Deep Learning Techniques for Image Recognition

Catarina NS Silva, Ricardo Cardoso Pereira
University of Coimbra

Funded by the European Union
Horizon Europe GA no. 101135583

Agenda

- Computer Vision
- Deep learning
- CNNs - Convolutional neural networks
- Transfer learning
- ViTs - Vision Transformers
- Practice - Colab

Computer vision tasks

Object detection

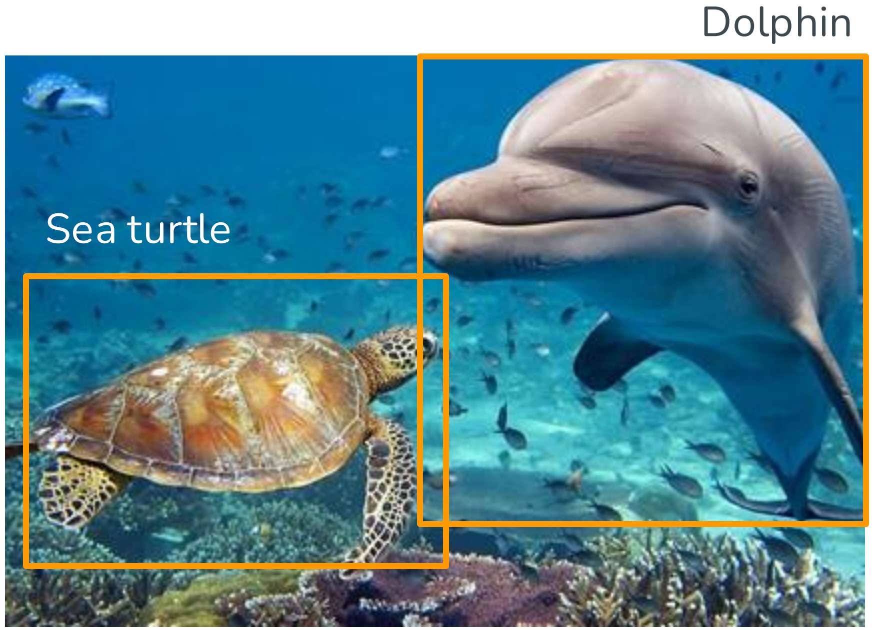


Image Classification

Dolphin

Sea turtle

Semantic Gap

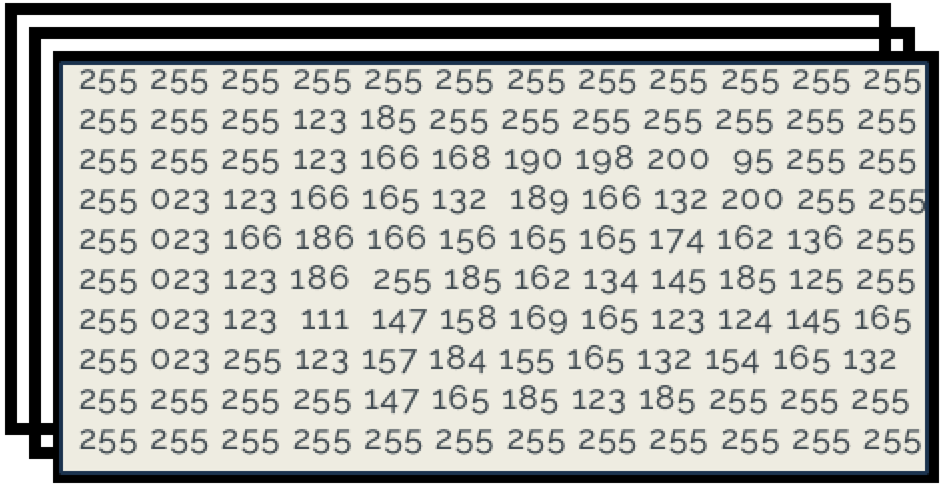
Semantic Gap: the difference between how humans and computers understand and interpret information

Cat
Fish
Bird
Dog

Semantic Gap

Semantic Gap: the difference between how humans and computers understand and interpret information

What the computer sees:
3D grid
RGB(255,0,0) = red
RGB(0,255,0) = green
RGB(0,0,255) = blue



Artificial deep neural network architecture

ImageNet competition

2012

AlexNet

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

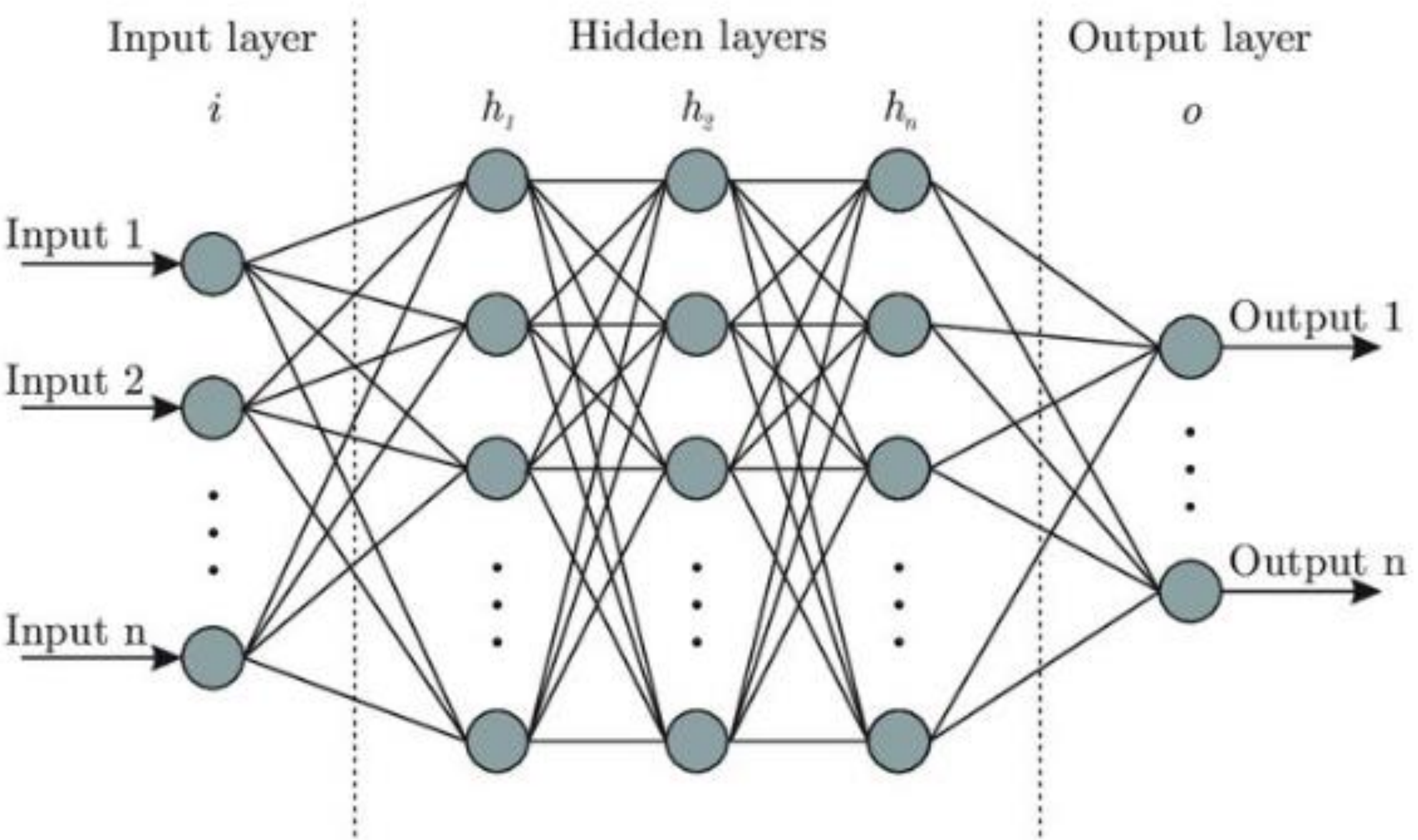
Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Abstract

