAME', where the size of the output image is equal to or larger than that of the input image.

Fonte: https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480

Pooling Layer

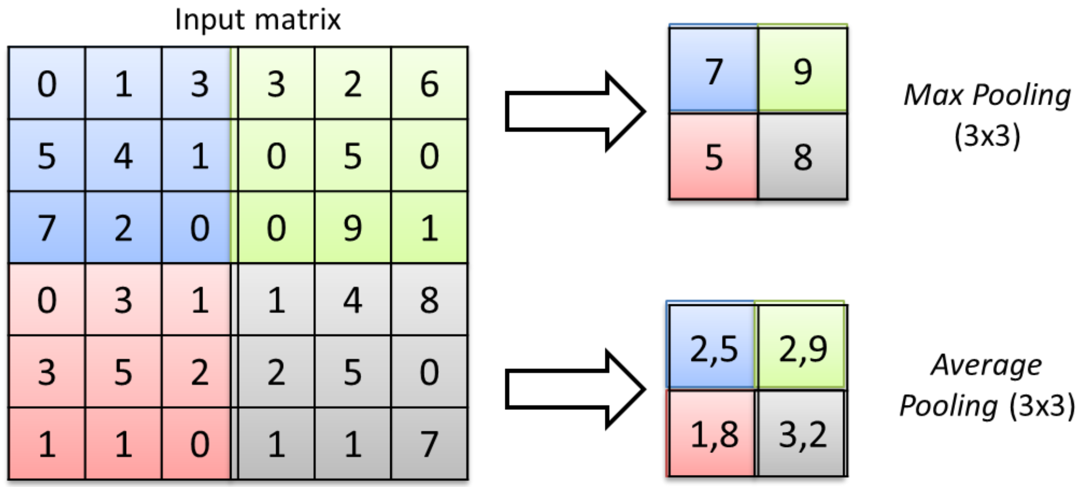
These layers typically follow the convolutional layer. They reduce a set of input data to a single number, decreasing the data passing through the neural network.

The pooling operation causes the analyzed output to be influenced by a set of neighboring outputs. It is crucial for making the representation approximately invariant to small translations of the input, meaning that if the input is slightly translated, the output values (with pooling) tend to remain the same. This property is essential to enable the neural network to recognize patterns in different positions of an image.

