REGINNA 4.0

Introduction to Convolutional Neural Networks and its application in Computer Vision

Eduardo Fidalgo Fernández
Universidad de León (Spain)
eduardo.fidalgo@unileon.es

Supported by eit Manufacturing

Funded by the European Union

INNOVATION CAPACITY BUILDING FOR HIGHER EDUCATION — PART OF EIT HEI Initiative — eit-hei.eu

www.reginna4-0.eu

Credits

- Chollet, F. (2022). *Deep learning with python.* Manning publications
- Fei-Fei, L. (2021). *CNN Architectures*. Lecture 9. Stanford University
- Amini, A. (2020). *Introduction to Deep Learning*. Lecture 3 - 6.S191 - from MIT.
- Convolutional Neural Networks for Visual Recognition (Stanford University): http://cs231n.stanford.edu/

universidad de león

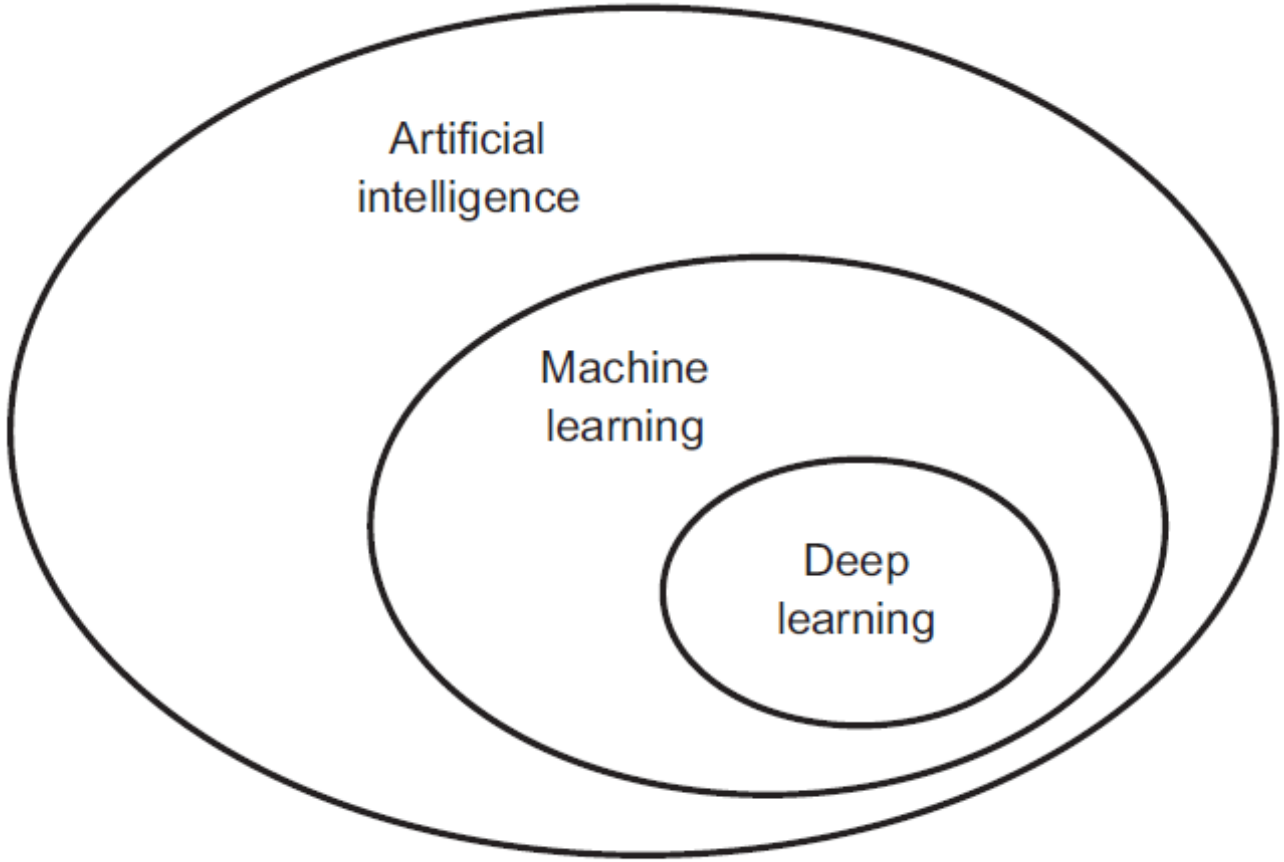
REGINNA 4.0

Table of contents

universidad de león

Artificial Intelligence

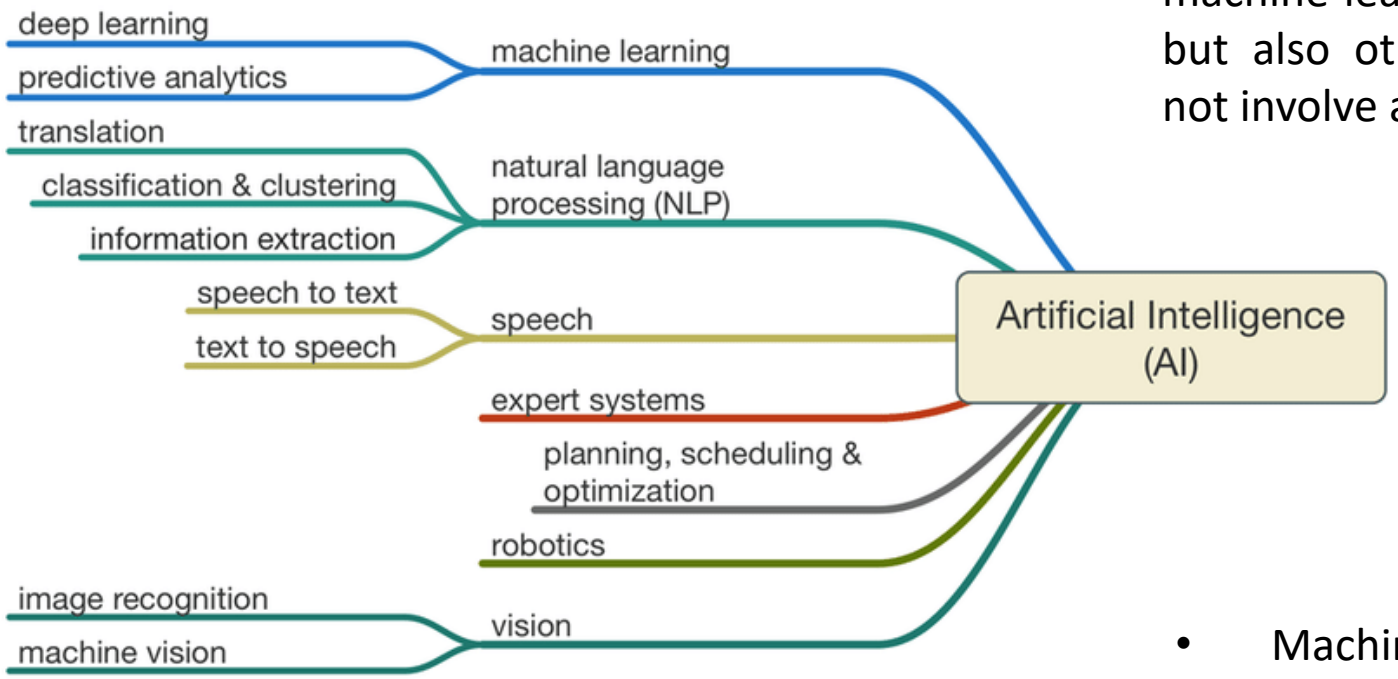
The effort to automate intellectual tasks normally performed by humans



universidad de león

REGINNA 4.0

Artificial Intelligence



AI is a general field that includes machine learning and deep learning, but also other approaches that do not involve any learning at all

- Machine Learning
- Natural Language Processing
- Knowledge representation
- Automated reasoning
- Computer Vision
- Robotics

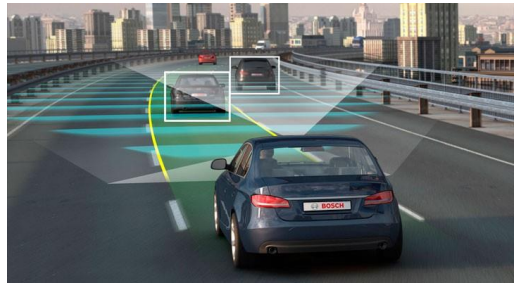
universidad de león

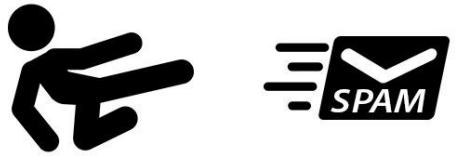
REGINNA 4.0

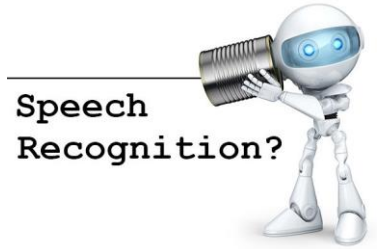
Artificial Intelligence

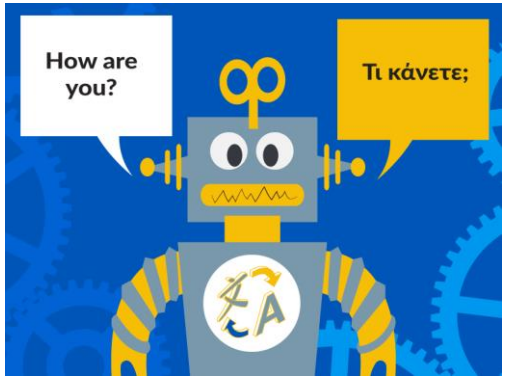
What can AI do? (Difficult question…)

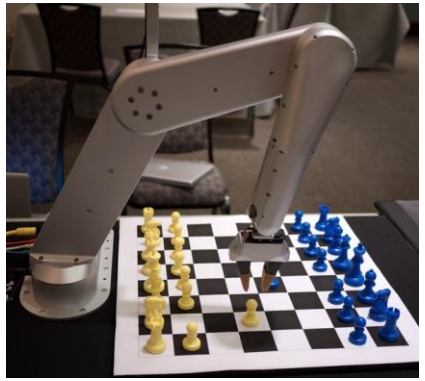
- Robotic vehicles
- Speech recognition
- Autonomous planning and scheduling
- Game playing
- Spam detection
- Logistic planning
- Robotics
- Machine Translation
- Face detection
- Face recognition
- Phishing detection
- …











universidad de león

REGINNA 4.0

https://ai.meta.com/blog/emu-text-to-video-generation-image-editing-research/
https://ai.meta.com/blog/ai-dataset-animation-drawings/
https://blog.research.google/2023/11/metnet-3-state-of-art-neural-weather.html

Artificial Intelligence

What can AI do today? (2023-¿2024?)