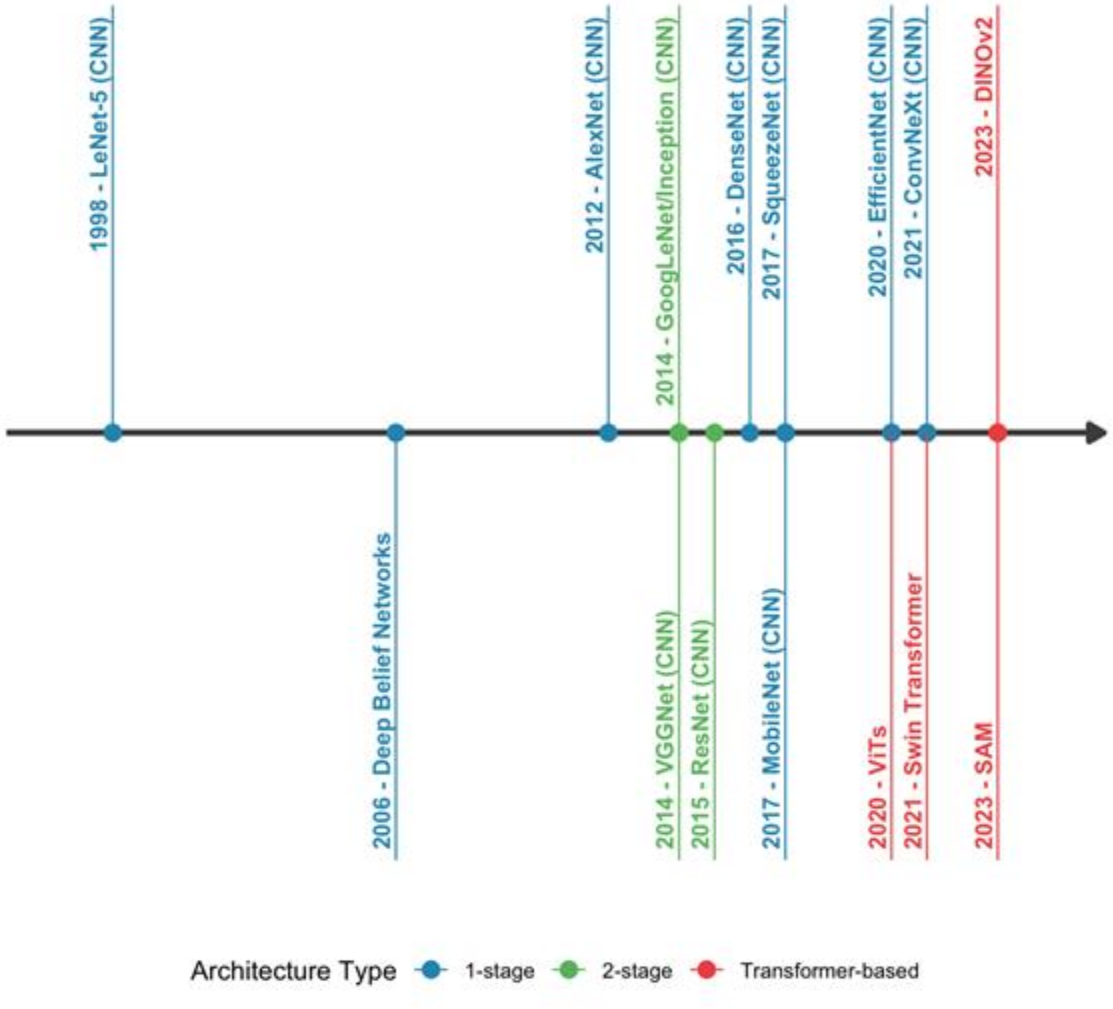
We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

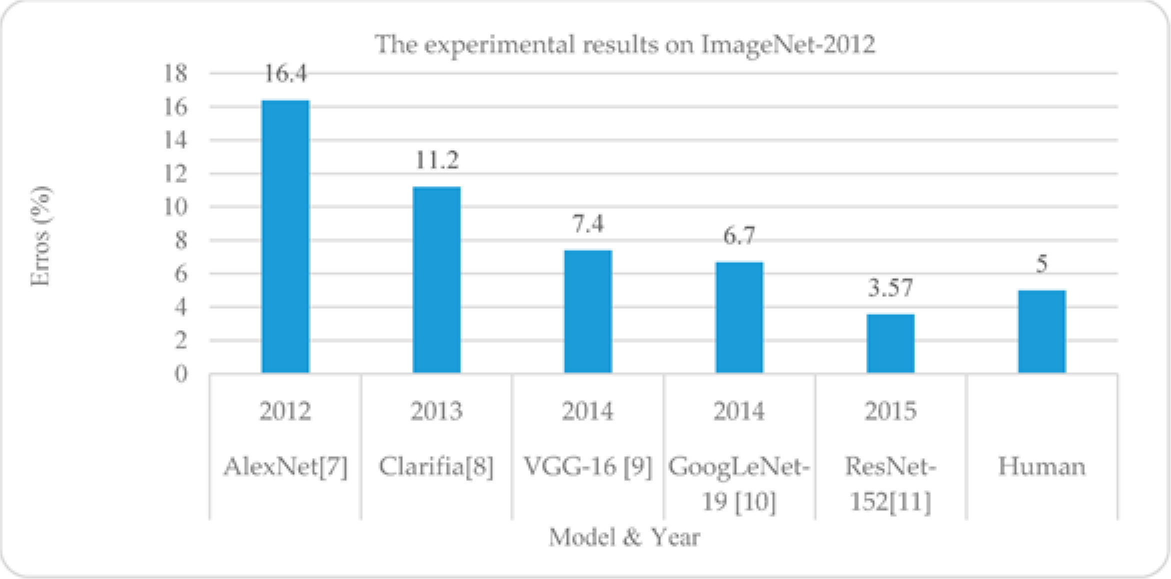
Artificial deep neural network architecture

A system that can process and reason about data using hierarchical learning algorithms, with many "layers" that are very loosely inspired by how the brain works



Artificial deep neural networks

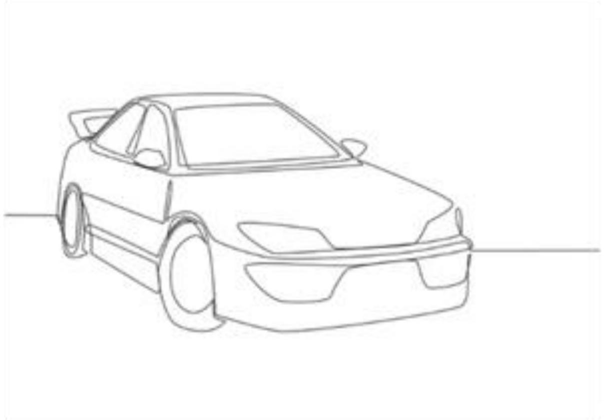




Alom *et al* 2019

Concepts in AI

Architecture



Overall design of the vehicle

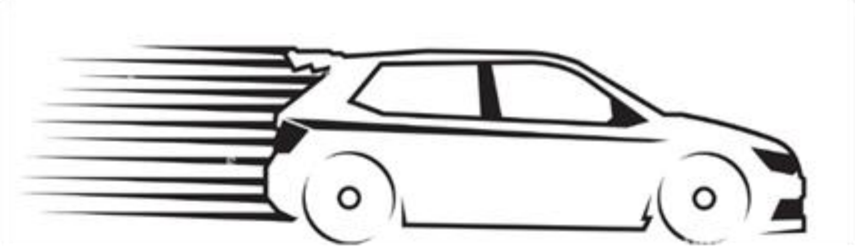
Overall design or general framework of the AI system

Algorithm

Set of rules that govern how the vehicle operates

Set of instructions that the system follows to perform a specific task

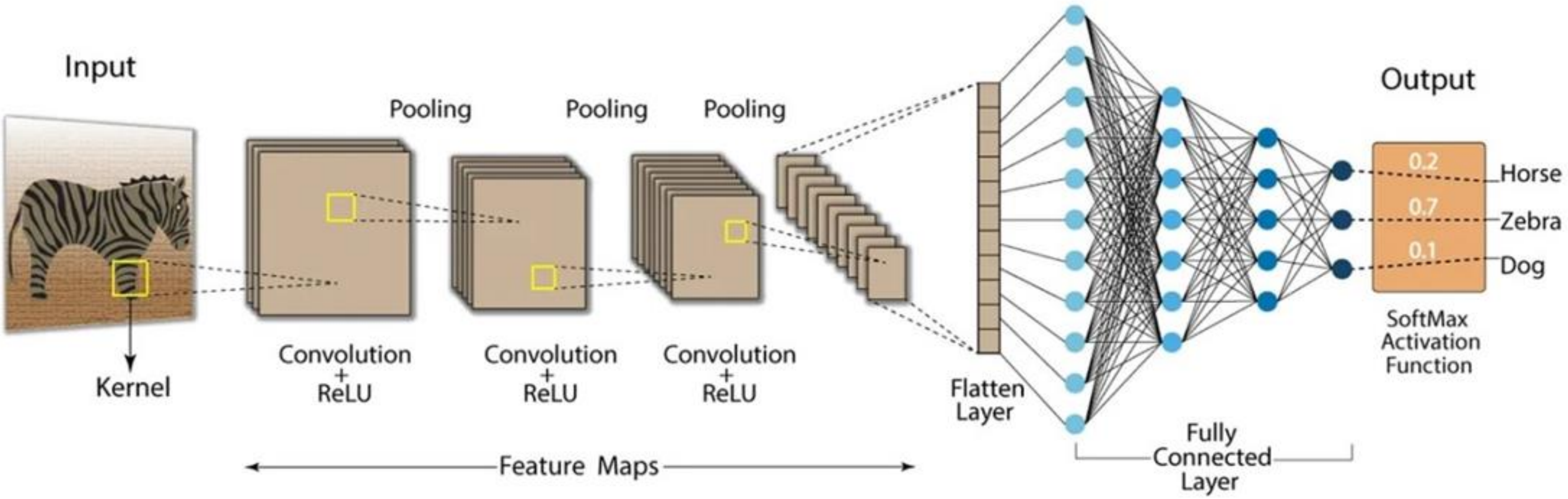
Model



Specific implementation of the design and rules that have been built and tested

Specific implementation of the algorithm trained on data

Convolutional neural networks - CNNs Architecture



Convolution

Convolution: the process of transforming an image by applying a kernel (or filter) over each pixel and its local neighbors across the entire image