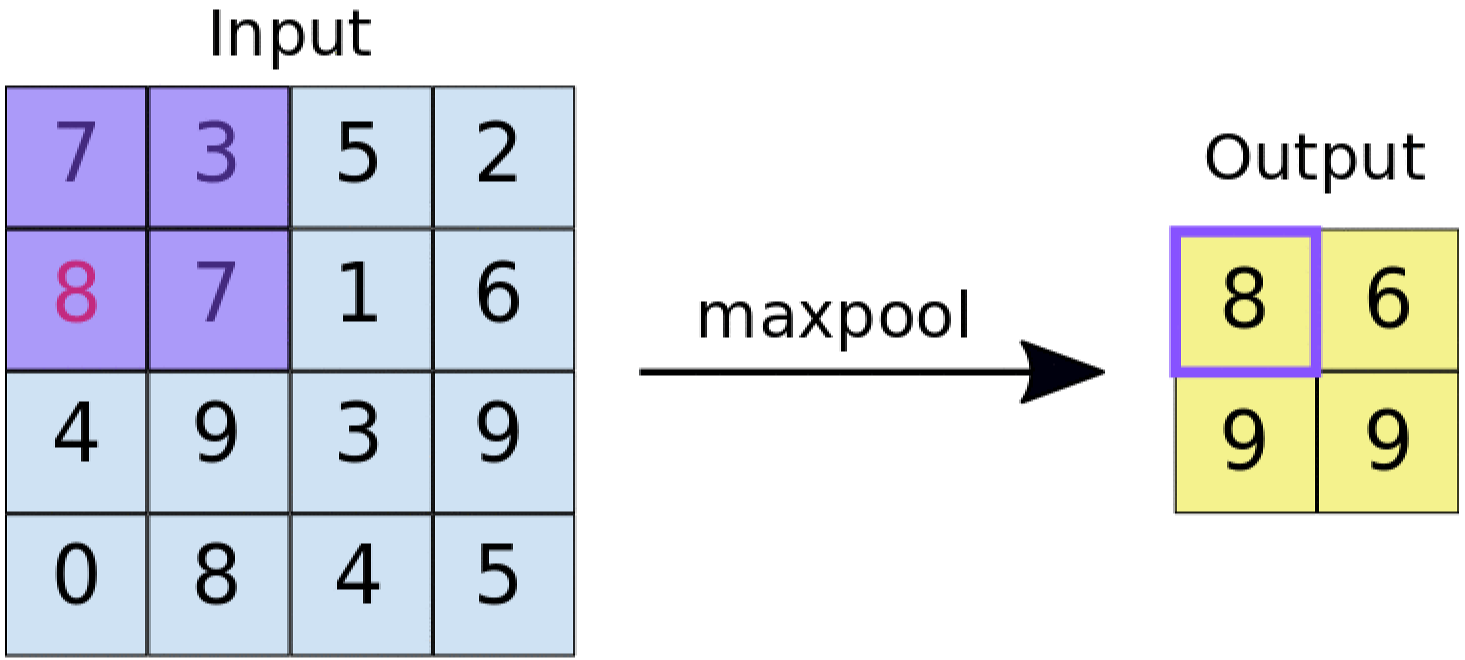
Pooling Layer

A commonly used pooling approach is **max pooling**, which synthesizes information by copying the maximum value observed in the analyzed region.

Another option is to use **average pooling**, which provides the average values observed in each region.



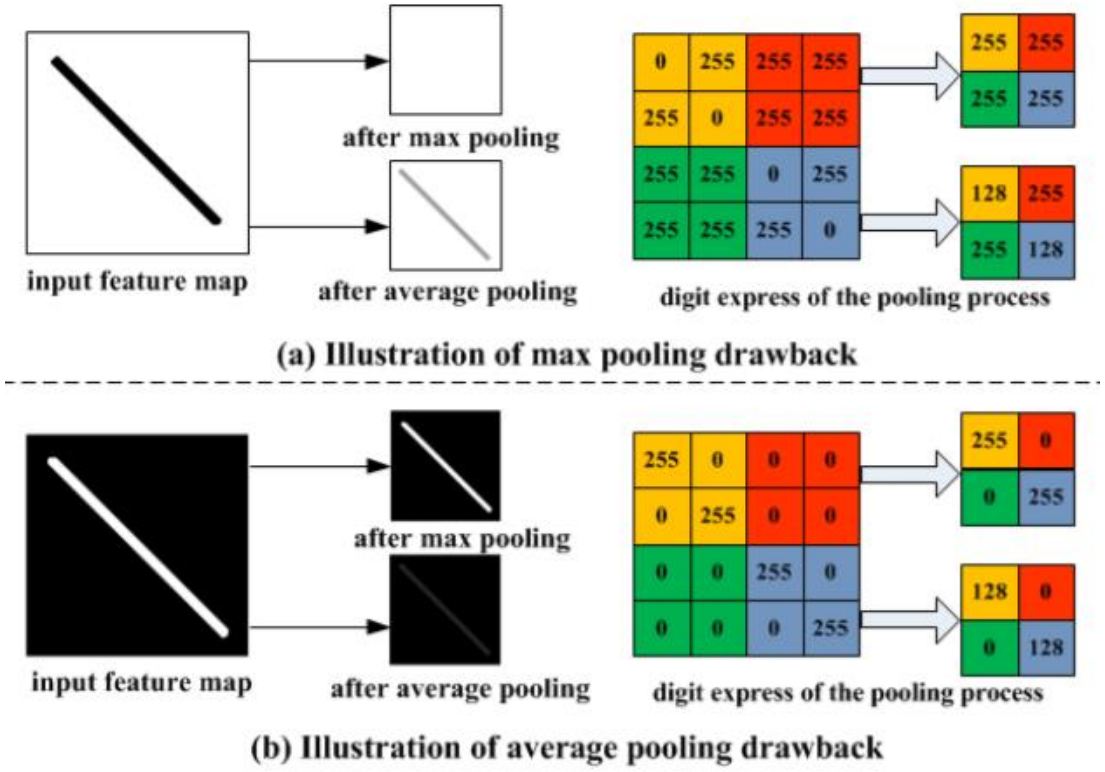
Example of max pooling



Source: https://medium.com/neuronio/understanding-convnets-cnn-712f2afe4dd3

Pooling Layer

Despite the advantages of pooling, we should be aware that it can lead to information loss.