- ChatGPT (https://chat.openai.com/)
- BARD (https://bard.google.com/?hl=es)
- LLAMA-2 (https://ai.meta.com/llama/)
- … (…)
- DALL-E 3 (https://cdn.openai.com/papers/dall-e-3.pdf)
- SeamlessM4-T (https://ai.meta.com/blog/seamless-m4t/)
  - Automatic speech recognition for nearly 100 languages
  - Speech-to-text translation for nearly 100 input and output languages
  - Speech-to-speech translation, supporting nearly 100 input languages and 35 (+ English) output languages
  - Text-to-text translation for nearly 100 languages
  - Text-to-speech translation, supporting nearly 100 input languages and 35 (+ English) output languages
- Emu Video (https://bit.ly/3MPaIaM)
- Microsoft (GitHub) Copilot
- MetNet-3
- Animate your kid drawings…
- …

A person is standing at a pizza counter, holding a gigantic quarter the size of a pizza. The cashier, wide-eyed with astonishment, hands over a tiny, quarter-sized pizza in return. The background features various pizza toppings and other customers, all of them equally amazed by the unusual transaction.

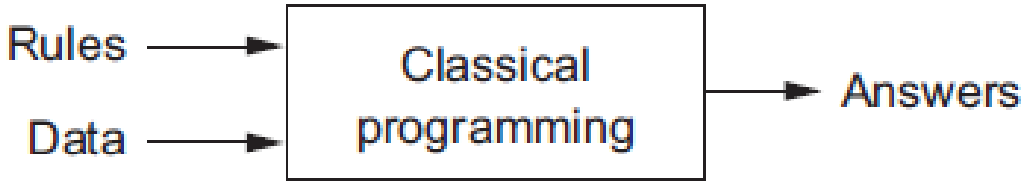
universidad de león

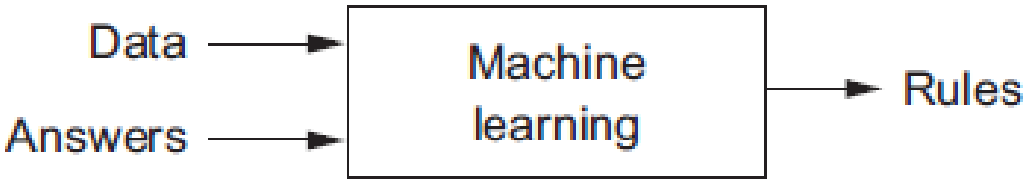
REGINNA 4.0

Machine Learning

Difference between classic AI and Machine Learning

- A machine-learning system is *trained* rather than explicitly programmed





Machine learning –> mapping inputs (e.g., images of inserts) to targets (such as the label “damaged insert”), which is done by observing many examples of inputs and targets.

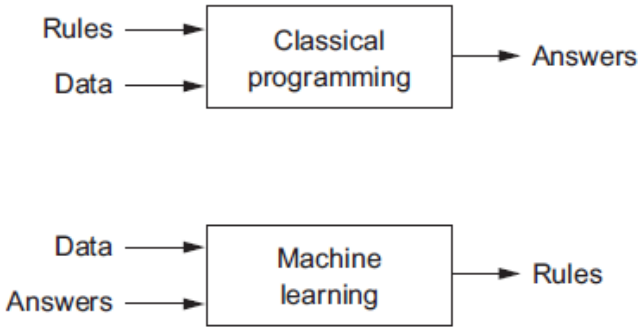
universidad de león

Machine Learning

Difference between classic AI and Machine Learning

- **Machine learning** –> mapping inputs (e.g., images of inserts) to targets (such as the label "damaged insert"), which is done by observing many examples of inputs and targets.

- A machine-learning system is *trained* rather than explicitly programmed



Example of machine learning with an autonomous cars?

- **Data**: images containing (or not) pedestrians

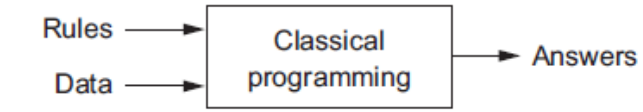
universidad de león

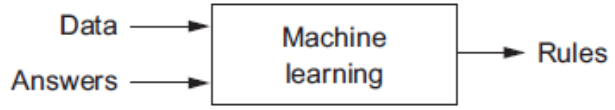
Machine Learning

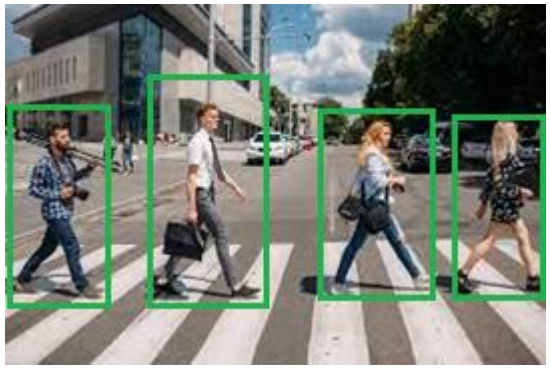
Difference between classic AI and Machine Learning

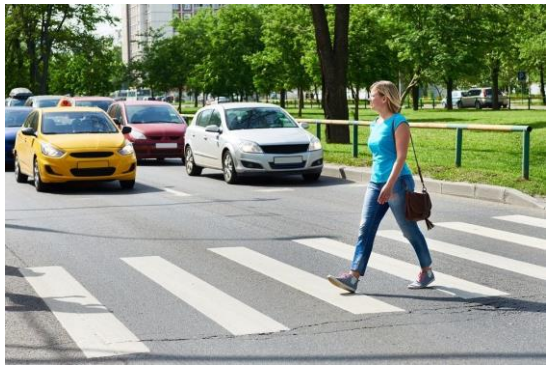
- **Machine learning** –> mapping inputs (e.g., images of inserts) to targets (such as the label "damaged insert"), which is done by observing many examples of inputs and targets.

- A machine-learning system is *trained* rather than explicitly programmed









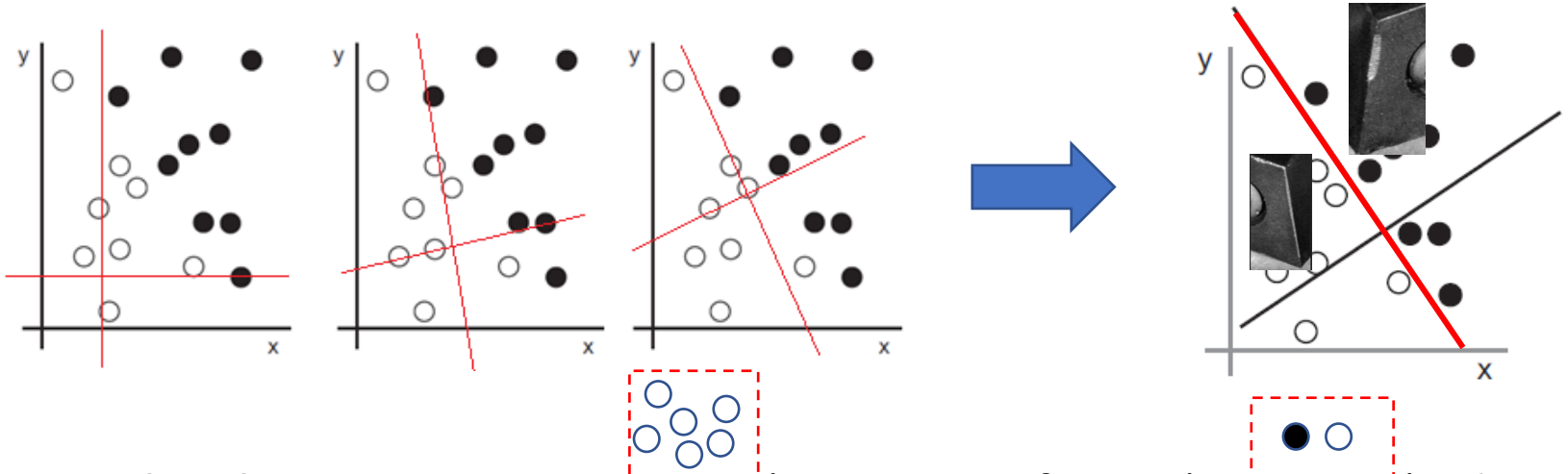
Example of machine learning with an autonomous cars?

- **Data**: images containing (or not) pedestrians
- **Answers**: indications about where are pedestrians
- **Rules? Model for** pedestrian detection?

universidad de león

Machine Learning

Difference between classic AI and Machine Learning



Machine learning –> mapping inputs (e.g., images of inserts) to targets (such as the label "damaged insert"), which is done by observing many examples of inputs and targets.

In machine learning:

- A *category* in a classification problem is called a *class*.
- Data points are called *samples*.
- The class associated with a specific sample is called a *label*.

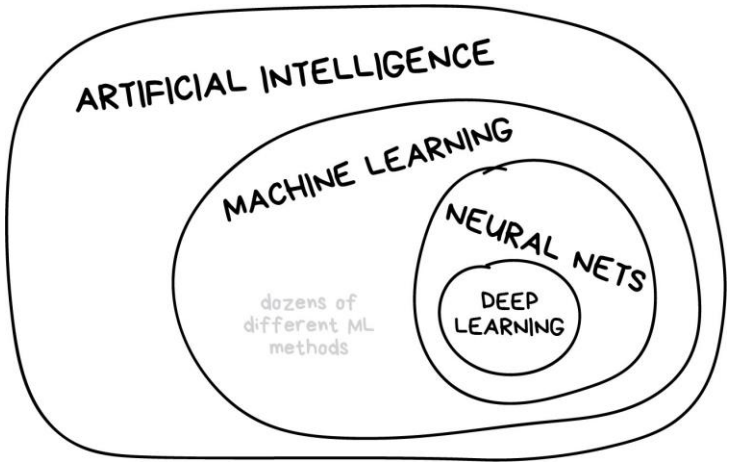
universidad de león

REGINNA 4.0

Deep Learning

Introduction and definition

- **Deep Learning**: **subfield** of Machine Learning, based on **neural networks** with a huge number of nodes, connections and layers. It is based on a **multistage way to learn data representations**.



DEEP: Learning successive *layers* of increasingly meaningful representations. **Depth** of a model: how many layers, learned automatically from exposure to training data

Machine learning –> mapping inputs (e.g., images) to targets (such as the label "cat"), which is done by observing many examples of inputs and targets.