Important for recognizing (and enhancing) edges, shapes and patterns in the images

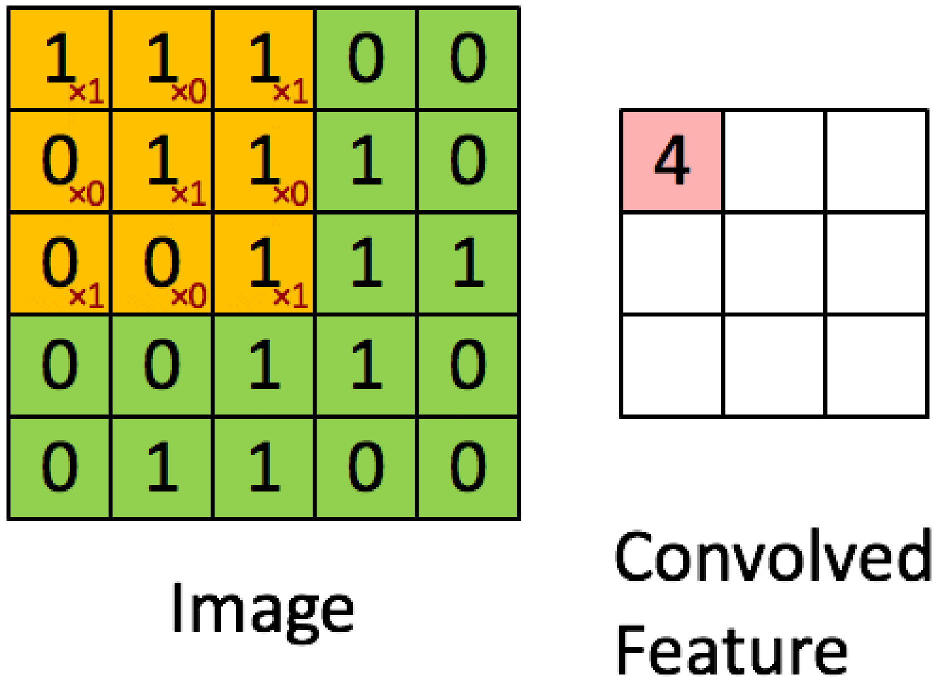


Image Kernels explained visually

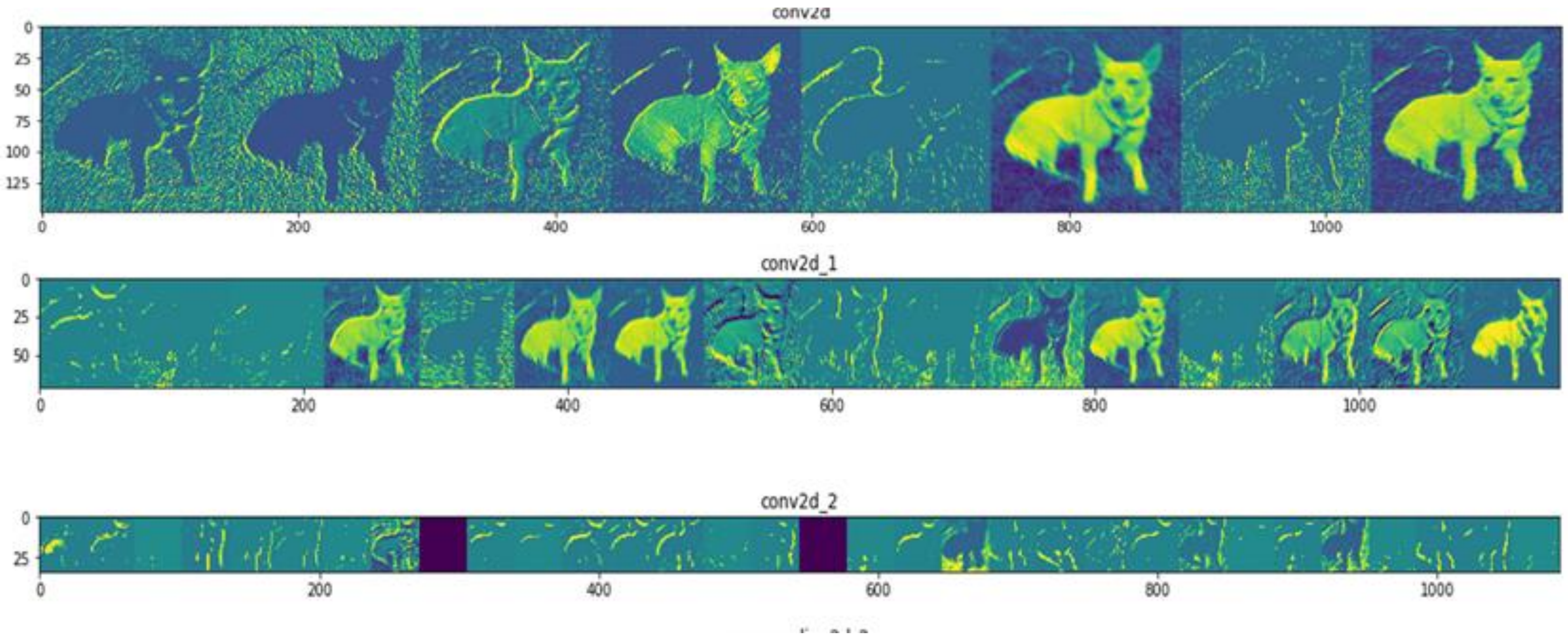
Feature Maps

2D array or grid of numbers resulting from the application of convolutional filters (or kernels) to an input image or a previous layer´s feature map

Low-level features (early layers): edges, corners, textures.

Mid-level (middle layers): shapes, contours, parts of objects.

High-level (deep layers): complex patterns like faces, animals, or abstract concepts depending on the task.



Stride

Stride: the number of pixels the filter (also called a kernel) moves across the input image during the convolution operation

Stride=1 means the filter moves one pixel at a time

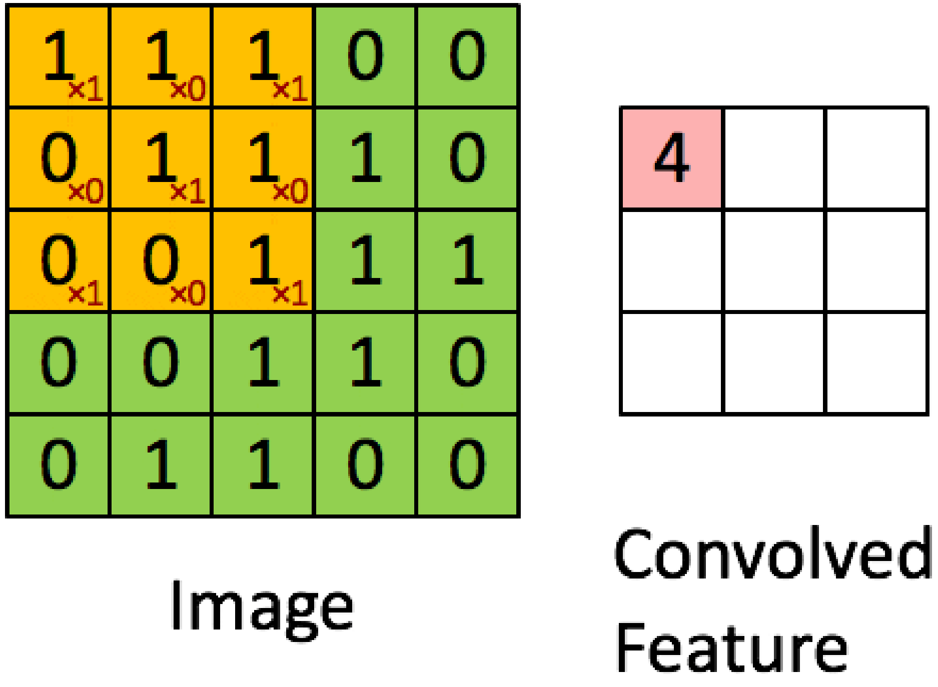


Image Kernels explained visually

REDUCE

Padding

Technique used to preserve the spatial dimensions (border information) of the input image after convolution operations