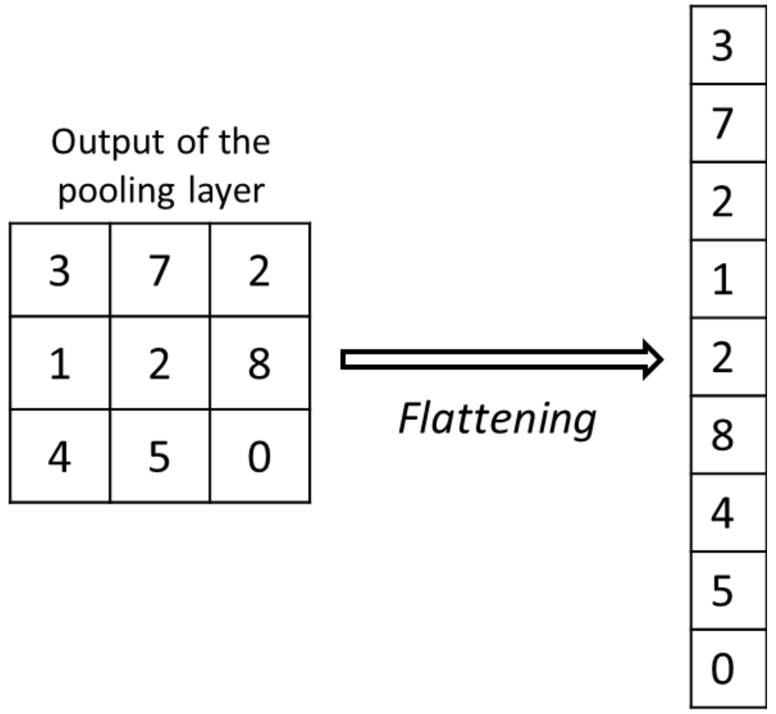


Source: https://www.dca.fee.unicamp.br/~vonzuben/courses/ia353_1s20.html

Fully connected layer

All inputs connect to all neurons. This layer is typically found at the end of the architecture of a CNN. For example, it can provide the probability of the input belonging to each possible class.

Generally, this layer is preceded by the 'flatten' operation, which converts the output matrix from the last pooling layer into a data vector with a single column.