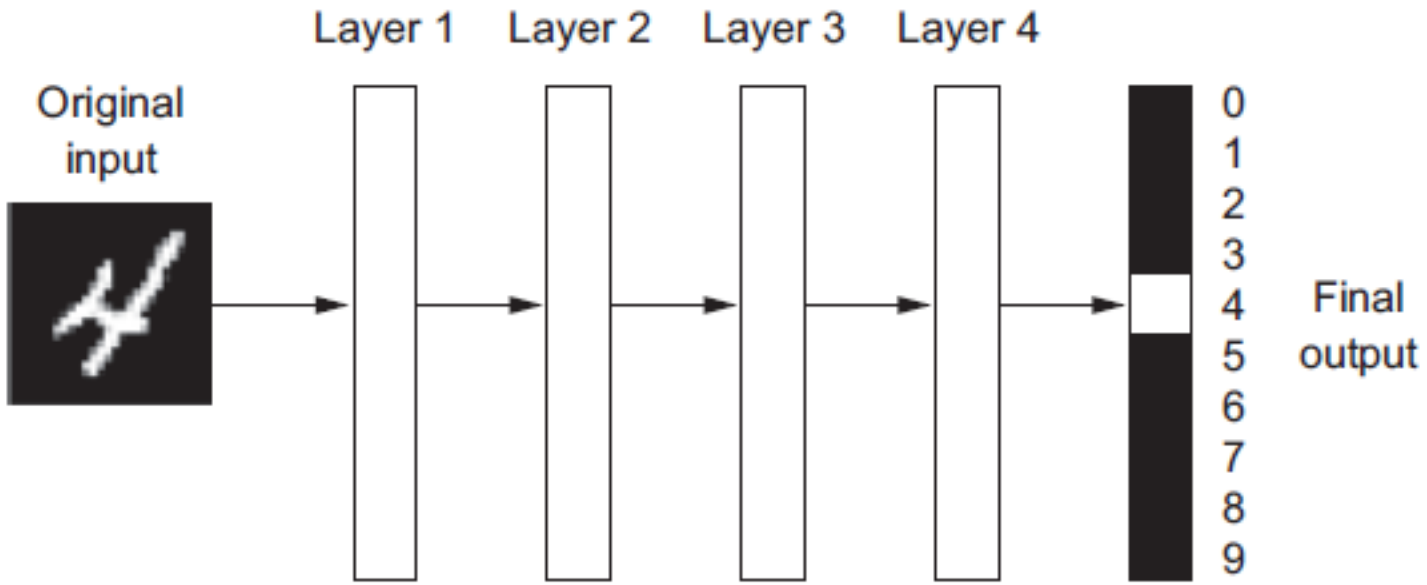
Deep learning –> mapping inputs to targets, via a deep sequence of simple data transformations (layers) which are learned by exposure to examples.

universidad de león

Deep Learning

The deep in Deep Learning

- **How we learn in Deep Learning?** Neural Networks
- Deep Learning models are not models of the brain!
- How it looks like?

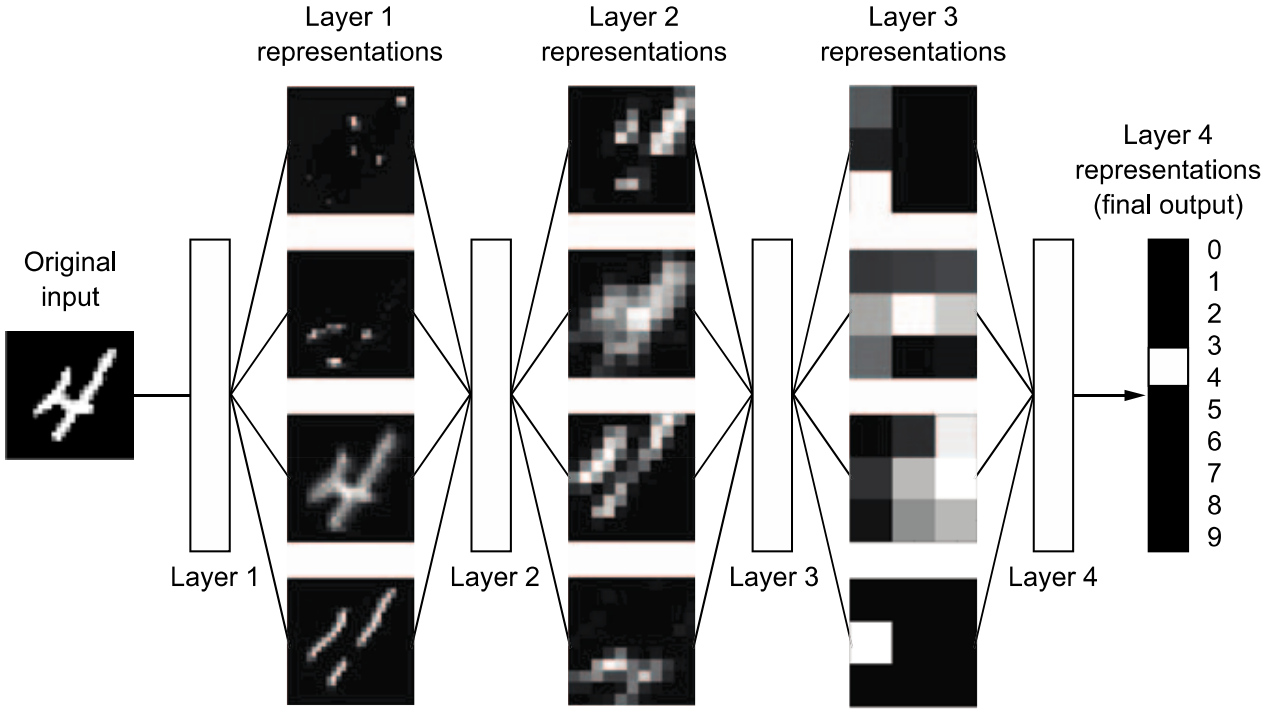


universidad de león

Deep Learning

The deep in Deep Learning

- **How we learn in Deep Learning?** Neural Networks

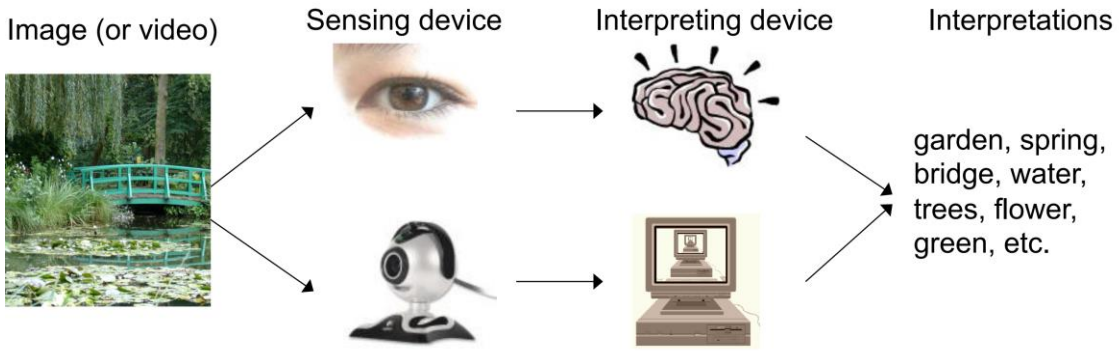


Deep network -> multistage information-distillation operation.

Information goes through successive filters and comes out more *purified* (more useful with regard to some task)

universidad de león

Computer Vision: basic concepts



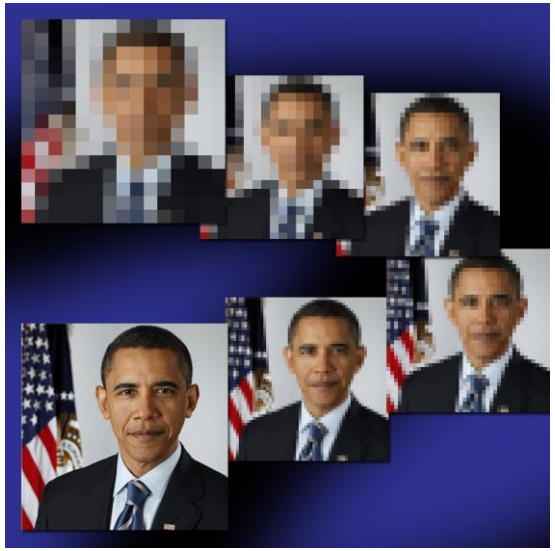
Computer Vision: science and technology that allows machine to see. Subfield of AI where we extract of information from images/videos to understand them.

Pixels: tiny little dots that form the image. The smallest visual elements that can be seen, physically located somewhere in a raster image. When an image is stored, the image file contains:

- Pixel Location
- Pixel Intensity

Resolution: total number of pixels in an image

Aspect Ratio: ratio width:height of the image.
E.g. an image of 1024x1024 has an aspect ratio of 1:1

