

# Deep Learning and Neural Network

Marie Beurton-Aimar, LE Van Linh

January 17, 2022

# Artificial Intelligence?

## The fathers

- Lee Marvin Minsky and John McCarthy created the expression - Working at MIT in the fifties.
- Foundation at the Dartmouth conference on summer 1956.
- Minsky defined AI as “the science of making machines do things that would require intelligence if done by men.”
- On the road to achieve the Turing machine.

# Artificial Intelligence?

## The fathers

- Lee Marvin Minsky and John McCarthy created the expression - Working at MIT in the fifties.
- Foundation at the Dartmouth conference on summer 1956.
- Minsky defined AI as “the science of making machines do things that would require intelligence if done by men.”
- On the road to achieve the Turing machine.

## Copying human being?

- Learning is the main characteristic of Intelligence.
- In the 80's symbolic approach wins against connexionist approach.
- End of the first turn !

# Back to the futur

## ECCV 2012

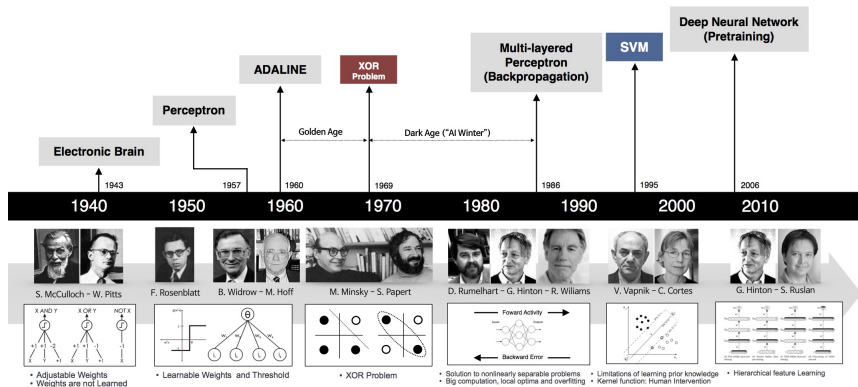
- Hinton, Krizhevsky and Sutskever propose the first model to classify ImageNet DB with a deep network - and outperform (g.t. 10%) all the results.

<https://hal.archives-ouvertes.fr/hal-01925644/document>

## Neural Network revival

- Reuse the architecture proposed by Lecun more than 20 years before (listen the Collège de France talk of Y. Lecun to know more about the main principles).
- Intelligence is a connexionist system with layers - composition of simple layers.

## History



[beamandrew, 2017]

# Deep Learning today

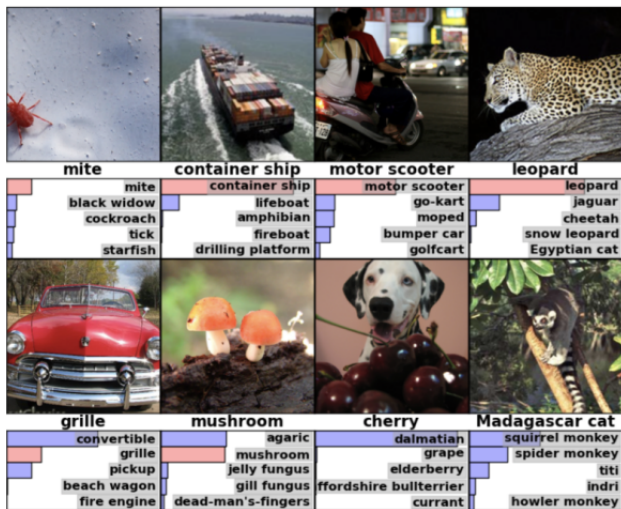
## Why using Deep learning?

- Production of huge amount of Data by machines.
- Evolution of Data Mining domain/statistical methods.
- Extraction of features to classify data, to predict results ...
- Pretty all domains are concerned: machine log, images, medical/biological data ...