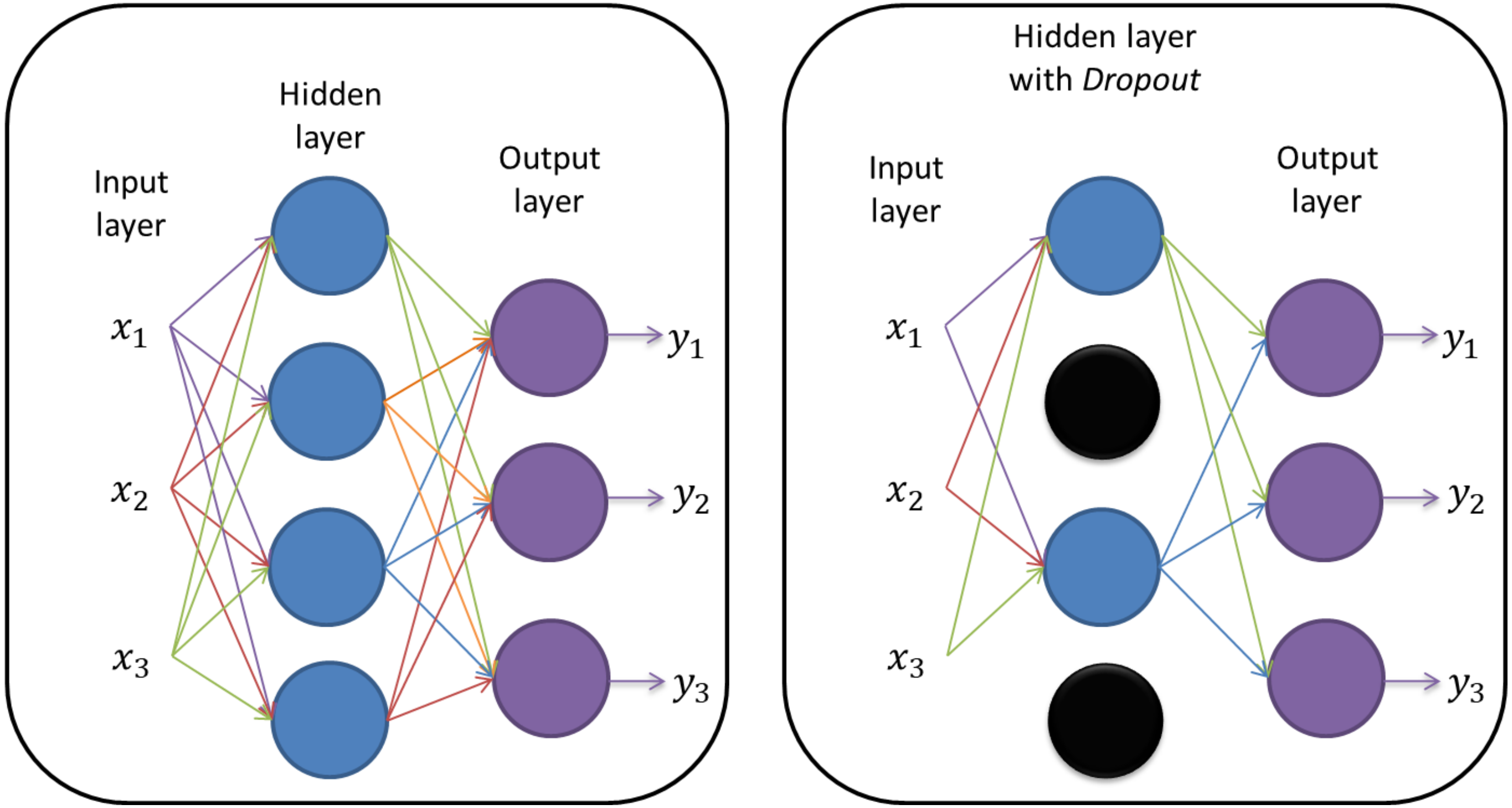


Dropout layer

Given the enormous flexibility associated with the input-output mappings that can be performed by a deep CNN, overfitting becomes relevant. One way to mitigate this problem is to use the dropout layer. This layer is only used during the training phase and is removed during the testing phase.

Dropout involves the random elimination of neurons from the neural network, along with their synaptic connections. In other words, we force their output to zero with a fixed probability. This procedure forces the neural network to be redundant and capable of classifying correctly even in the absence of some neurons. Consequently, the network learns not to depend on any specific neuron, apparently preventing overfitting.

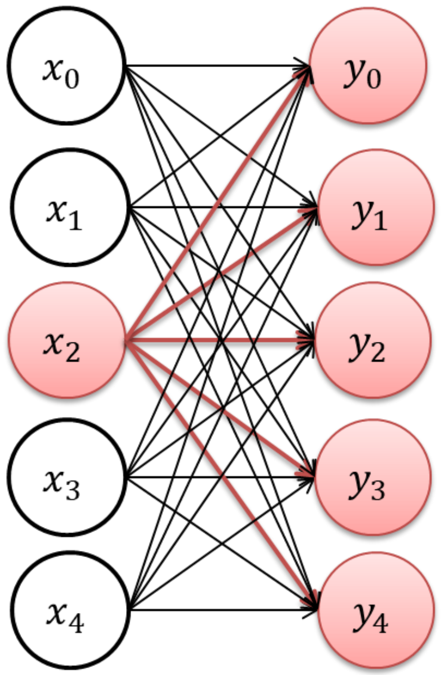
Dropout layer



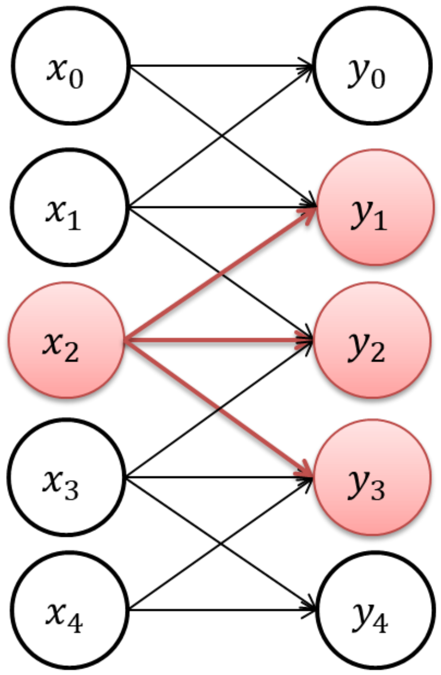
Local receptive fields

Unlike fully connected networks like the MLP, the convolutional layer leads to sparse input/output interactions, preserving spatial information

Observe how input x_2 affects the outputs in each case.