0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

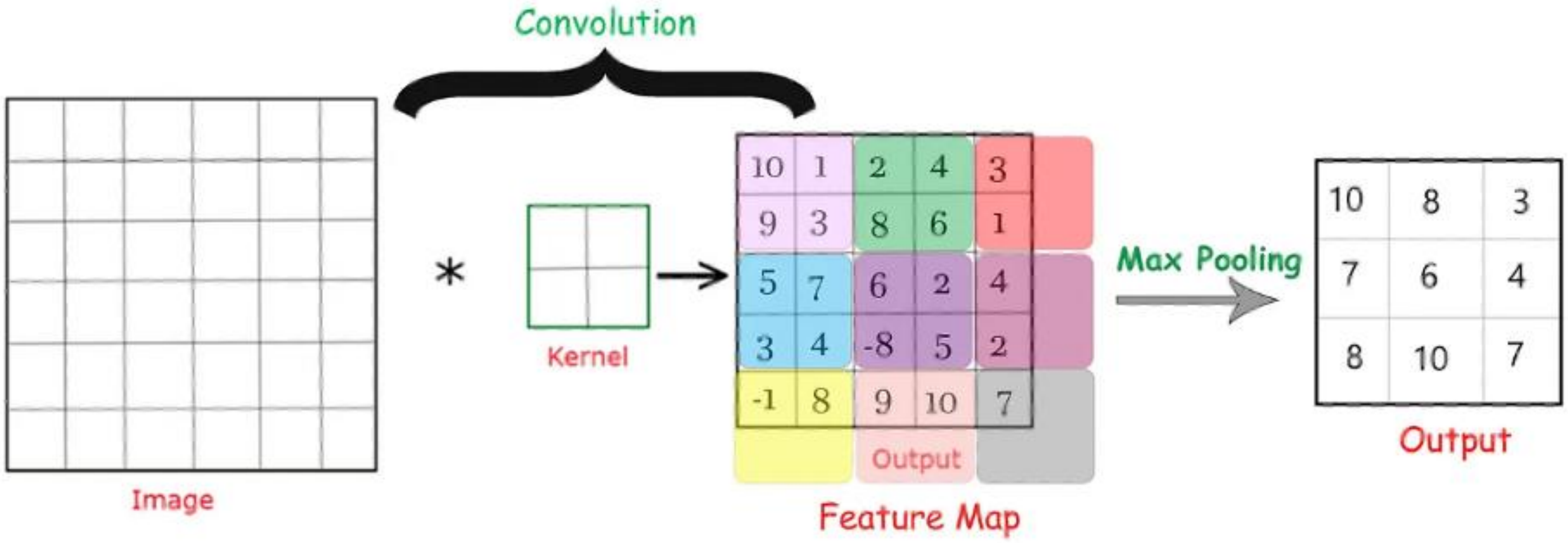
Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

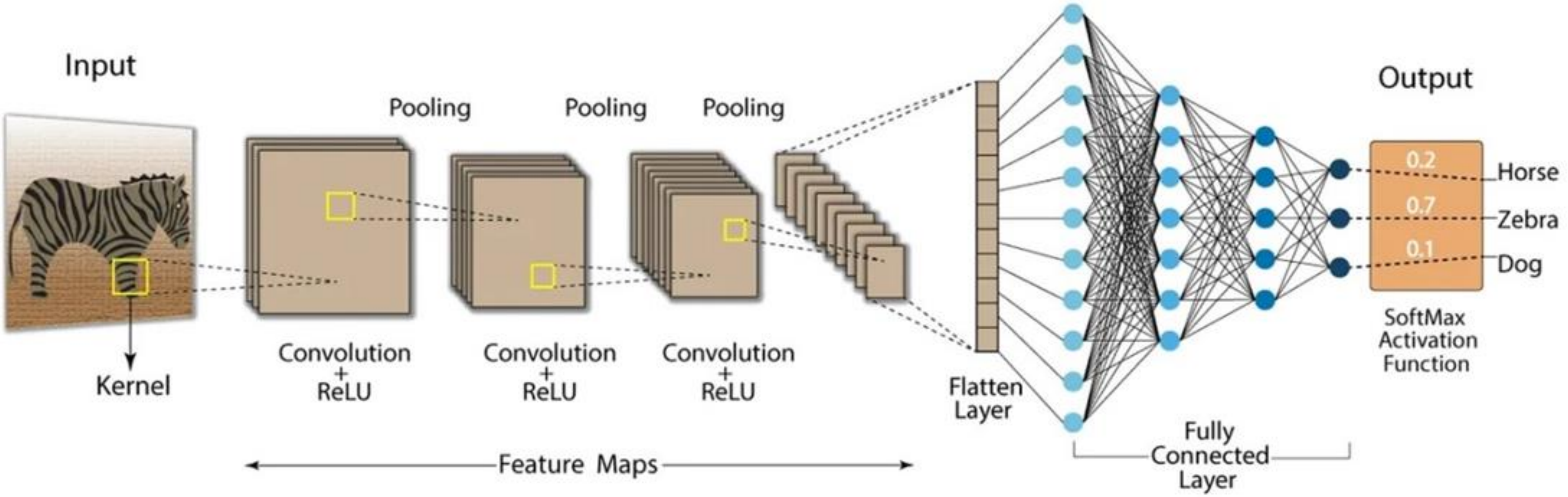
Max pooling

A downsampling technique that reduces the spatial dimensions of feature maps
Downscale the image to extract the most important features (usually, the maximum value from the feature map)



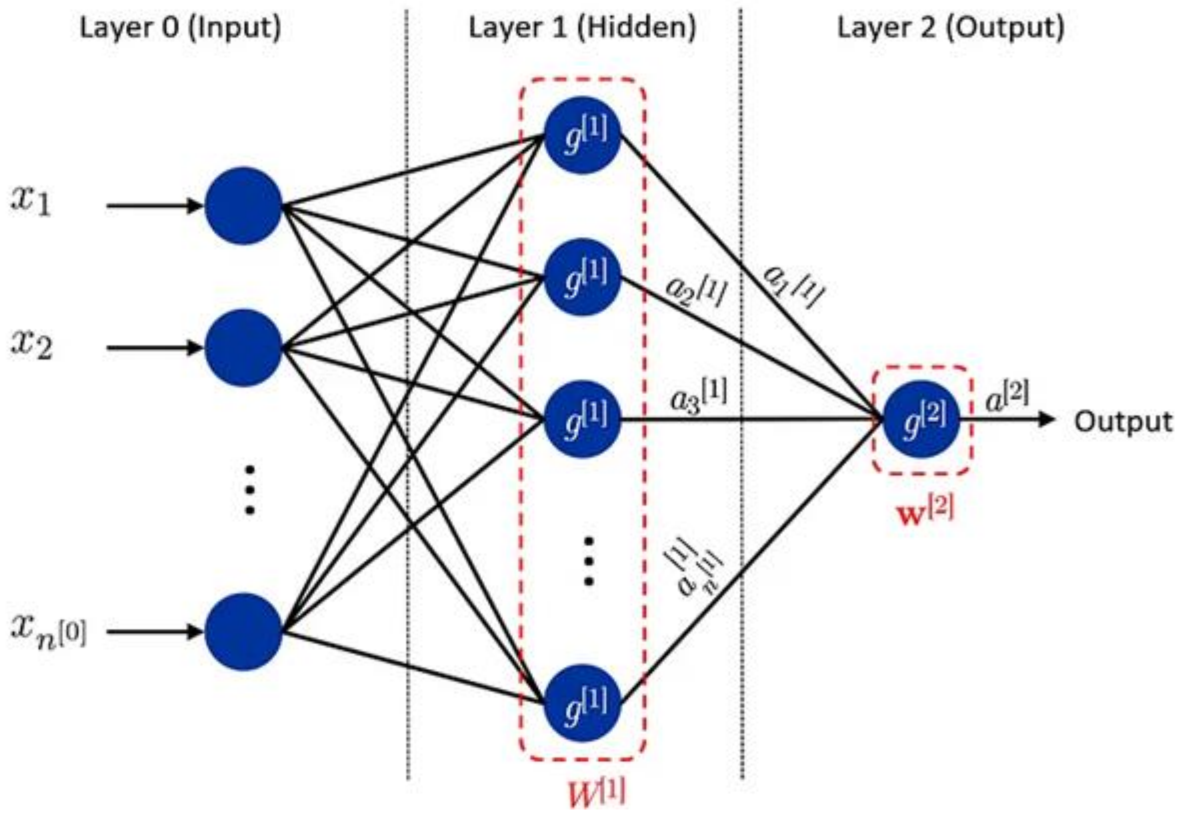
@Codicals

Flattening: convert all the resultant 2-Dimensional arrays from pooled feature maps into a single continuous linear vector