## Requirements

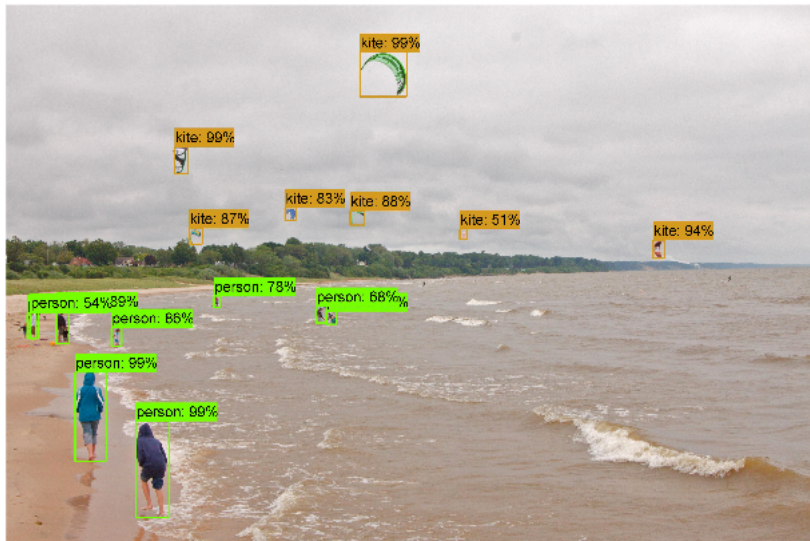
- Data storage and computing resources.
- Way to display and to perform the results.
- Graphical interfaces and statistics tests are mandatory.

# Deep Learning - Images classification



[Machine Learning Mastery, 2016]

# Deep Learning - Object detection



[Medium, 2018]



# Deep Learning - Machine translation



## Traduction

Anglais

Français

Vietnamien

Détecter la langue



bonjour  
Je m'appelle Linh



# Deep Learning - Speech recognition



## Deep learning is a class of machine learning that:

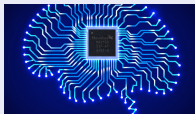
- use a cascade of multiple layers for extracting and transforming the features.
- learn in supervised or unsupervised mode
- learn with different levels of presentation



[ZDNet, 2018]



[Siecle Digital, 2016]



[Silicon ANGLE, 2016]

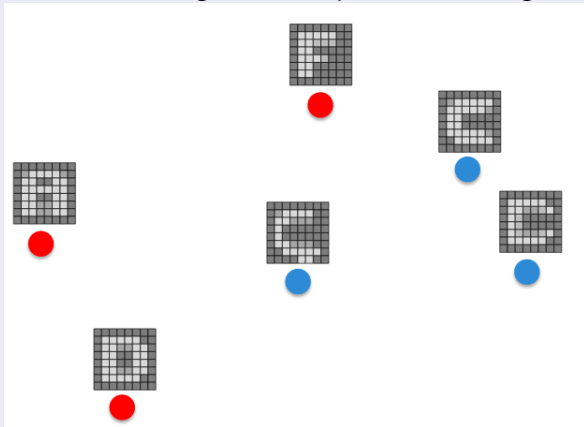


ROSS Intelligence, 2018

# Neural architecture

## Perceptron bases

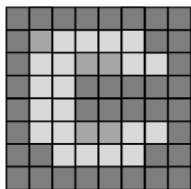
- Collect data for training: includes positive and negative examples



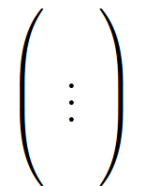
# Neural architecture

## Perceptron bases

- Represent data as vectors:



Image

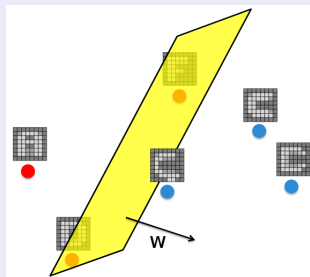


vector

# Neural architecture

## Perceptron bases

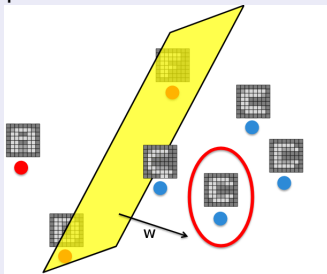
- Function:  $f(x) = w^T x + b$
- Train the function on training data: find  $w$  and  $b$  so that
  - If sample  $x$  is positive,  $f(x)$  is positive
  - If sample  $x$  is negative,  $f(x)$  is negative