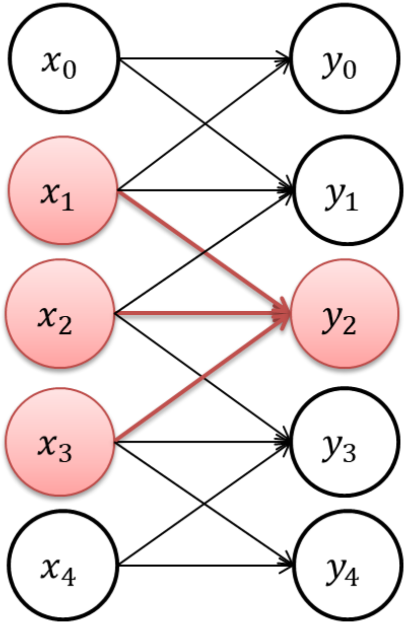
Fully connected



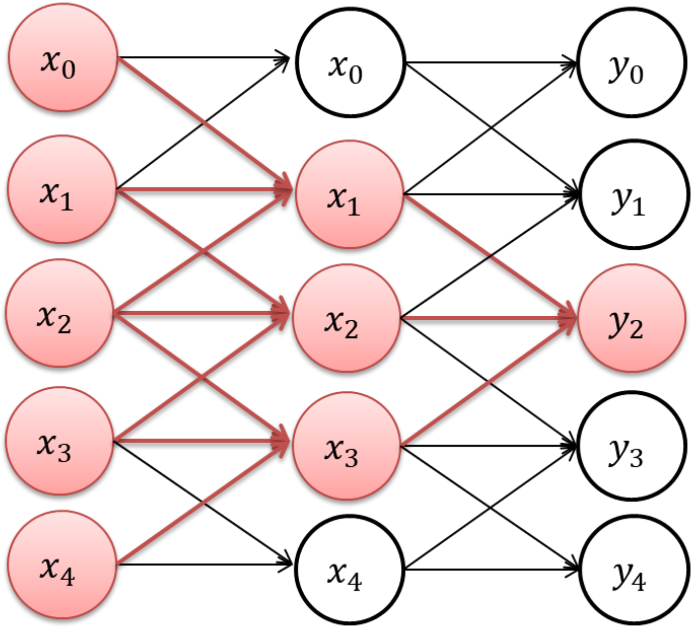
Convolutional

Local receptive fields

For a deep network, it can be observed that the receptive field significantly expands, even with sparse connections.



Receptive field- 2 layers

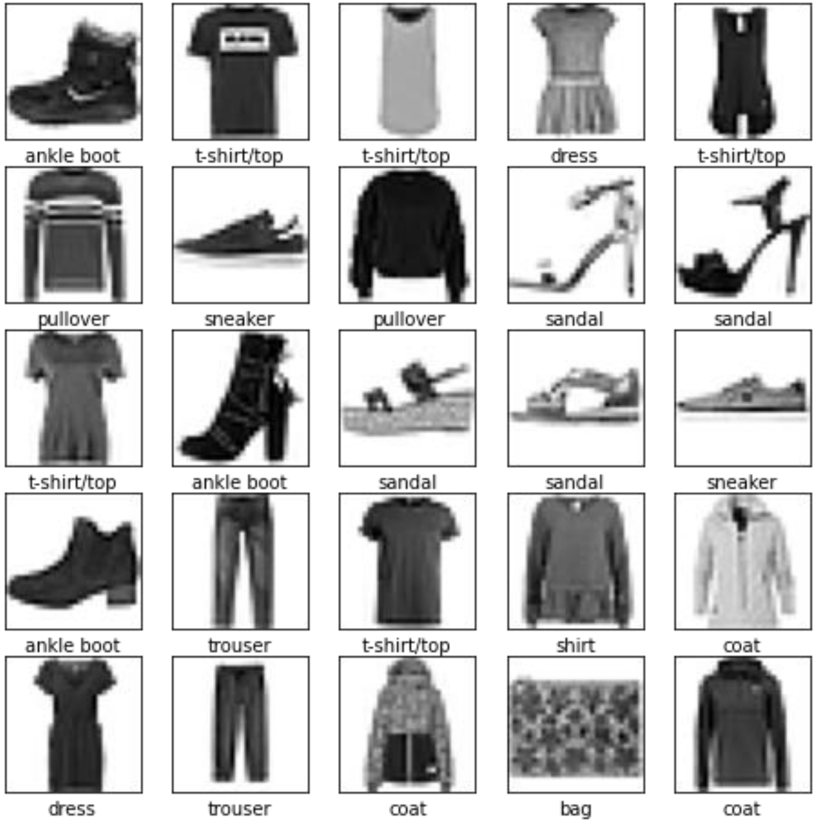


Receptive field- 3 layers

Exercise: Fashion-MNIST image recognition using CNN

The Fashion-MNIST database consists 60 thousand images for training and 10 thousand for testing. All images are grayscale and 28x28 pixels.
There are 10 fashion categories in it.