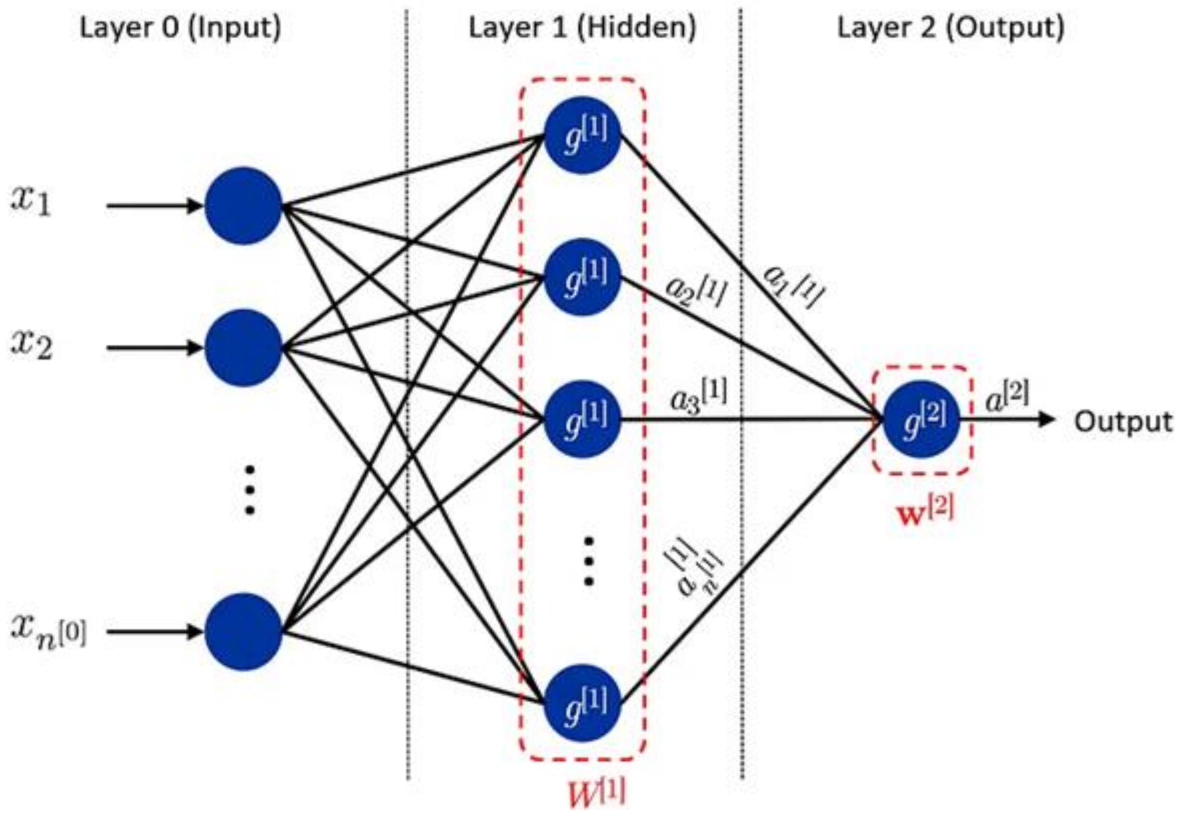
Deep neural networks - Fully Connected layers

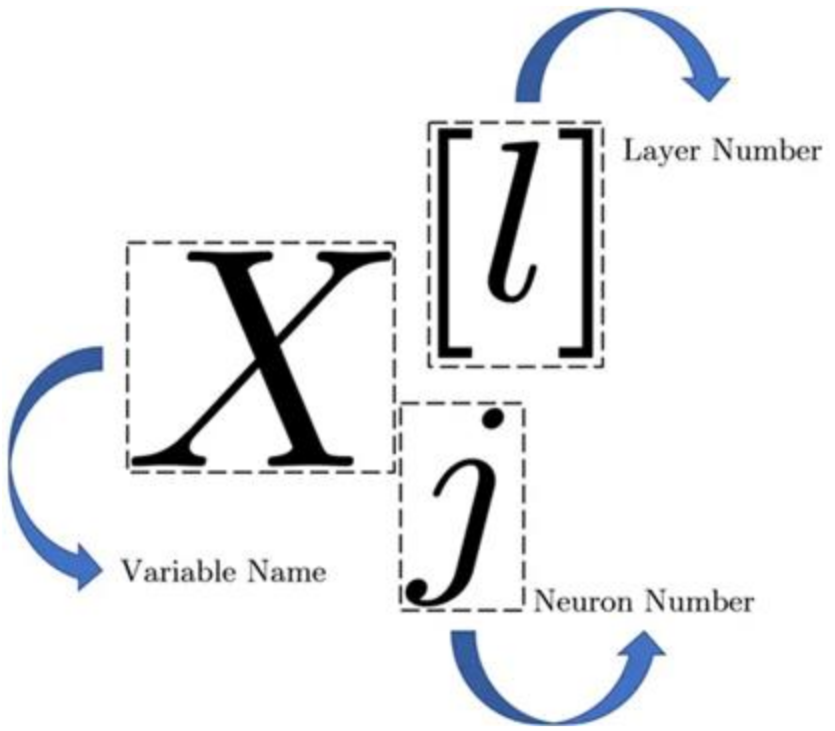
Each neuron in a layer is connected to every neuron in the previous layer or receives input from it



Deep neural networks - Fully Connected layers

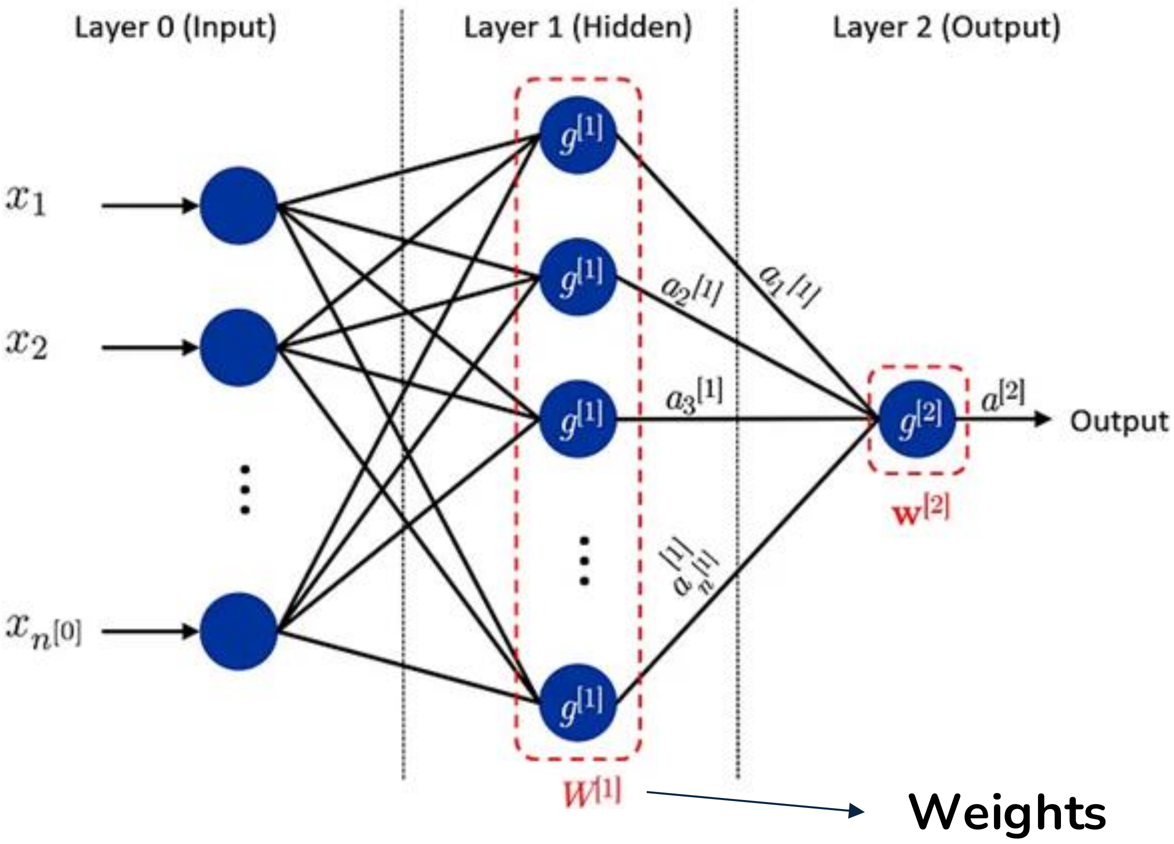
Each neuron in a layer is connected to every neuron in the previous layer or receives input from it