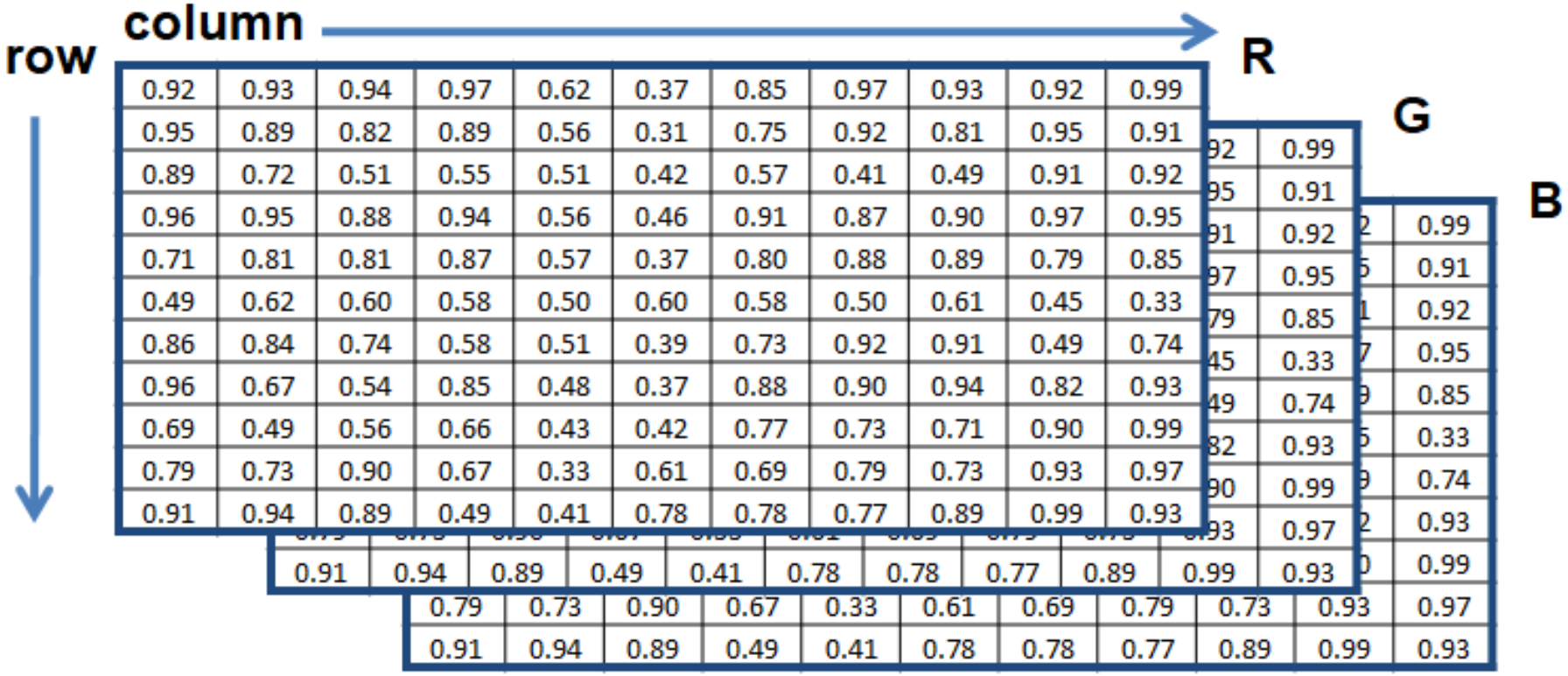


universidad de león

REGINNA 4.0

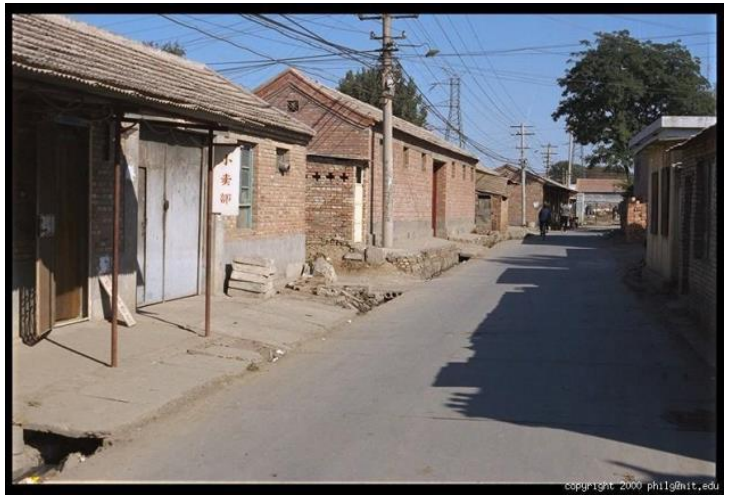
Computer Vision: basic concepts

- Images represented as a matrix



universidad de león

Computer Vision: basic concepts



universidad de león

REGINNA 4.0

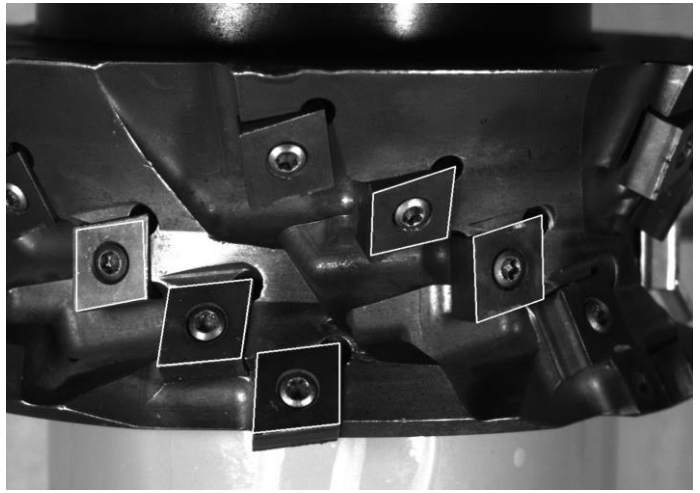
Computer Vision: image classification

Image classification: Assigning a class label to the image

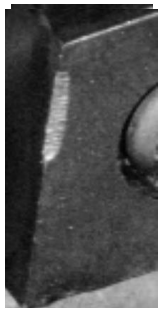
Does this image contain a cat?

Computer Vision: image classification

Image classification: Assigning a class label to the image



Does this image contain inserts?

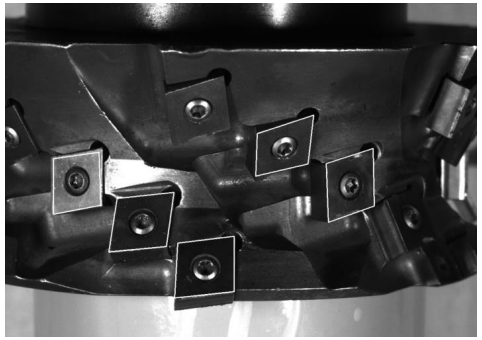


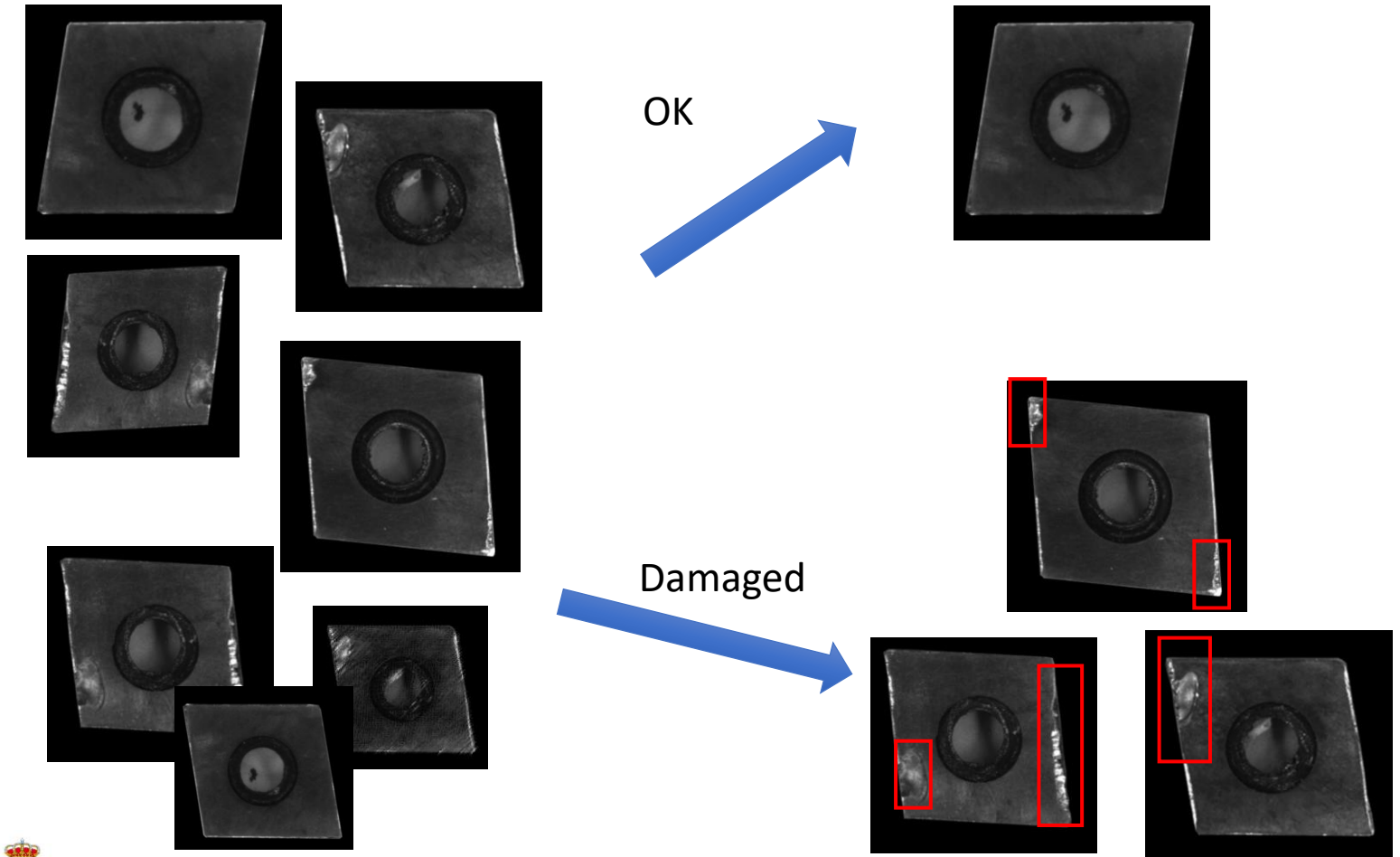
Does this image contain a damaged insert?

Does this image contain an integer insert?

Computer Vision: image classification

How do you classify the following images based on their content?