The database is available in Keras: https://keras.io/api/datasets/fashion_mnist



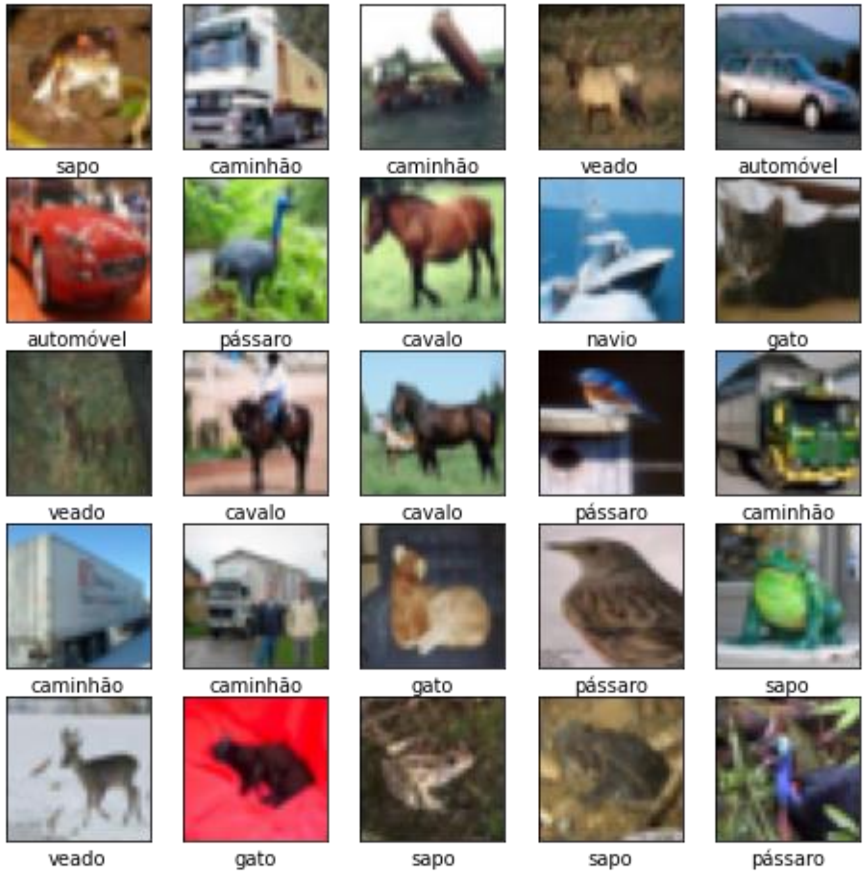
Exercise: CIFAR10 image recognition using CNN

CIFAR stands for the Canadian Institute for Advanced Research, and the CIFAR-10 dataset was developed along with the CIFAR-100 dataset by researchers from the CIFAR institute.

This database consists of 60 thousand color photographs of 32 × 32 pixels and contains photos belonging to 10 classes.

The CIFAR-10 database is available in Keras: https://keras.io/api/datasets/cifar10/

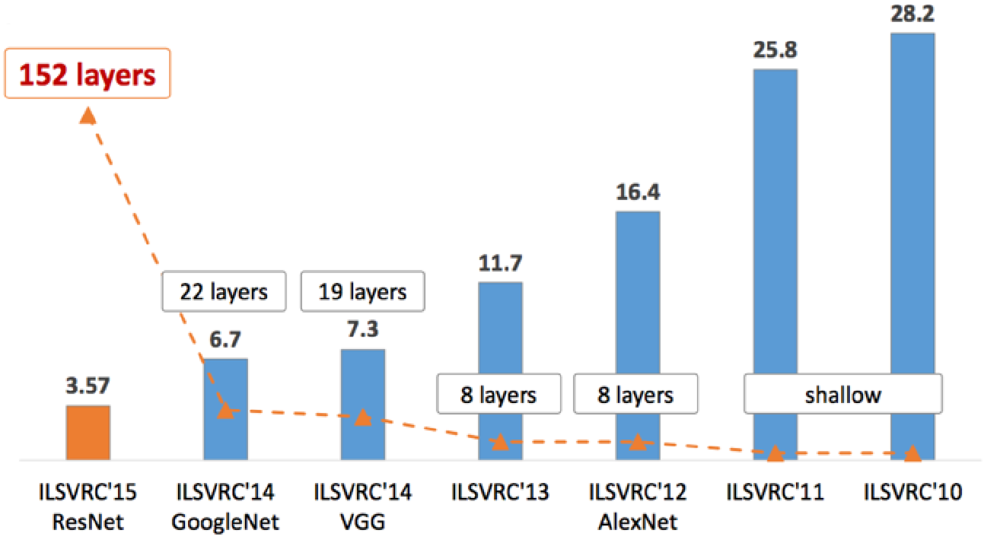
It is a classification problem that employs computer vision and deep learning.