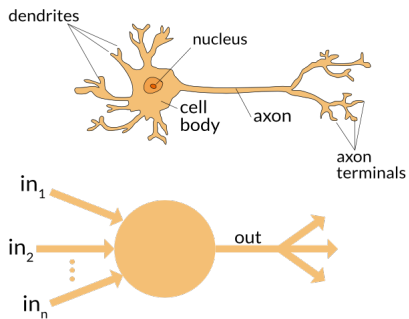
# Neural architecture

## Perceptron bases

- Testing on new examples



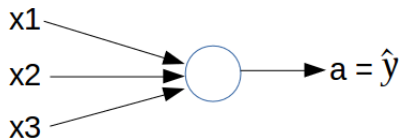
# Biological Inspiration



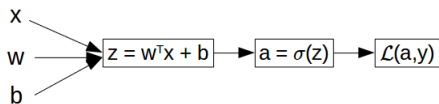
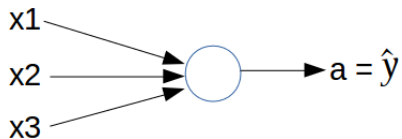
[appliedgo, 2016]



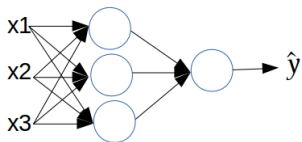
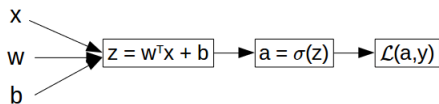
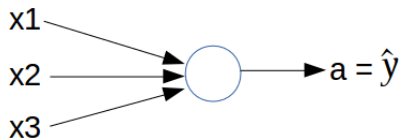
# What is an artificial neural network?



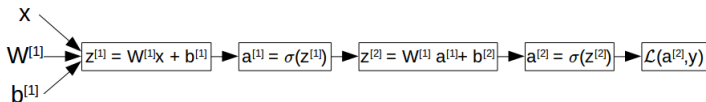
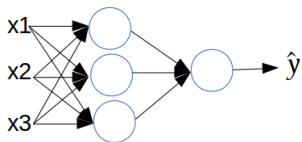
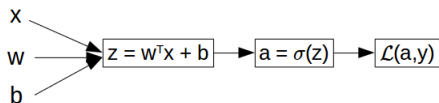
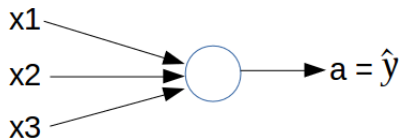
# What is an artificial neural network?



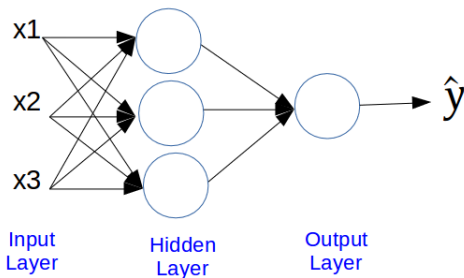
# What is an artificial neural network?



# What is an artificial neural network?



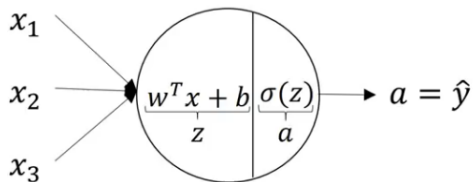
# NN Representation



This is a:

- 2 layers neural network
- The hidden layers and output layer (sometime) will have the parameters  $(W,b)$

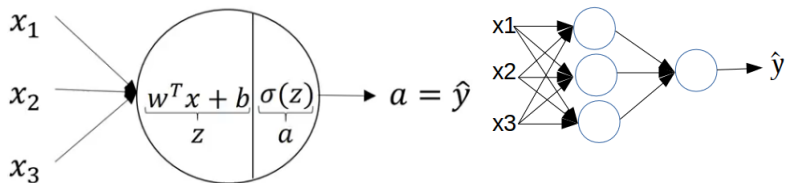
# NN Computing



$$z = w^T x + b$$

$$a = \sigma(z)$$

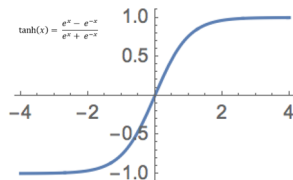
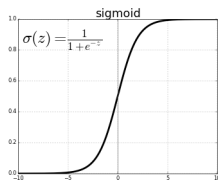
# NN Computing



$$z = w^T x + b$$

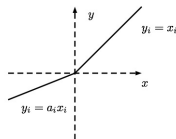
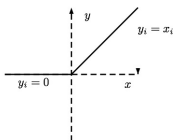
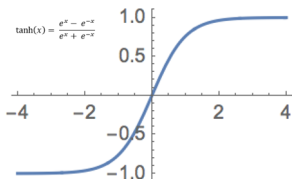
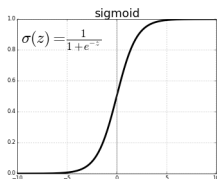
$$a = \sigma(z)$$