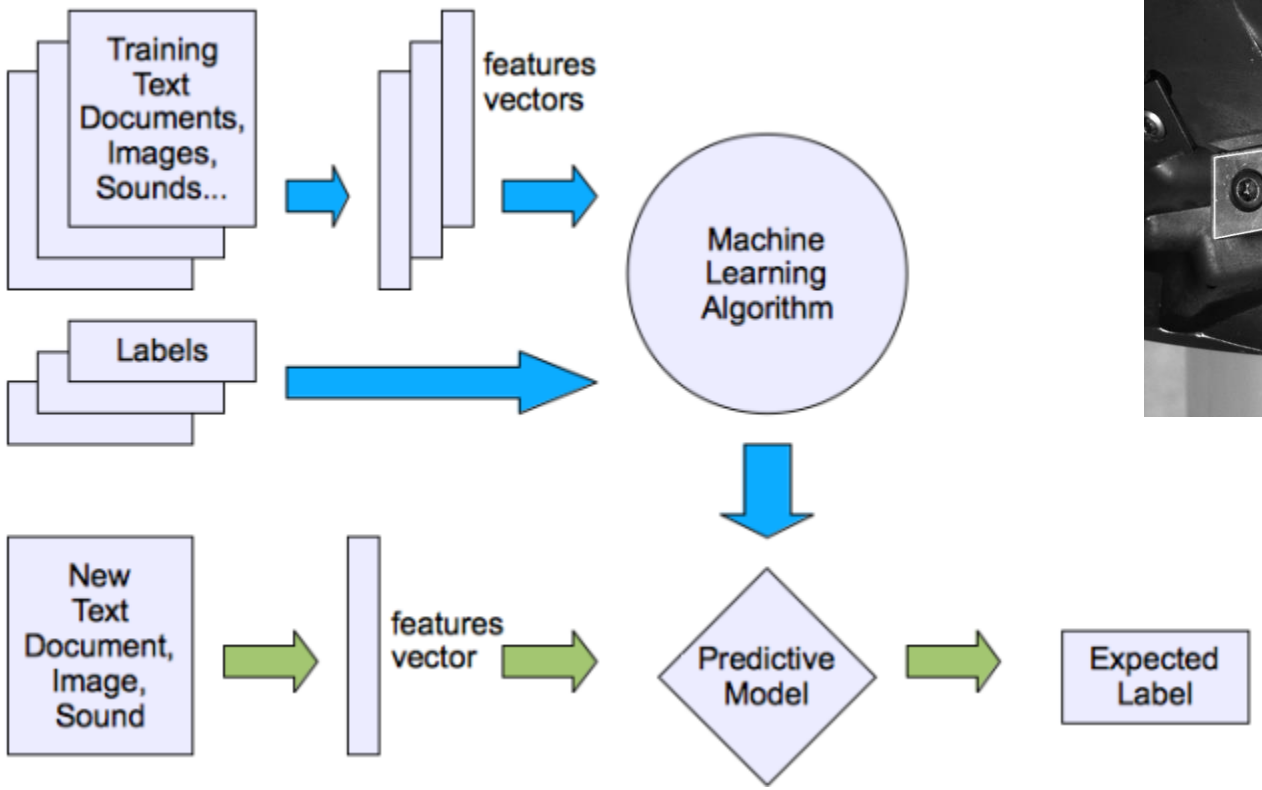


universidad de león

REGINNA 4.0

Computer Vision: image classification

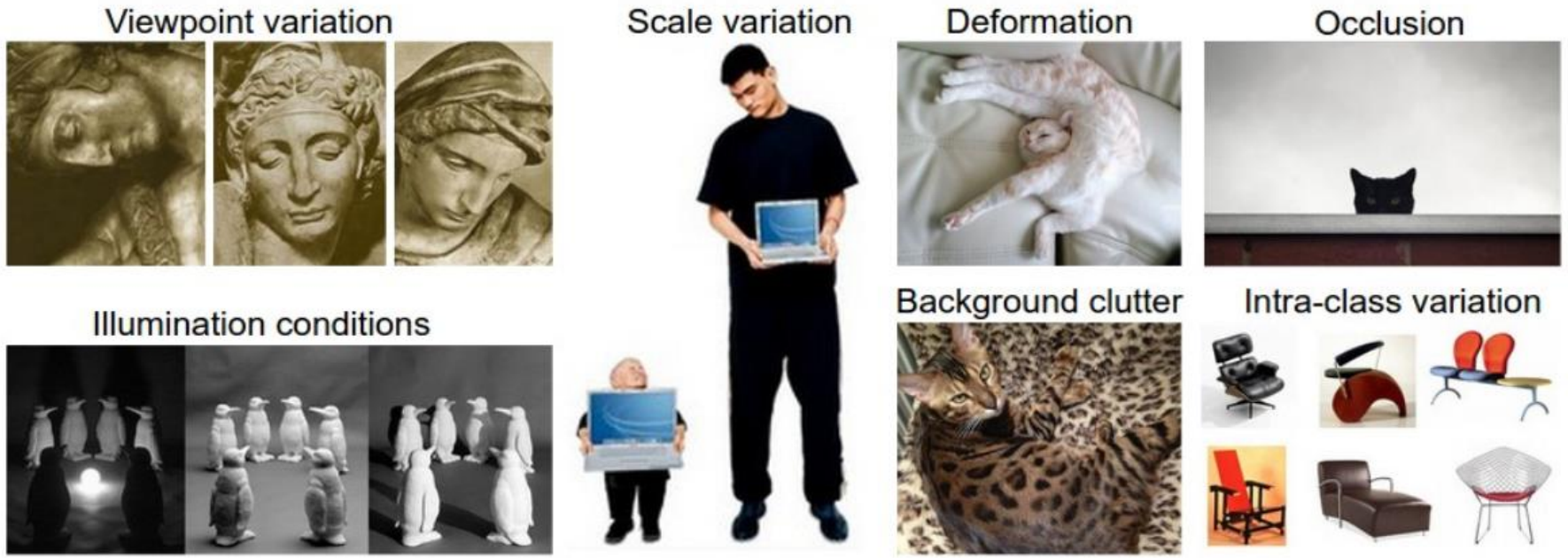
Supervised Learning



universidad de León

Computer Vision: image classification

Supervised Learning



universidad de león

REGINNA 4.0

Table of contents

universidad de león

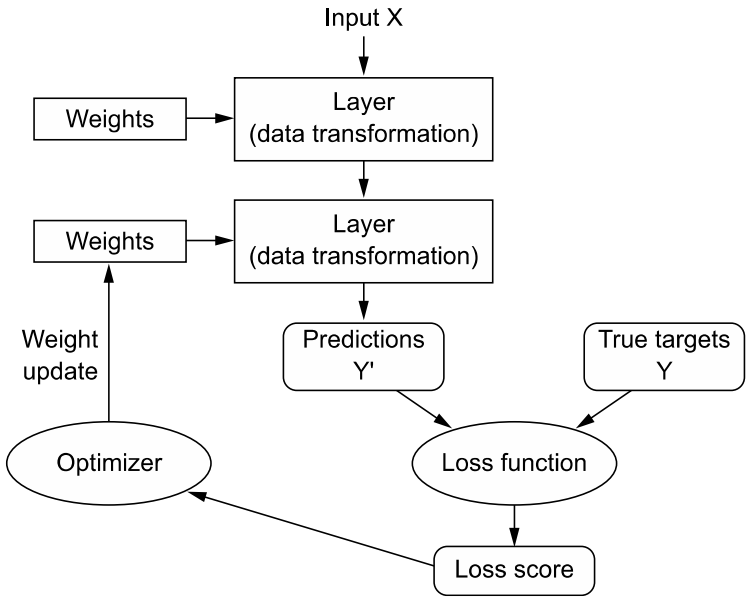
REGINNA 4.0

Neural Networks

What we need to build and train a Neural Network (parts of a NN)

- **Input data** and its corresponding *targets* or *labels*.
- **Layers of Neurons**, *(think about them as filters).*
  - Extract representations of the data fed into them.
  - Data goes in/out in a more useful form (for the problem at hand).
  - Are combined into a *network* (or *model*).
- **Loss function**: how the network measures its performance on the training data.
  Defines the feedback signal used for learning.
- **Optimizer**: Mechanism that defines how the network updates itself based on data and the loss function.
- **Metrics** to monitor during training/testing

Category: class
Data points: samples
Sample class: label



universidad de león

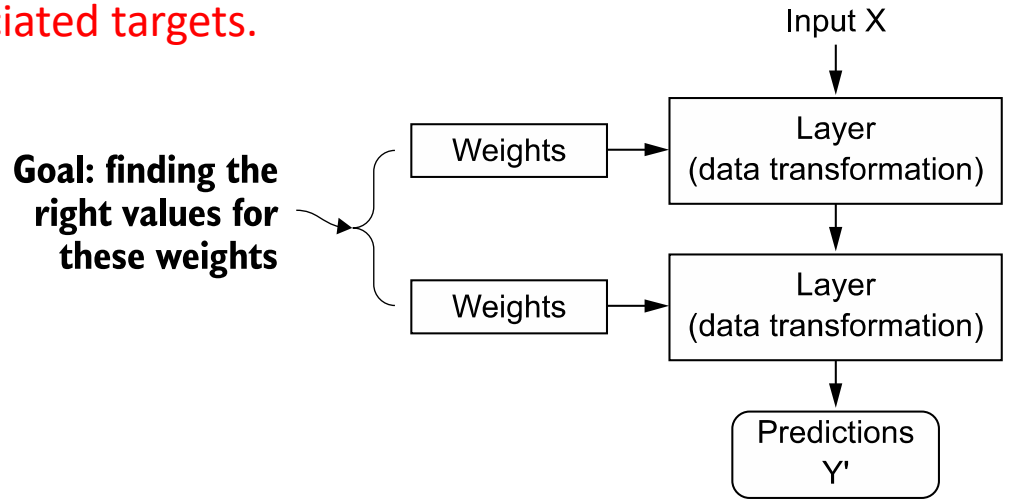
Neural Networks

What is the goal of training a neural network?

- Specification of **what a layer does to its input data is stored in the layers' *weights*** *(a bunch of numbers)*.

- More technical: transformation implemented by a layer is *parameterized* by its weights. Weights a.k.a network parameters.

- **Learning** means finding a set of values for the weights of all layers in a network, such that the network will correctly map example inputs to their associated targets.



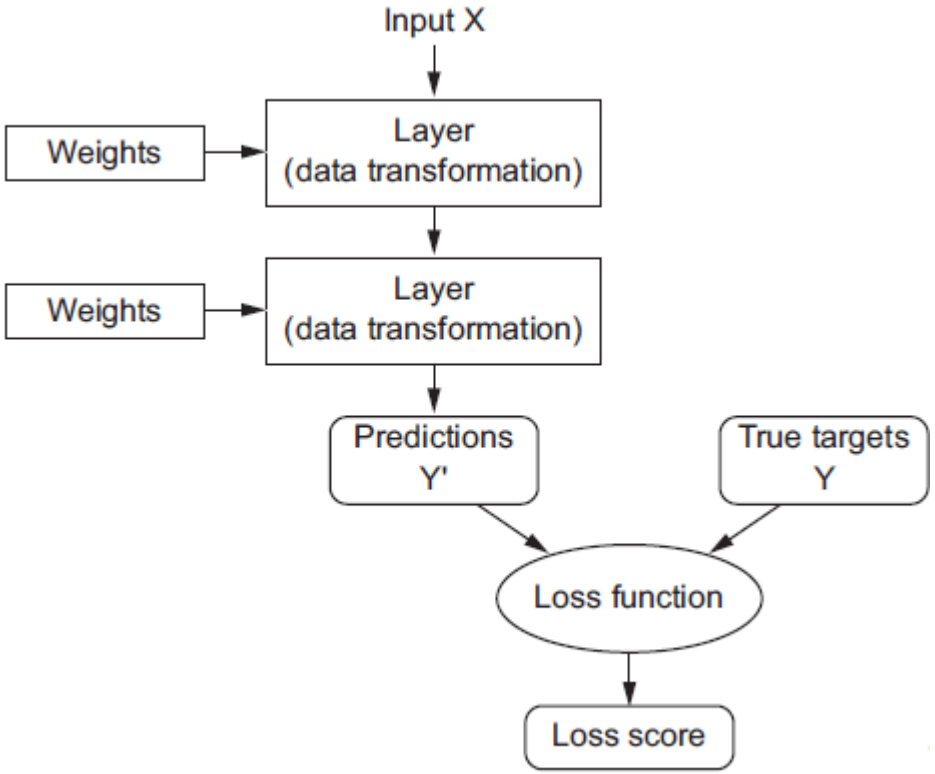
universidad de león

Neural Networks

What is the goal of training a neural network?

- To control something, first you need to observe it ☺
- To control the output of a neural network, you need to measure how far its outputs (**predictions**) are from what you expected (**true target** or real outputs)
- This is the job of the ***loss function*** of the network, also called the ***objective function***.