Classic architectures of convolutional neural networks

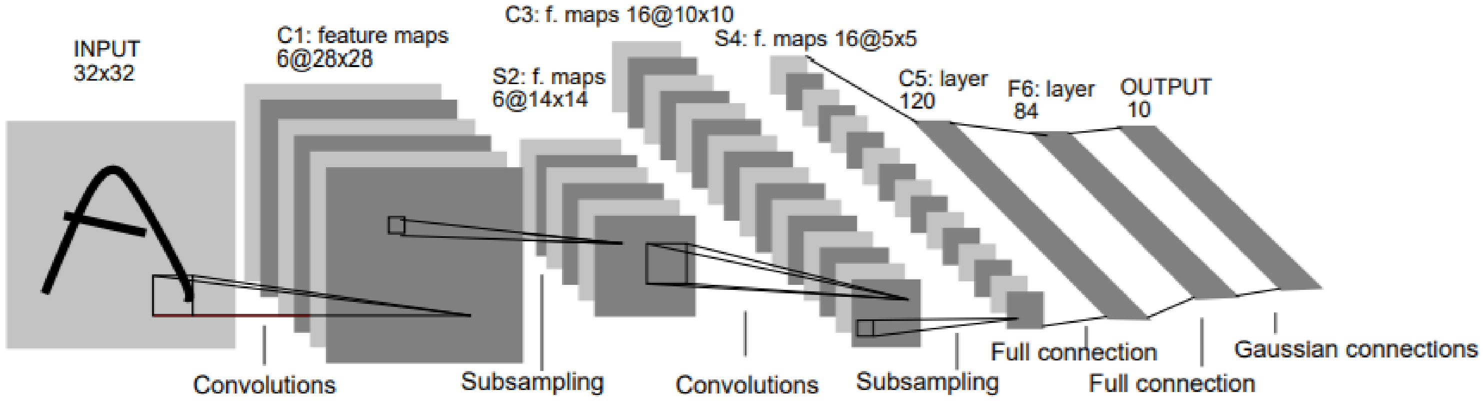
The ImageNet project provides a massive image database designed for use in visual object recognition software research.

Annually, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) takes place, proposing classification systems that can detect objects and scenes with better accuracy.



LeNet

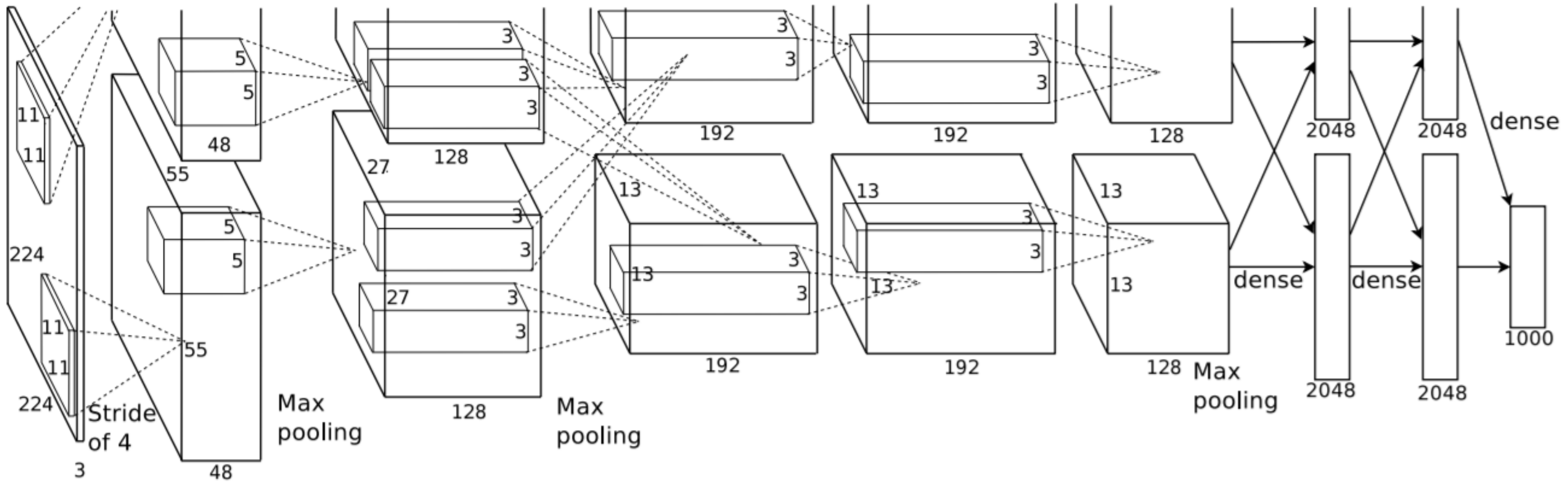
LeNet is a 7-level convolutional network that classifies digits. It was proposed by Yann LeCun et al. in 1998 and is applied, for example, to recognize handwritten digits on postal codes in mail and numbers on scanned checks. The ability to process higher resolution images requires more convolutional layers, so this technique is limited by the availability of computing resources.



AlexNet

In 2012, the AlexNet architecture significantly outperformed all competitors, reducing the classification error rate.