# Activation functions



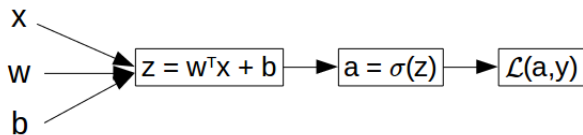


# Activation functions

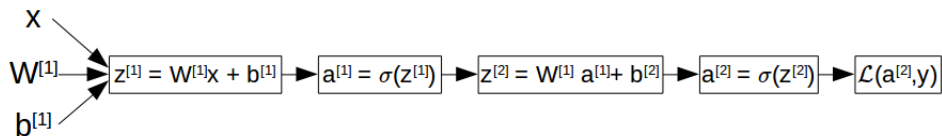
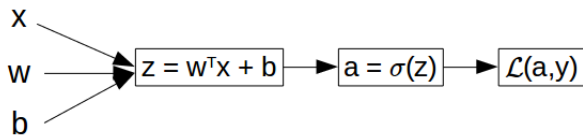


- For hidden units: use tanh, ReLU should be better than sigmoid
- For output layer ( $0 \leq \hat{y} \leq 1$ ), we can use sigmoid.

# Backward propagation intuition

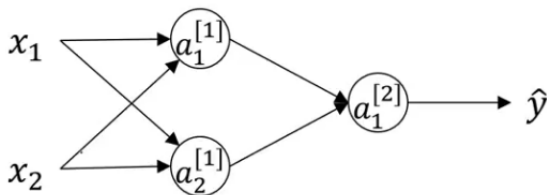


# Backward propagation intuition



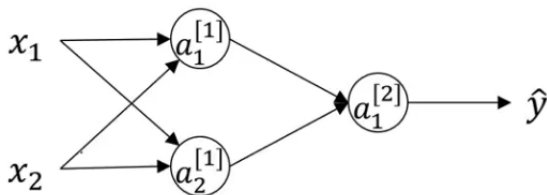
# Parameters random initialization

If  $W = [0]$  then the computing of neural network is exactly the same function at every layer?



# Parameters random initialization

If  $W = [0]$  then the computing of neural network is exactly the same function at every layer?



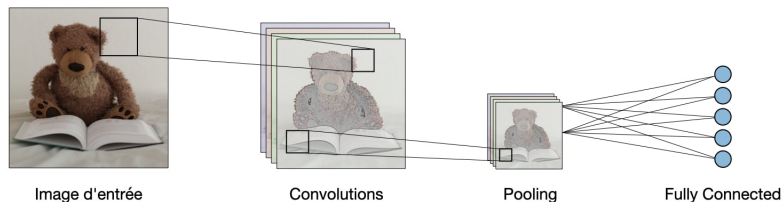
**Solution:** Randomly initialization the parameters

- $W = \text{np.random.randn}((2,2)) * 0.01$
- $b = \text{np.zeros}((2,1))$

# Convolutional Neural Network

## Introduction

- In CNN we identify :
  - Input data that has to be transform.
  - Convolutional layers
  - Pooling layer(s)
  - Fully connected layer.



# Convolutional Neural Network

## Goal

- Extract features of the input volume.
- Need to set the size of each filter on the layers. Could be different for each layer.

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input

1	0	1
0	1	0
1	0	1

Filter / Kernel

# Convolutional Neural Network

## Goal

- Extract features of the input volume.
- Need to set the size of each filter on the layers. Could be different for each layer.

1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

Input x Filter

4		

Feature Map



# Convolutional Neural Network

## Goal

- Extract features of the input volume.
- Need to set the size of each filter on the layers. Could be different for each layer.