Deep neural networks - Fully Connected layers

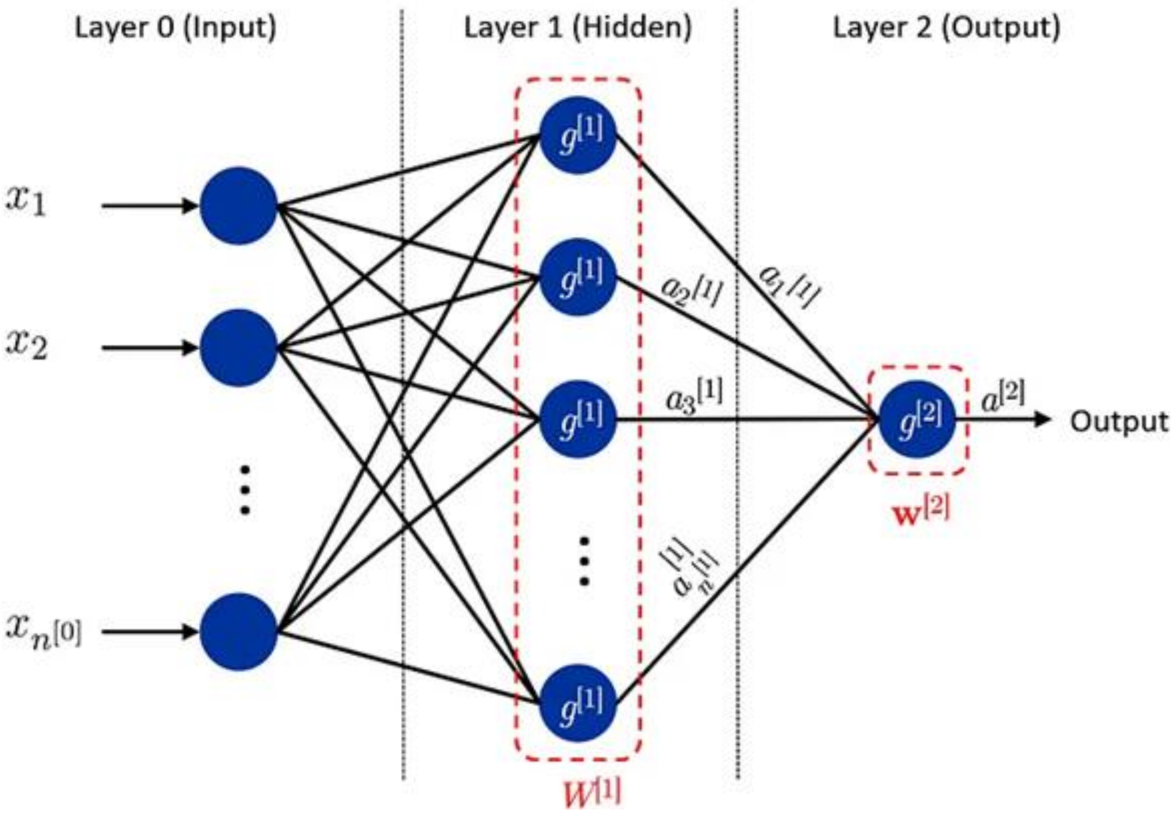
Weights: Determine the strength and direction of the influence one neuron has on another.



Initially, weights are randomly initialized, and then they are updated during training using e.g. backpropagation.

Deep neural networks - Fully Connected layers

Training is iterative and goes forward and backwards in the network!



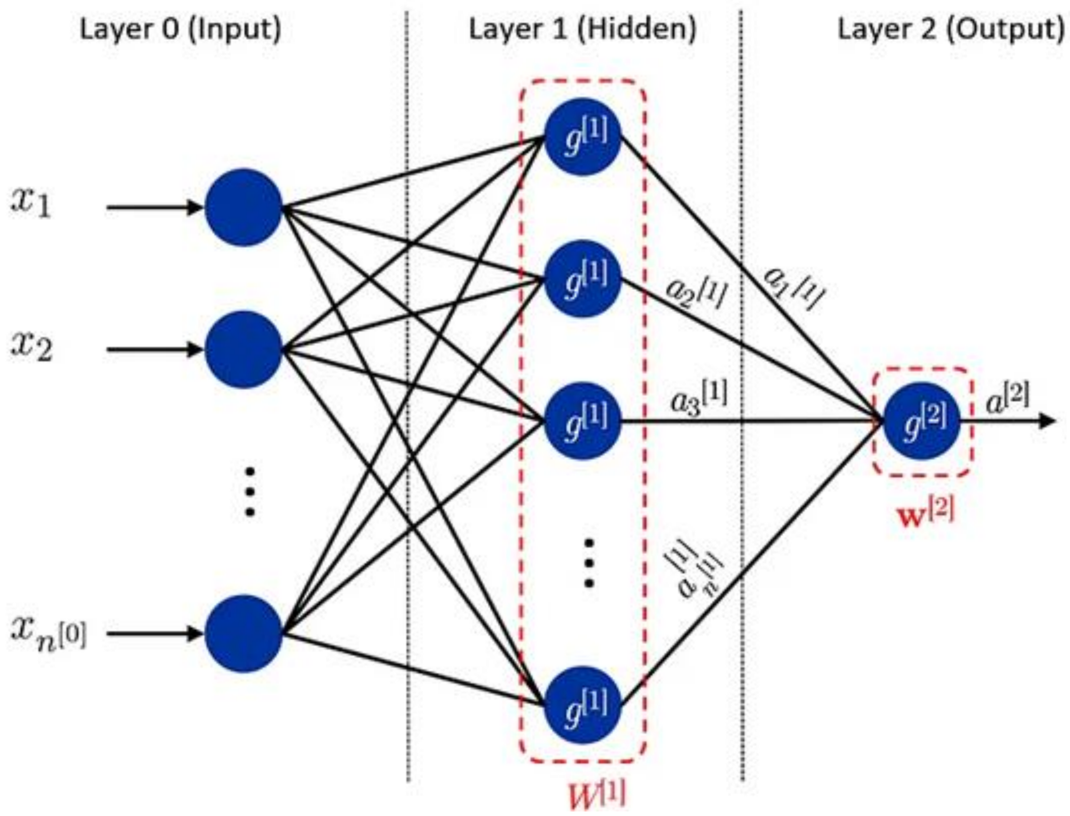
It uses its current weights to make a guess (forward pass).

The loss function checks how wrong prediction was.

Backpropagation “tells” each weight how to adjust so the network improves itself.

Loss function

Quantify the difference between a CNN's predictions and the actual data, guiding the model's learning process, specifically at the end of each epoch.



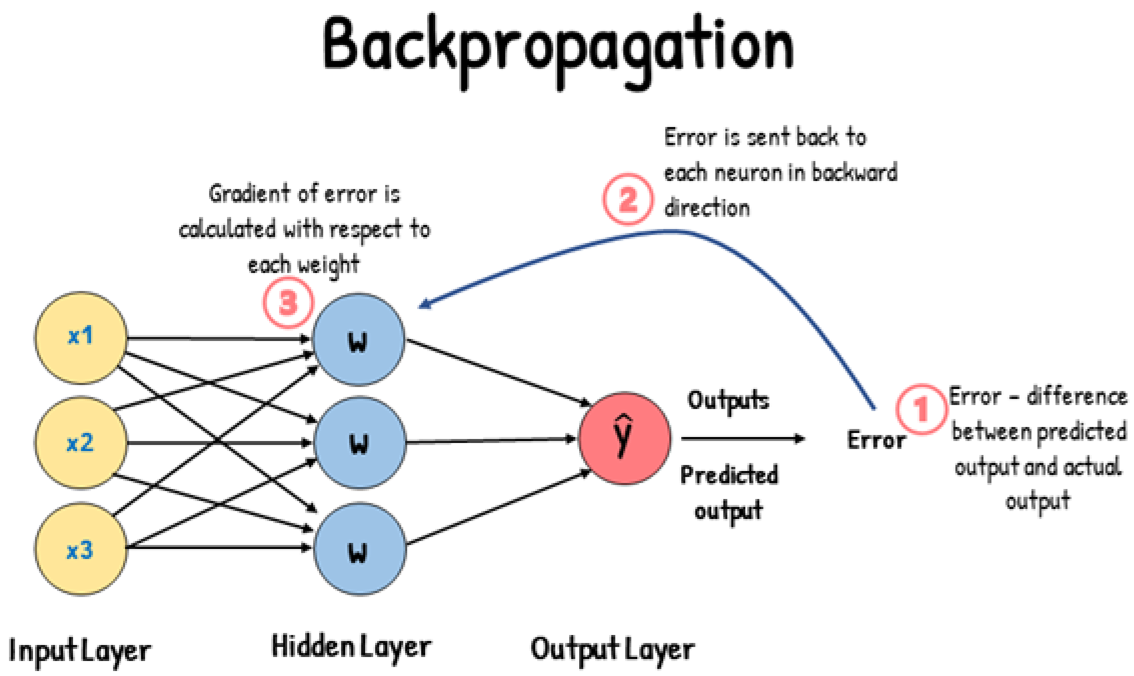
Predicted Output (Y_pred)