Loss function computes a ***distance score***



Distance score: how well the network has done on a specific example

universidad de león

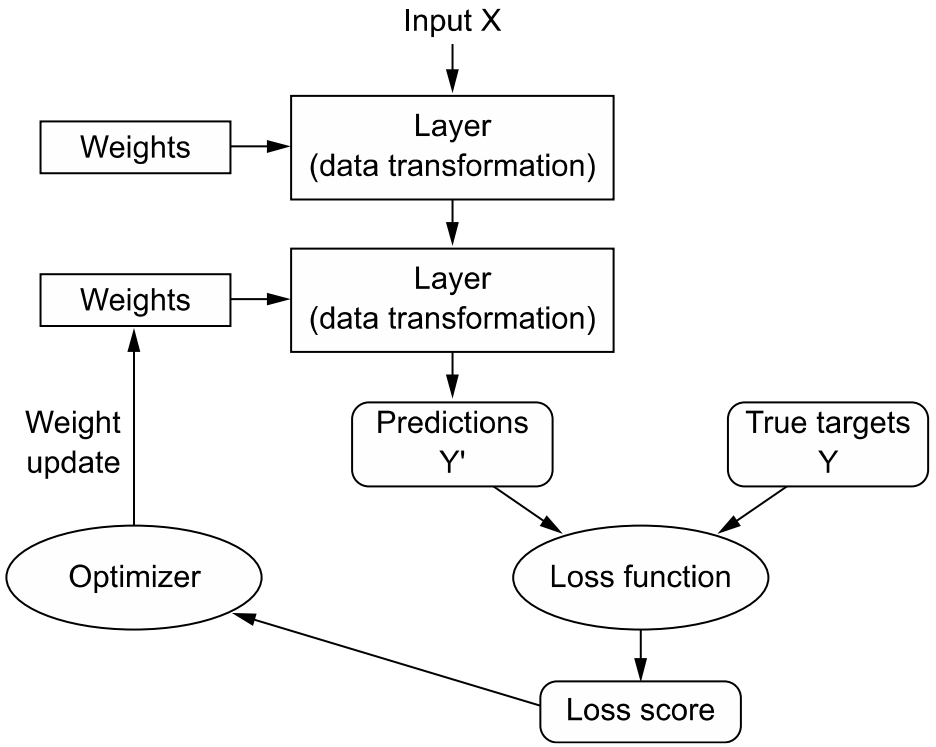
Neural Networks

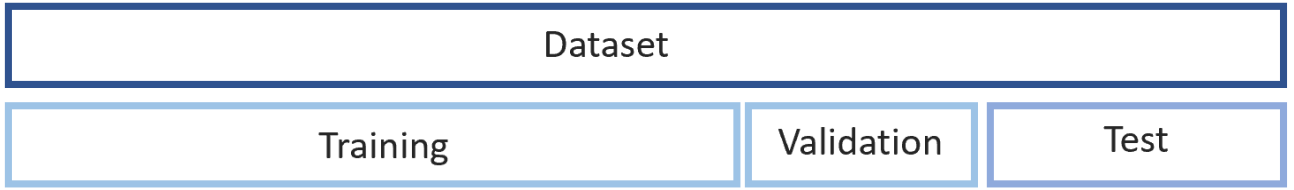
What is the goal of training a neural network?

Fundamental trick of Deep Learning -> use the loss score as feedback signal to **adjust** the weights a little, to decrease the score for the observed sample.

Optimizer makes the **adjustment**, and implements the ***Backpropagation***, central algorithm of Deep Learning.

Initially, random weights (Random transformations)… big score… thousands of iterations (**training loop**)… weight values that minimize loss function… ***trained network***.



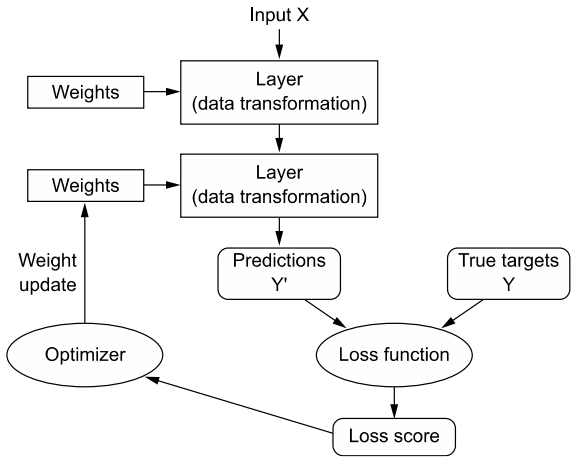


universidad de león

Neural Networks

Network Layers

Fundamental data structure of neural networks.
Input: tensors – Output: tensors

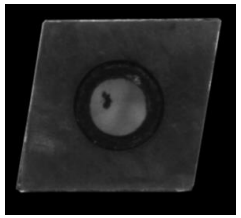


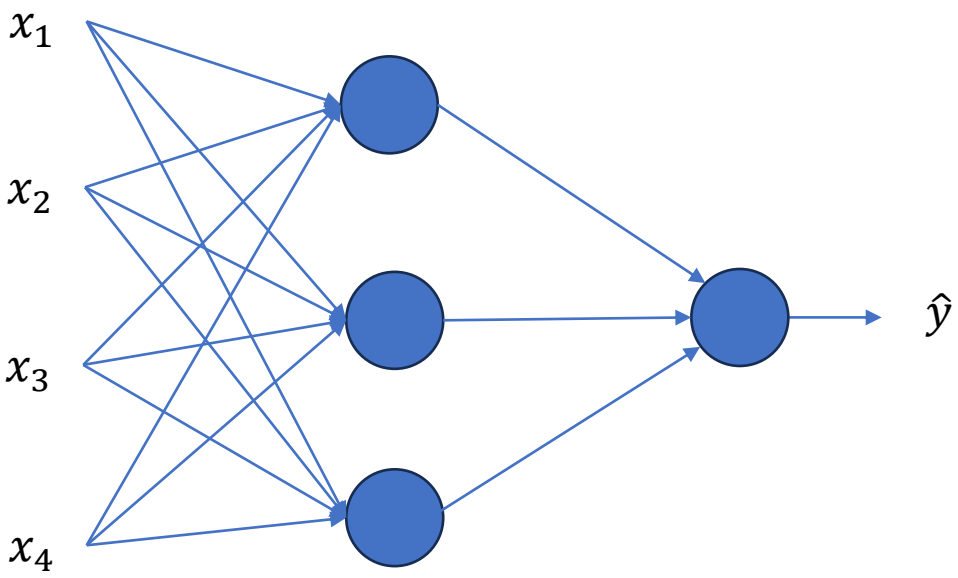
- Made by a set of nodes or **neurons**.
- Neurons **connected** between them.
- Frequently, layers have a state: the layers *weights*.
- **Weights**: Tensors learned with stochastic gradient descent, which are the **network's knowledge**.
- Depending on the tensor and data type:
  - **Fully Connected layers**: Process 2D tensors. Often used for classification.
  - **Recurrent layers**: Process 3D tensors storing time series data.
  - **Convolutional Layers**: Process Image data, stored in 4D tensors.

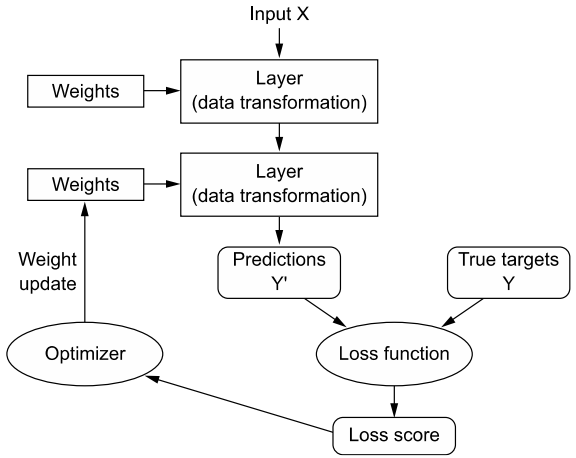
universidad de león

Neural Networks

Network Layers (and the weights?)







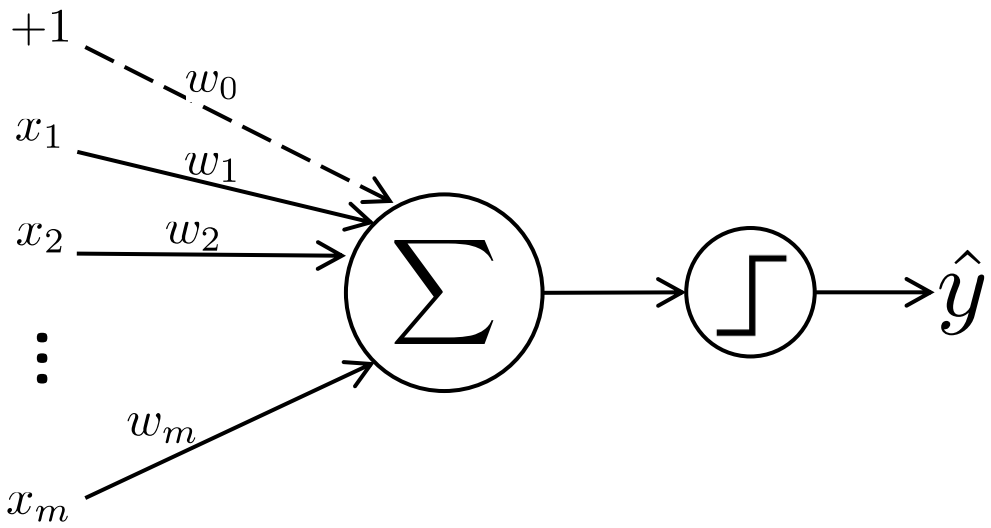
Integer
Damaged

universidad de león

Neural Networks

Network Layers (and the weights?)

The perceptron is the most basic building unit of neural networks. Neurons are functions!!



Inputs Weights Sum Activation function Output

Activation function

$$\hat{y} = \sigma(z)$$

$$z = b + \sum_{i=1}^{m} x_i \, w_i$$

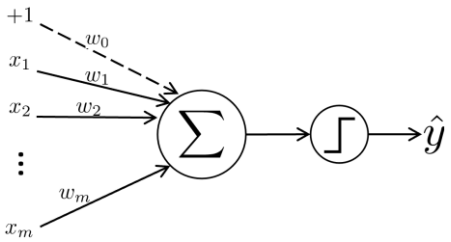
$$z = w^T x + b$$

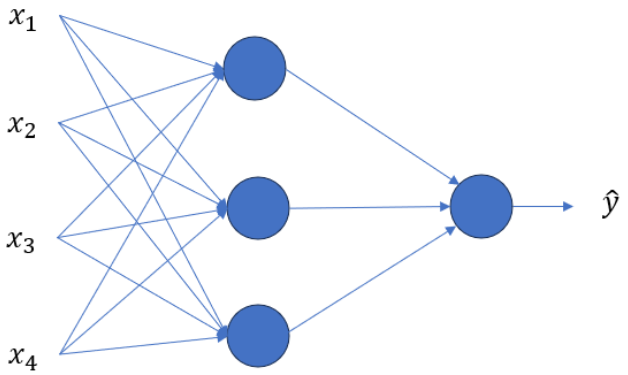
universidad de león

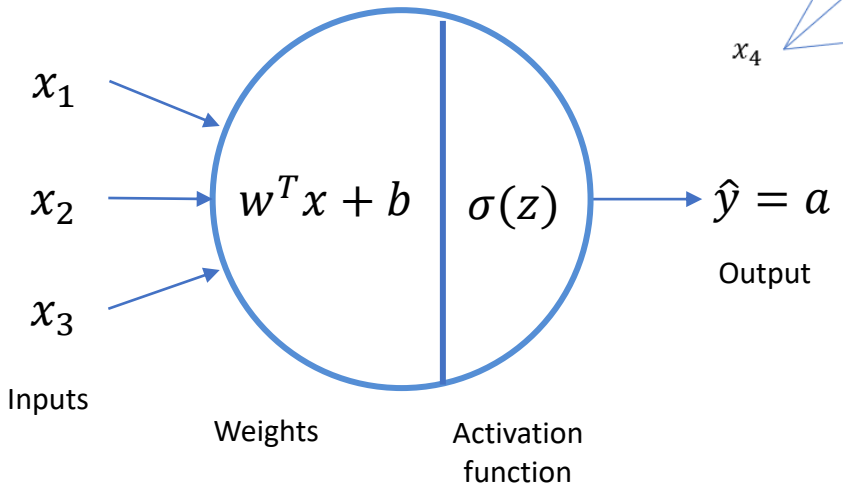
Neural Networks

Network Layers (and the weights?)