The AlexNet network has a similar architecture to LeNet but is deeper, has more filters per layer, and has stacked convolutional layers.

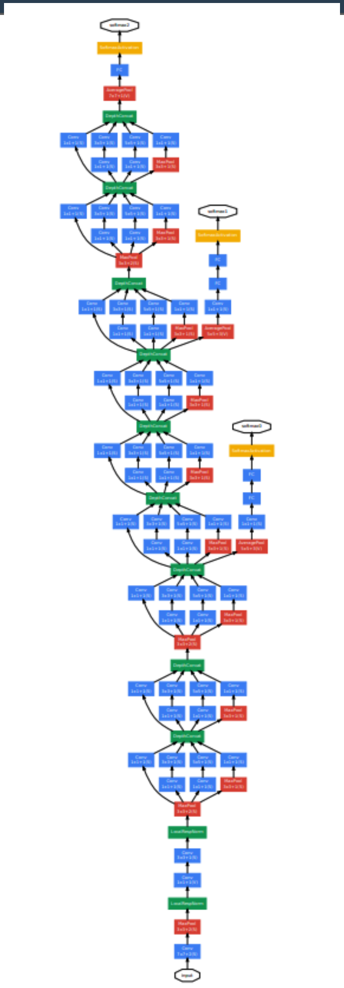


GoogLeNet

The winner of the ILSVRC 2014 competition was the GoogLeNet architecture (also known as Inception).

It achieved an error rate close to human-level performance, forcing challenge organizers to reevaluate the images.

This CNN network is also inspired by the LeNet architecture and features 22 layers.

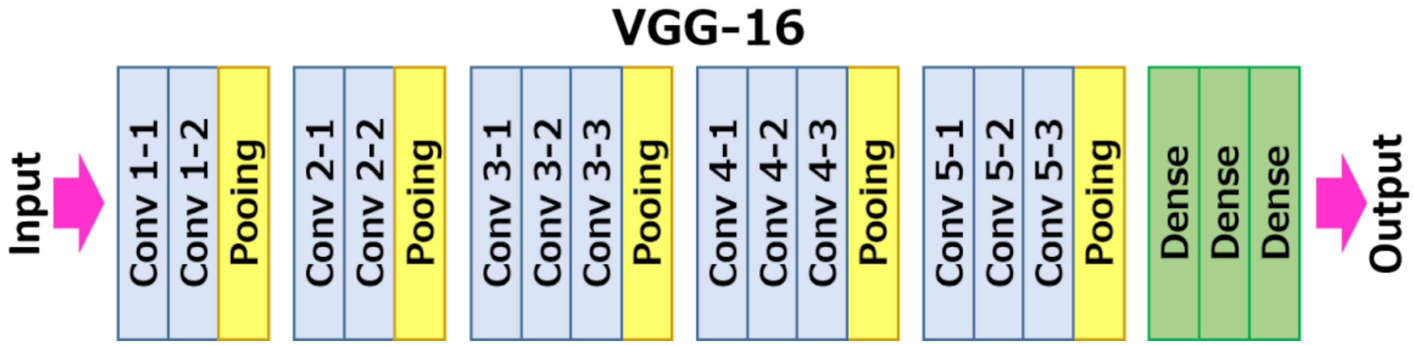


VGG

The VGG or VGGNet architecture was developed by Simonyan and Zisserman and came in second place in the ILSVRC 2014 competition.

VGGNet consists of 16 or 19 convolutional layers and is very attractive due to its uniform architecture.

It is one of the most commonly used architectures for working with images.



Exercise: Imagenet

ImageNet é um enorme banco de dados de imagens construído para pesquisa e desenvolvimento de algoritmos de reconhecimento visual de objetos. Desde 2010, o projeto ImageNet promove anualmente a competição ImageNet Large Scale Visual Recognition Challenge (ILSVRC), onde programas competem para classificar corretamente e detectar objetos em imagens.

Exercise: Imagenet

Training a convolutional neural network (CNN) using the ImageNet dataset demands substantial computational resources. However, it's feasible to leverage pre-trained models with weights already fine-tuned on ImageNet, ready to be utilized or adjusted for other image recognition tasks.
These pre-trained models are accessible, for instance, at: https://keras.io/api/applications/.

In this exercise, we'll employ a pre-trained model trained on the ImageNet dataset for image recognition.

1. Download the model, customize the code, and fine-tune the parameters to integrate the new model for image recognition.
2. Load some images and understand how the output is generated.
3. Repeat the process using an alternative model.
4. Compare the performance of both models in classifying at least 3 images.

Project 4 – Target identification in radar signals

Gif retirado de: CassidyGE, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=36734692