1	1x1	1x0	0x1	0
0	1x0	1x1	1x0	0
0	0x1	1x0	1x1	1
0	0	1	1	0
0	1	1	0	0

Input x Filter

4	3	

Feature Map

# Convolutional Neural Network

## Goal

- Extract features of the input volume.
- Need to set the size of each filter on the layers. Could be different for each layer.

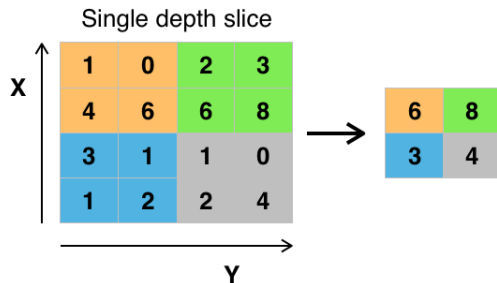
1	1	1	0	0
0	1	1	1	0
0	0	1x1	1x0	1x1
0	0	1x0	1x1	0x0
0	1	1x1	0x0	0x1

4	3	4
2	4	3
2	3	4

# Convolutional Neural Network

## Pooling Layer

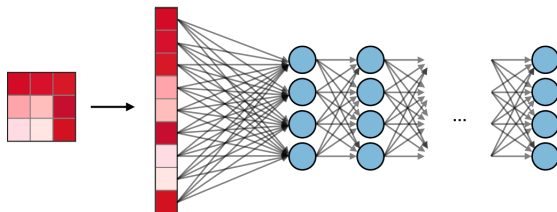
- Pooling layer reduces the spatial dimensions of the input.
- Allows to control **overfitting**.
- Several functions are available: max pooling, average pooling, L2-norm ....



# Convolutional Neural Network

## Fully Connected Layer

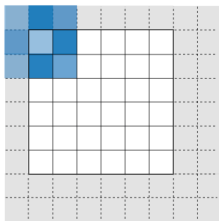
- Use as input a vector where each data is connected to a neurone.
- Main goal is to achieve the task - classification, scores...



# Convolutional Neural Network

## Parameters

- Stride : pixel number of the sliding window for convolution or pooling.
- Zero-padding : adding value to adjust image/matrix size.



# Convolutional Neural Network

## Parameters

- Batch size : define the number of samples to work through before updating the internal model parameters.
- Learning rate : control how much to change the model in response to the estimated error each time the model weights are updated.
- Epochs number : define the number times that the learning algorithm will work through the entire training dataset.

# Convolutional Neural Network

## Evaluation - Metric

- Loss value : evaluate how the model is training - produced from the loss function that estimates the loss of the model so that the weights can be updated to reduce the loss on the next evaluation.
- Accuracy : a metric to measure the algorithm performance at the end.
- Mean Average Precision : compute the average precision value for recall value over 0 to 1. See Precision/Recall.
- F1 Score : combine precision and recall relative to a specific positive class. It is a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst at 0

# Activation Functions -

## Goals

- Modified in a non-linear way the representation of data.
- As multi-layers are activated, several modifications can occur.
- Don't confuse between activation function and loss function.

## Way to work

- Each neurone of one layer will apply the activation function of this layer to the data, as each neurone has a different weight, this will have a different effect.



# Activation Functions

## ReLU Layer<sup>a</sup>

<sup>a</sup>Relu Module <http://cs231n.github.io/convolutional-networks/>

- Layer to correct data - take into account the non-linearity of the process of classification.
- One example of Relu function  $f(x) = \max(0, x)$
- By consequences all values less than 0 are set as 0.

## PyTorch coding

- Just define a layer : `relu1=nn.ReLU(inplace=False)`
- Add a module with this layer : `model.add_module('Relu1', relu1)`
- It is authorized to define and add module at the same time.
- Never use the ReLU function in the last layer. It is only for the intermediate layers.

# Activation Functions

## SoftMax

- Transform a vector of reals in a vector of probability.
- Use a score vector as inputs.
- Often used in the final layer of classifier model.

## PyTorch coding

- calling Softmax :

```
y=torch.softmax(x,dim=0)
// dim=-1 if your input is a vector.
```

# Using a Deep Network

## Data organisation

- Training and Validation data set are rebuild at each epoch.
- Don't miss to shuffle at each epoch.
- Usual parameters are 80% for training and 20% for validation.
- Test data has to be kept apart and never seen before the test step.