The perceptron is the most basic building unit of neural networks. Neurons are functions!!







Activation function

$$\hat{y} = \sigma(z)$$

$$z = b + \sum_{i=1}^{m} x_i \, w_i$$

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

universidad de león

Neural Networks

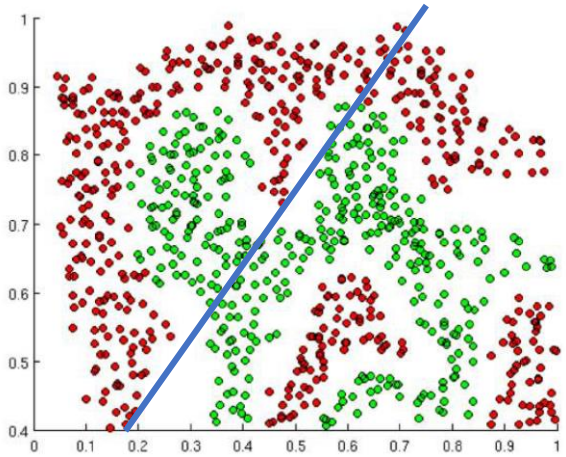
Network Layers (activation function, what for?)

$$\hat{y} = g(z) = g(w^T x + b)$$

The purpose of activation functions is to introduce non-linearities.

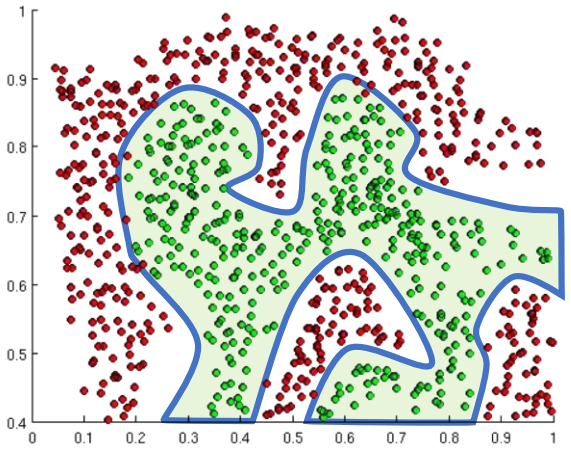
When you build a neural network, one of the choices to make is what activation function to use in

- the **hidden layers** and
- at the **output units**.



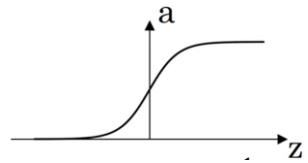
Linear activation functions produce linear decisions, no matter the network size.

$$\hat{y} = w^T x + b$$



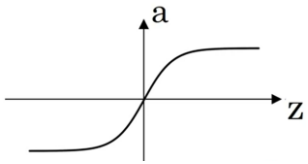
Non-linearities in the activation functions allow us to approximate complex functions.

Sigmoid



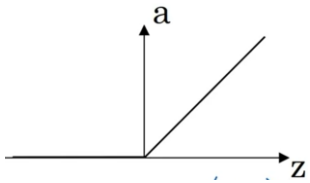
$$\sigma(z) = \frac{1}{1+e^{-z}}$$

Tanh



$$tanh(z) = \frac{e^z+e^{-z}}{e^z+e^{-z}}$$

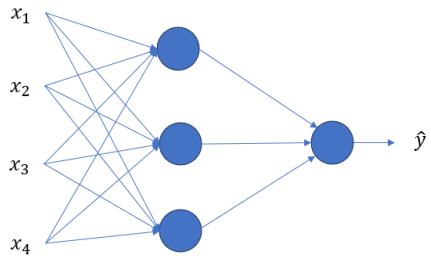
Rectified Linear Unit (ReLU) :

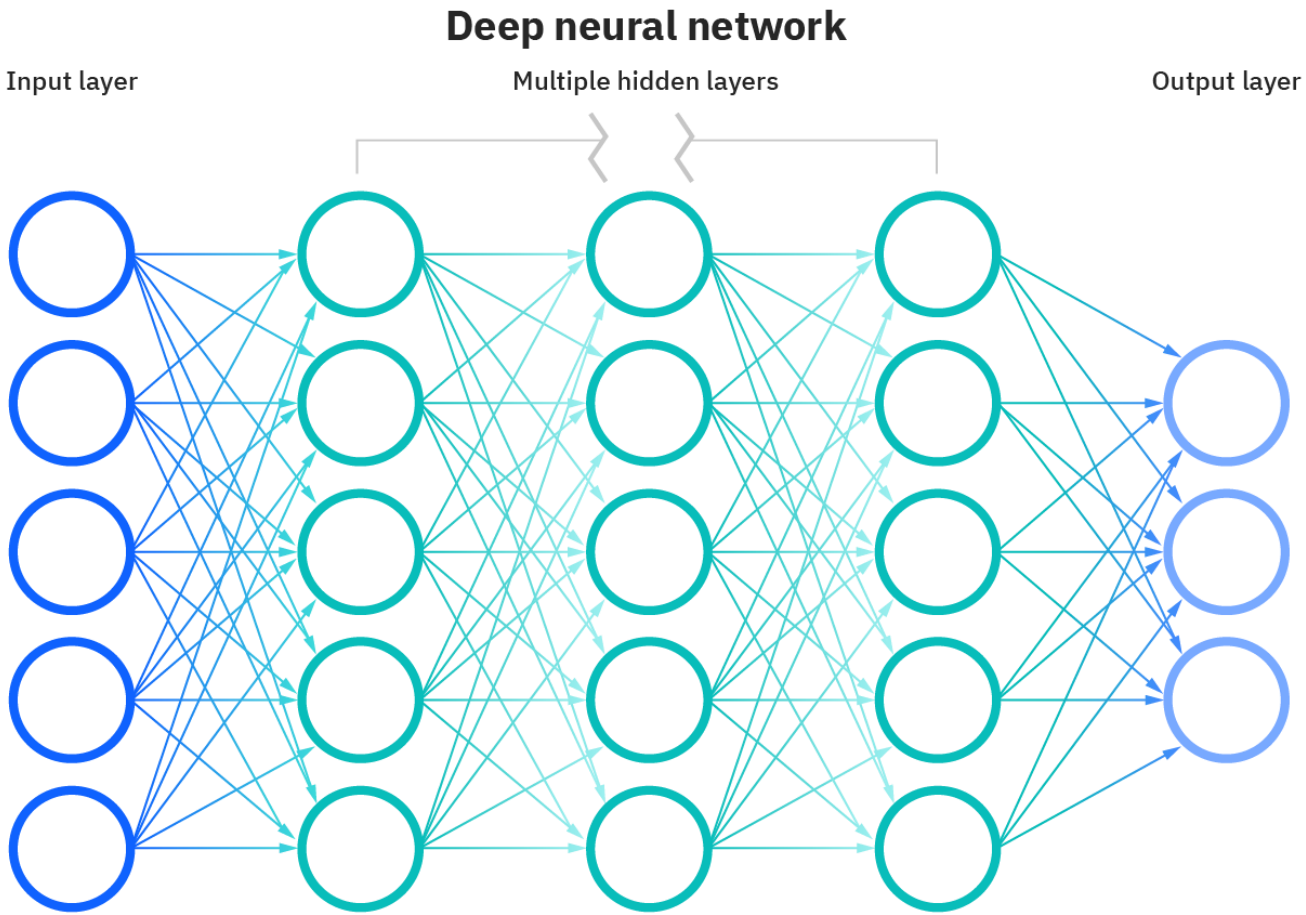


$$ReLu(z) = \max(0, z)$$

universidad de león

Neural Networks





universidad de león

REGINNA 4.0

Table of contents

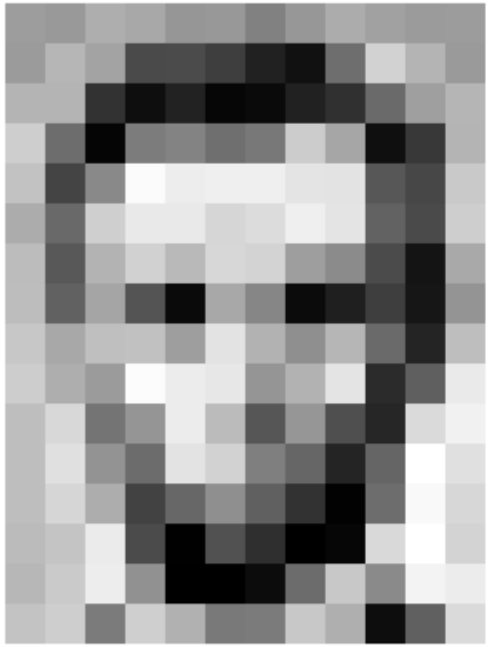
universidad de león

REGINNA 4.0

Visual features

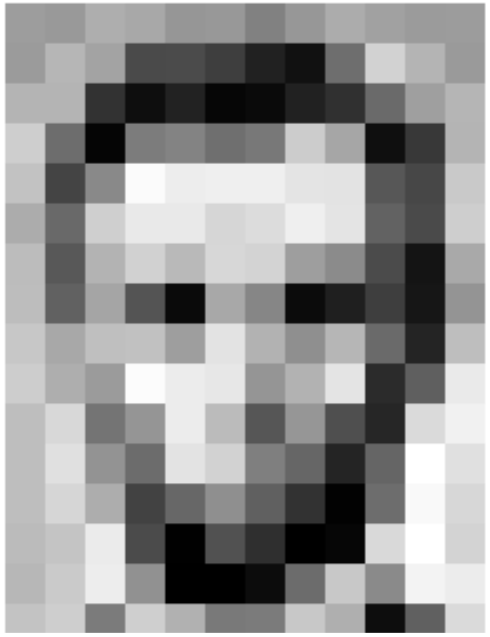
Example:

Which US president is this?



Visual features

What we see… is not what a computer sees…



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
196	206	123	207	177	121	123	200	175	13	96	218

Visual features

We cannot work with pixel values (intensities) as they are.
Not invariant to certain changes

3	75	12
39	80	102
150	195	200

≠

80	15	30
103	110	23
196	208	19

We should use **descriptors**!