True Output (Y)

High loss value = model performing poorly

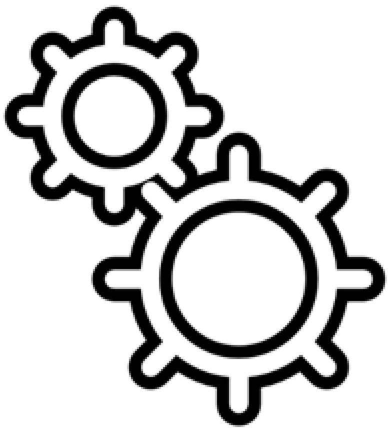
Backpropagation

Loss value is used to calculate the gradients of the loss function with respect to the CNN's weights. Allows the network to adjust its weights to minimize the loss in the following epoch.



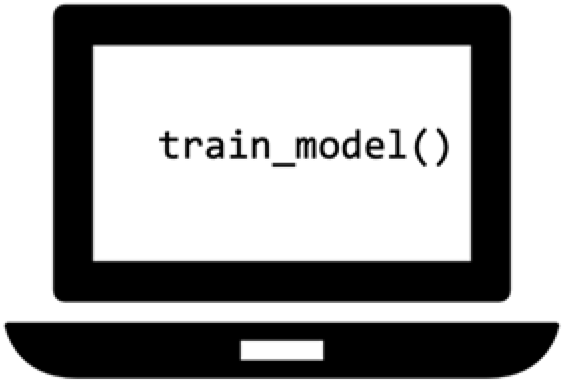
More concepts...

Hyperparameters



Defined before training (e.g. loss functions, epochs)

Model parameters



Defined during training (e.g. weights)

Model performance

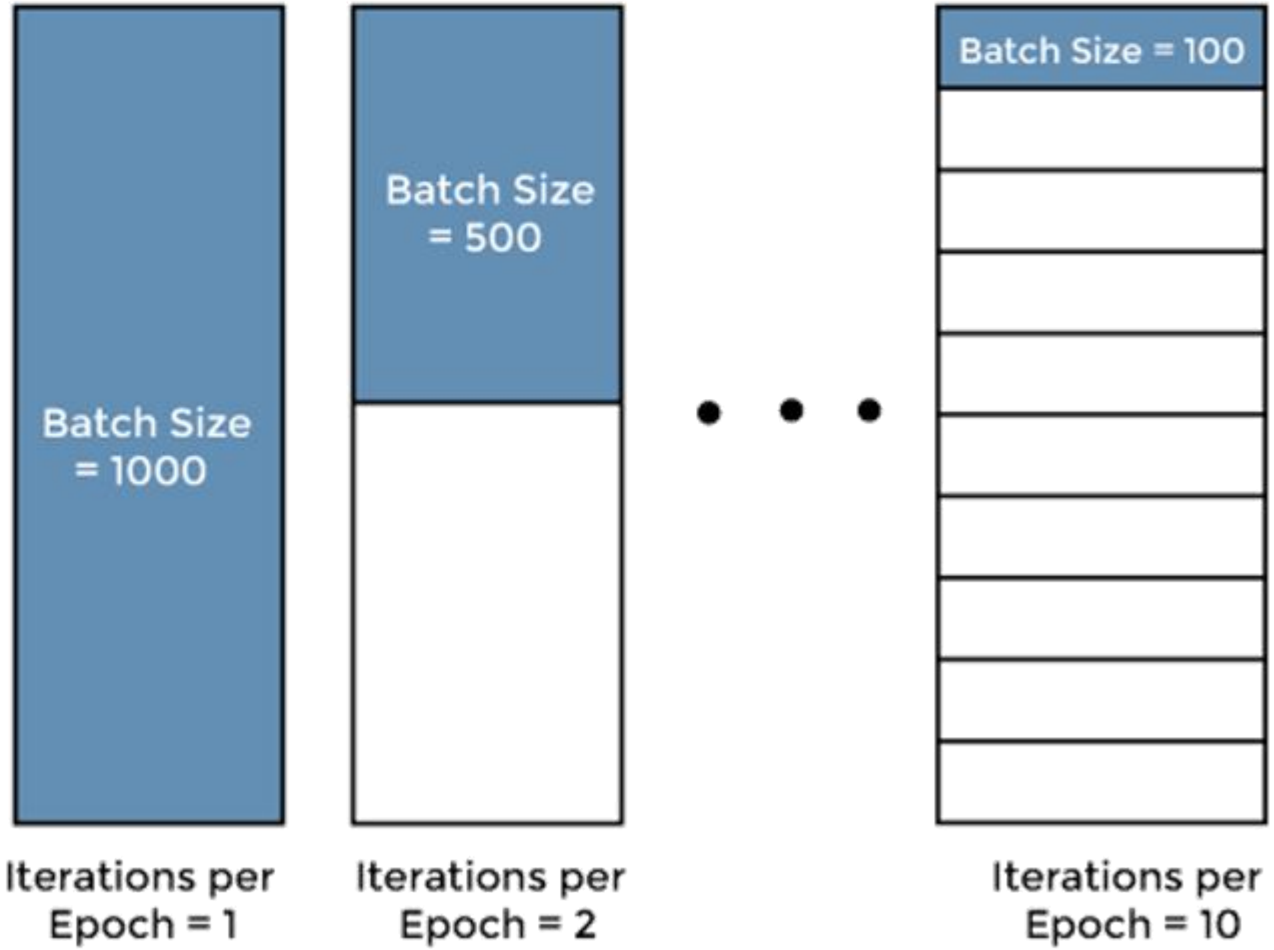
After training (e.g. accuracy)

Hyperparameters

Epochs: complete pass of the training dataset through the algorithm

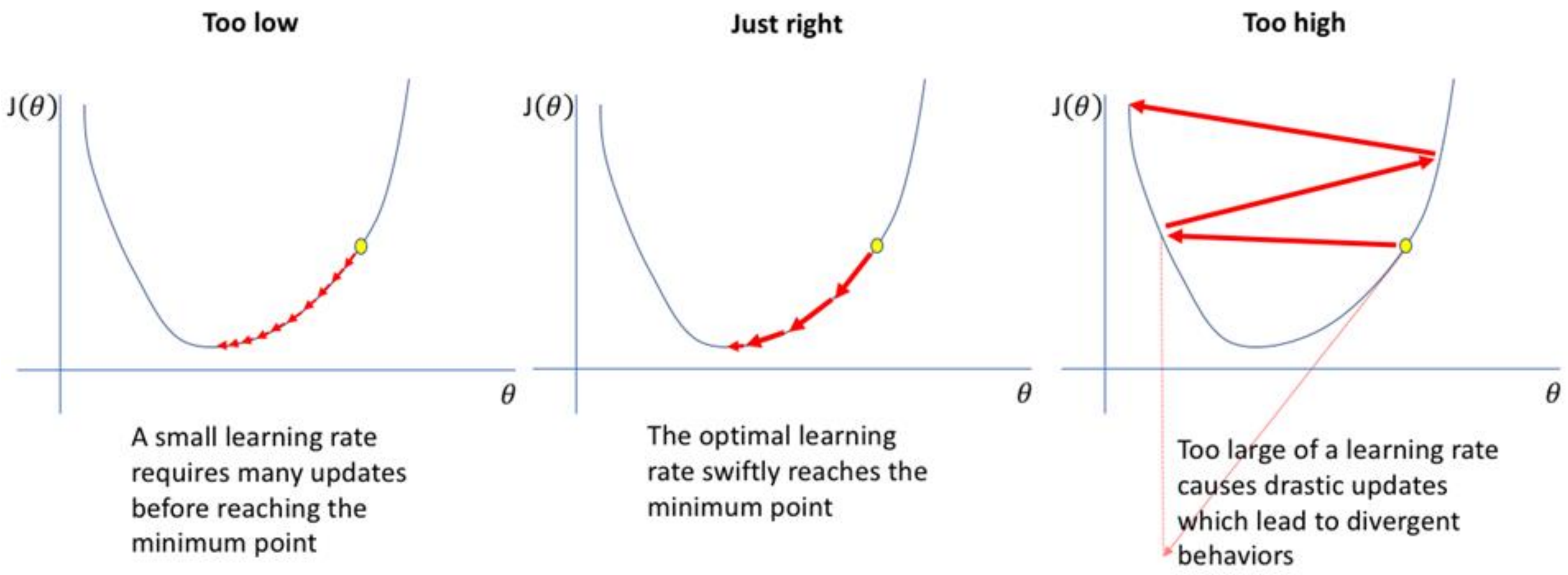
Batch size: total number of training examples (photos, in our case) present in a single batch