## Re-using a model

- Saving the model to apply to test data or to fine-tune another CNN.
- Transfert learning or fine-tuning: re-use a training model to feed another one for another task - and another data set.

# CNN Architecture

## AlexNet

- Created in 2010 for the ILSVRC challenge, won in 2012 with an error-rate of 15.3%.
- Original paper : ImageNet Classification with Deep Convolutional Neural Networks by Alex Krizhevsky et al.
- Imagenet dataset : 22 000 classes and 15 millions of images.
- 8 Layers - including Dropout layer. producing 60 millions of parameters.
- Replace  $\tanh$  function by Rectified Linear Units (ReLU), fastest computing.
- Able to use multiple GPU.
- Special pooling with overlap.
- Data augmentation : 2048 times.

# AlexNet Architecture

## AlexNet

- Conv. layer 1: input  $224 \times 224 \times 3$  image tensor, 96( $11 \times 11$ ) kernels, stride 4 and padding 2. Produced a  $55 \times 55 \times 96$  output tensor to pass to ReLU activation function - This layer contains 34,944 parameters to train.
- Max-Pooling layer 2:  $55 \times 55 \times 96$  tensor as input. Perform a zero-padding with a kernel  $3 \times 3$  and a stride 2. Produce a  $27 \times 27 \times 96$  output tensor.
- Conv. Layer 3: convolution layer input  $27 \times 27 \times 96$ , apply a convolutional operation 256 ( $5 \times 5$ ) kernels with a stride 1 and a padding 2. Produce  $27 \times 27 \times 256$  output tensor, passed to ReLU function. Contain 614,656 parameters to train.
- Max-Pooling layer 4:  $27 \times 27 \times 256$  input tensor, zero padded operation and  $3 \times 3$  kernel with a stride 2. Produce a  $13 \times 13 \times 256$  output tensor to pass to ReLU function.

# AlexNet Architecture

## AlexNet

- Conv. Layer 5: receive a  $13 \times 13 \times 256$  input tensor. Conv operation uses  $384(3 \times 3)$  kernels with stride and padding 1. Produce  $13 \times 13 \times 384$  output tensor to feed the ReLU function. Contain 885,120 parameters.
- Conv. Layer 6: perform the same operation than layer 5 with same kernel. Contain 1.327.488 parameters.
- Conv. Layer 7:  $13 \times 13 \times 384$  input tensor, convolutional operation with a  $256(3 \times 3)$  kernels with stride and padding 1. Produce a  $13 \times 13 \times 256$  output tensor to feed the ReLU function. Contain 884,992 parameters.

# AlexNet Architecture

## AlexNet

- Max-Pooling layer 8: receive a  $13 \times 13 \times 256$  input tensor. Perform a zero-padding, apply a  $3 \times 3$  window with a stride 2. Produce a  $6 \times 6 \times 256$  output tensor to pass to the ReLU function.
- Fully-Connected layer 9: accept the  $6 \times 6 \times 256$  input tensor. Perform a weighted sum operation including a bias term. Produce a  $456 \times 1$  output tensor to pass to the ReLU function. Contain 37,752,832 parameters.
- Fully-connected layer 10: accept the  $4096 \times 1$  input tensor, perform the same operation than layer 9 and produce the same size output tensor.
- Fully-connected layer 11: accept the input tensor from the previous layer, perform the same operation and produce a  $1000 \times 1$  output tensor, i.e. number of waited classes, to pass to the softmax activation function.

# AlexNet Architecture

## AlexNet

- For ImageNet experiment, the other parameters values are:
  - Batch size 128
  - Momentum 0.9
  - Learning rate 0.0005
  - Dropout has been added in the first two fully-connected layer.
- A graphic card with 2 GPU has been used.



# CNN Architecture

## VGG Family

- Created in 2014 for the ILSVRC challenge.
- VGG16: 16 convolutional layers. 138 millions of parameters.
- 13 convolutional layers, Reduce the size filter to 3X3.
- 3 fully connected layers.
- Original paper : Very Deep Convolutional Networks for Large Scale Image Recognition by Karen Simonyan et al.

# CNN Architecture

Full description at :

<https://neurohive.io/en/popular-networks/vgg16/>