Iterations: number of batches required to complete one epoch



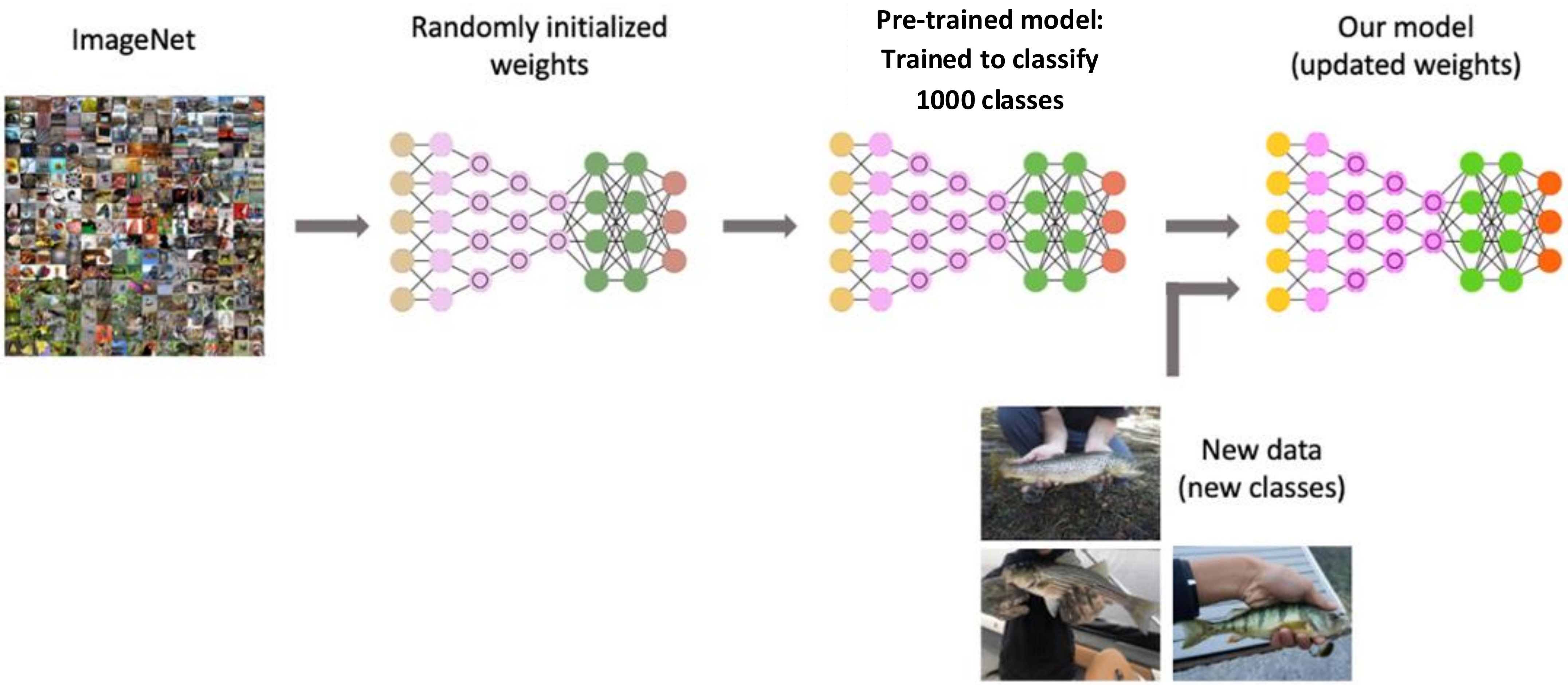
Learning rate

Hyperparameter that determines how much weights are adjusted during each training step

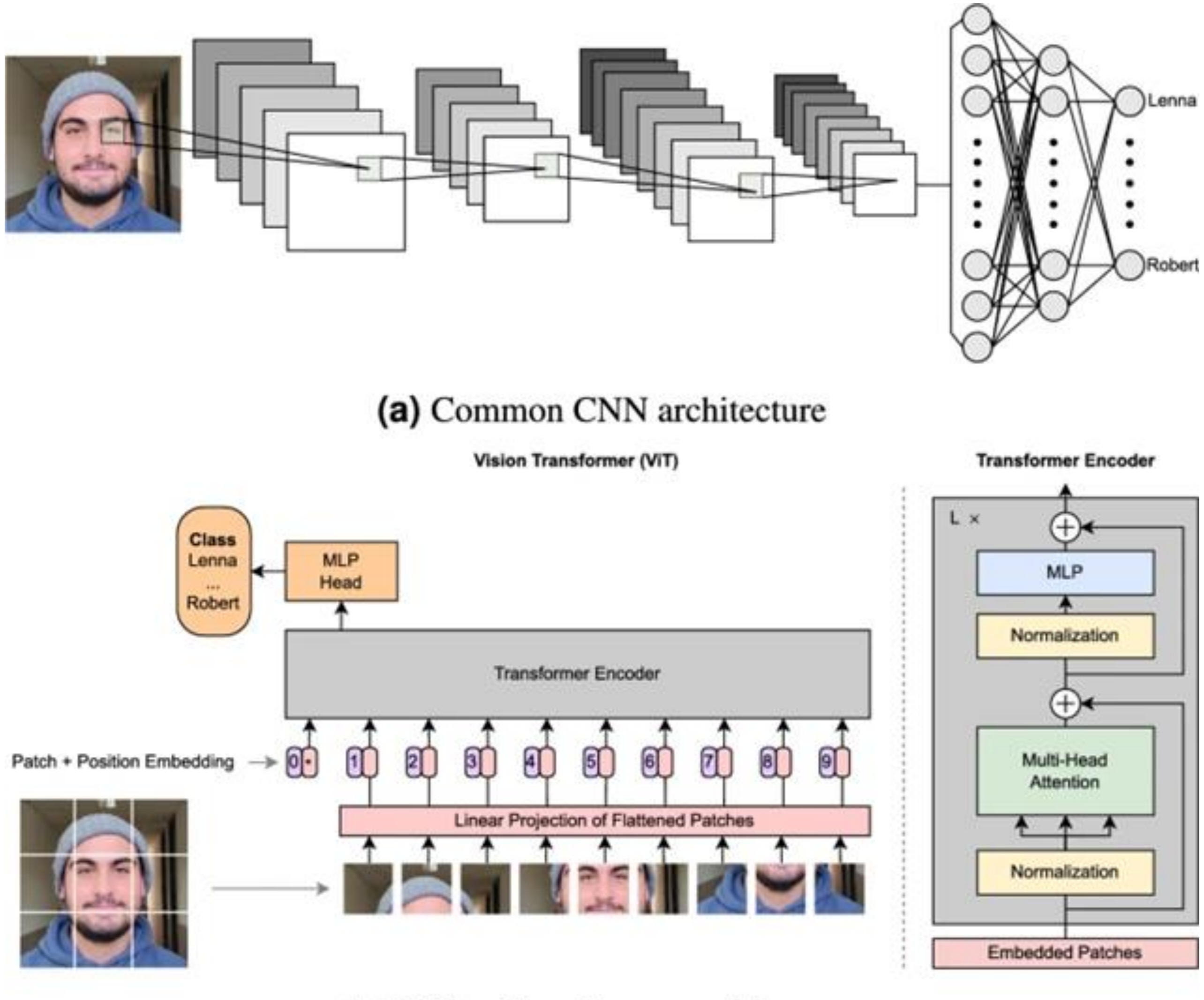


How fast to climb down the “hill” of the loss function?

Transfer learning



Vision Transformers



(a) Common CNN architecture

(b) Vision Transformer architecture

Good at spotting local features

Convolution-based

Can learn very complex patterns across the whole image, but they need to see a lot of examples to get really good

Attention-based

Rodrigo et al 2024 https://www.nature.com/articles/s41598-024-72254-w https://colab.research.google.com/drive/1Z73mrpk95USSVBChc438-eoOXgA9JyI8

Other techniques: Augmentation

Artificially increase the size and diversity of a training dataset by creating modified versions of existing data

Original

Horizontal flip

Vertical flip