An image descriptor “describes” a **feature** of a region in an image. Describes -> “a vector that represents…”

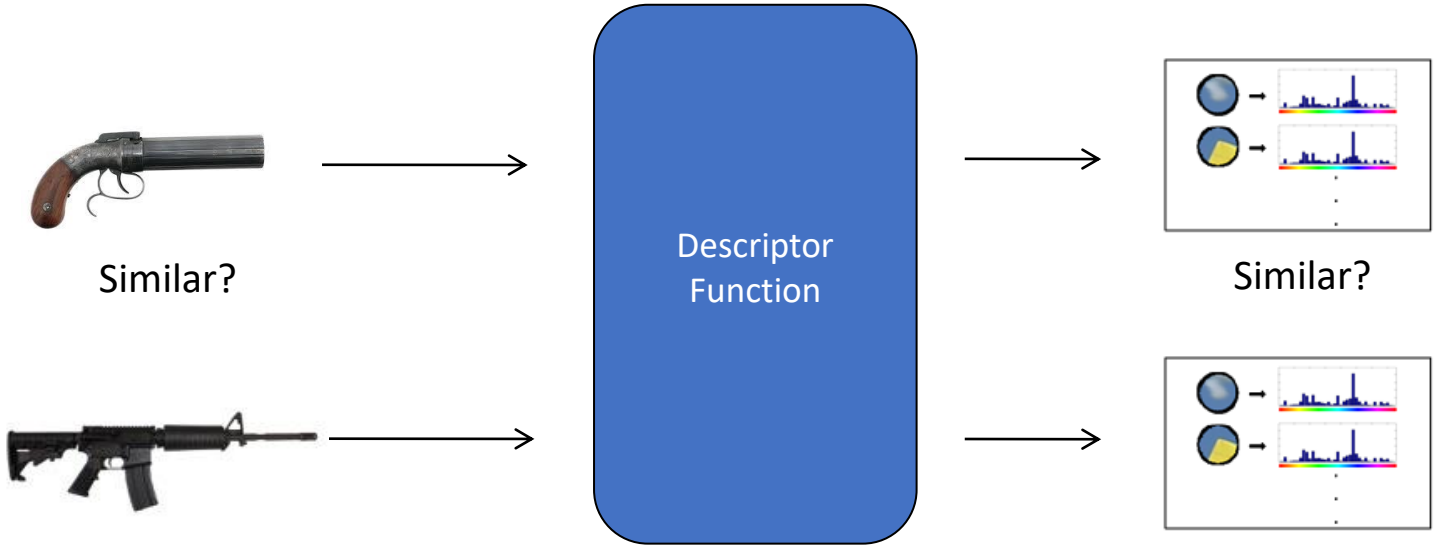
What is a feature? Shape, colour, texture, motion, location…

Visual features

An image descriptor is a vector with *n* variables that represents a region within an image

- **The region may be the complete image**

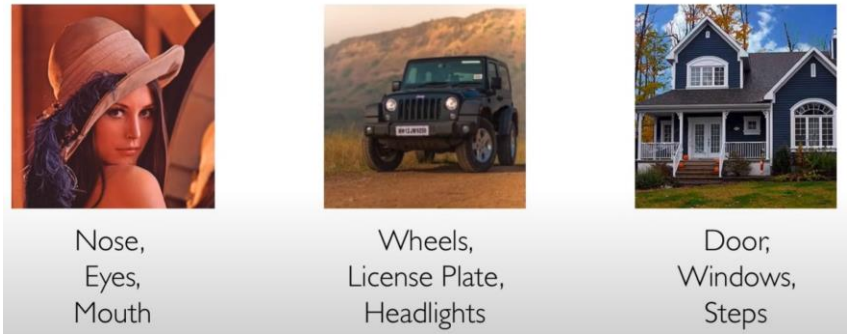
To compare two such regions/images, descriptors will be compared



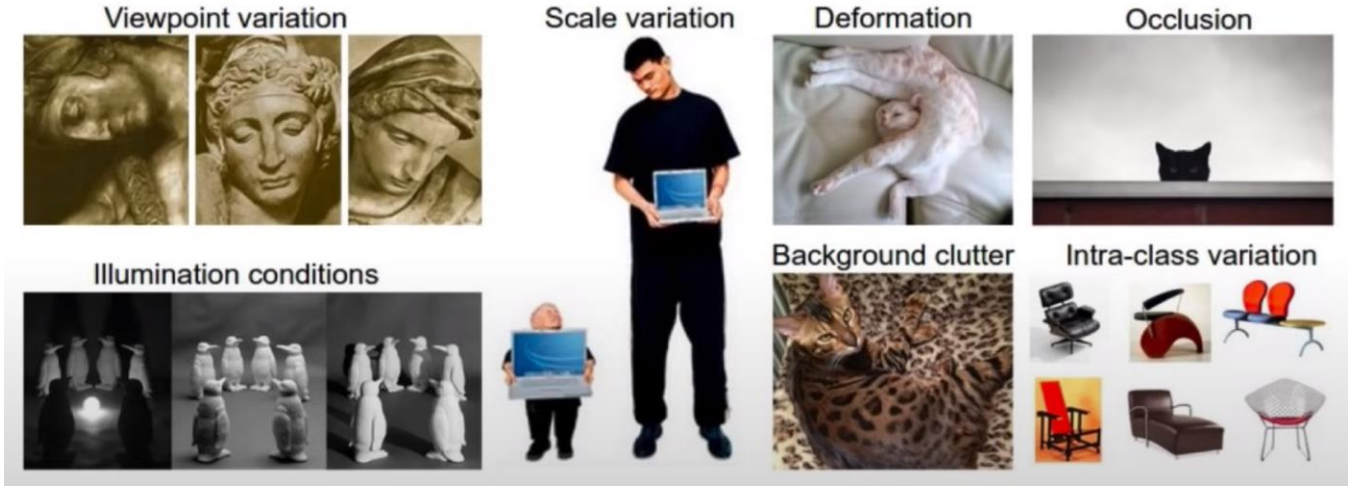
universidad de león

Visual features

Identify key features on each category we are working in



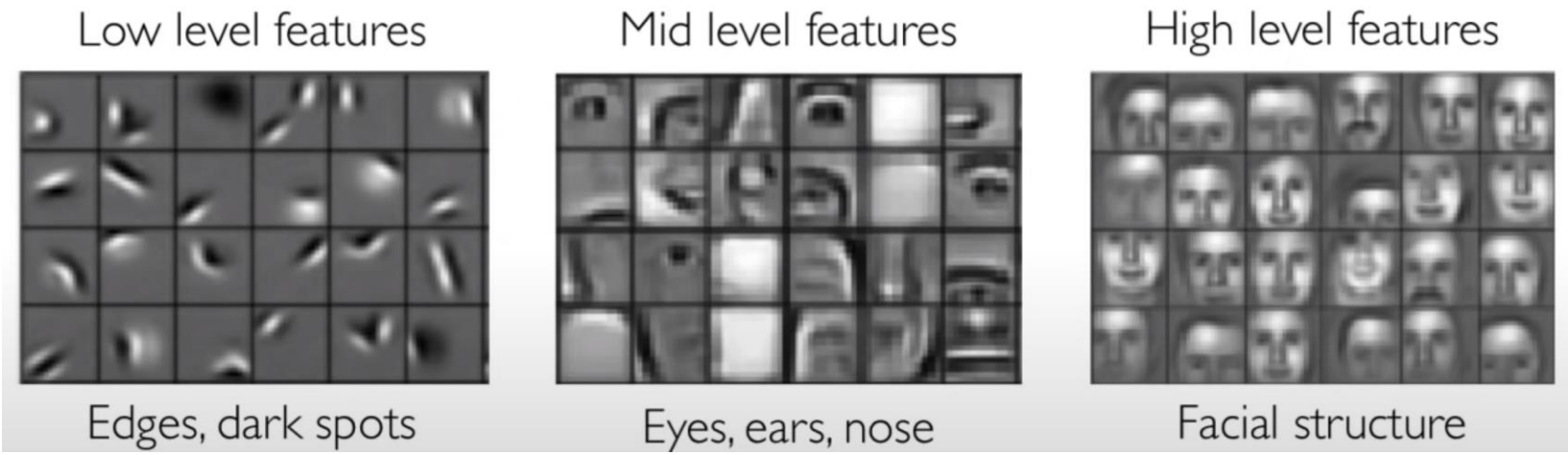
Not an easy task… we are looking a bunch of brightness values (intensity) that can represent anything (e.g., objects, persons…), and "anything" can be presented in multiple situations (i.e. multiple problems)



universidad de león

Visual features

Can we learn a hierarchy of features directly from data Instead of hand engineering? **For any specific task**

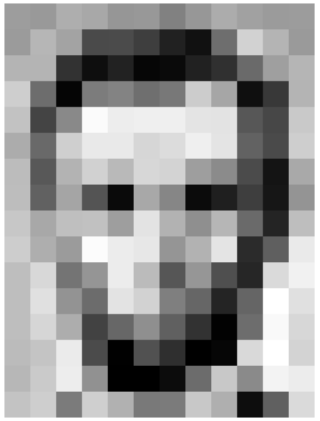


universidad de león

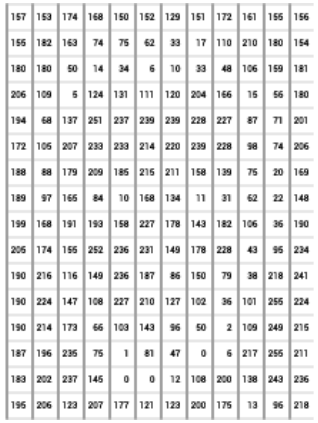
REGINNA 4.0

Visual features

NN allow us to build (learn) these features
If we build them correctly ☺



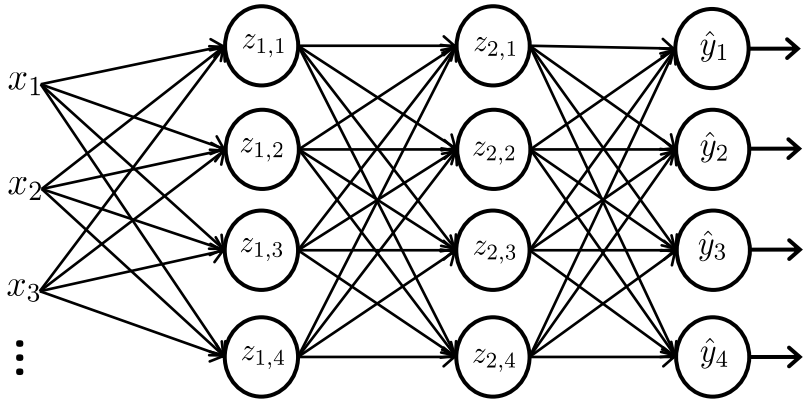
Input image



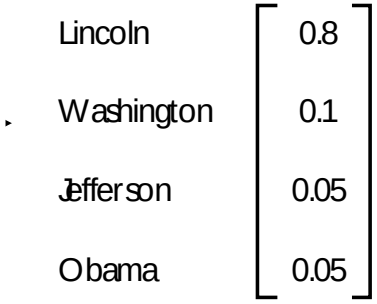
Pixel representation

$$\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}]$$

Feature extraction
Image Descriptor (in this case global)



Classification

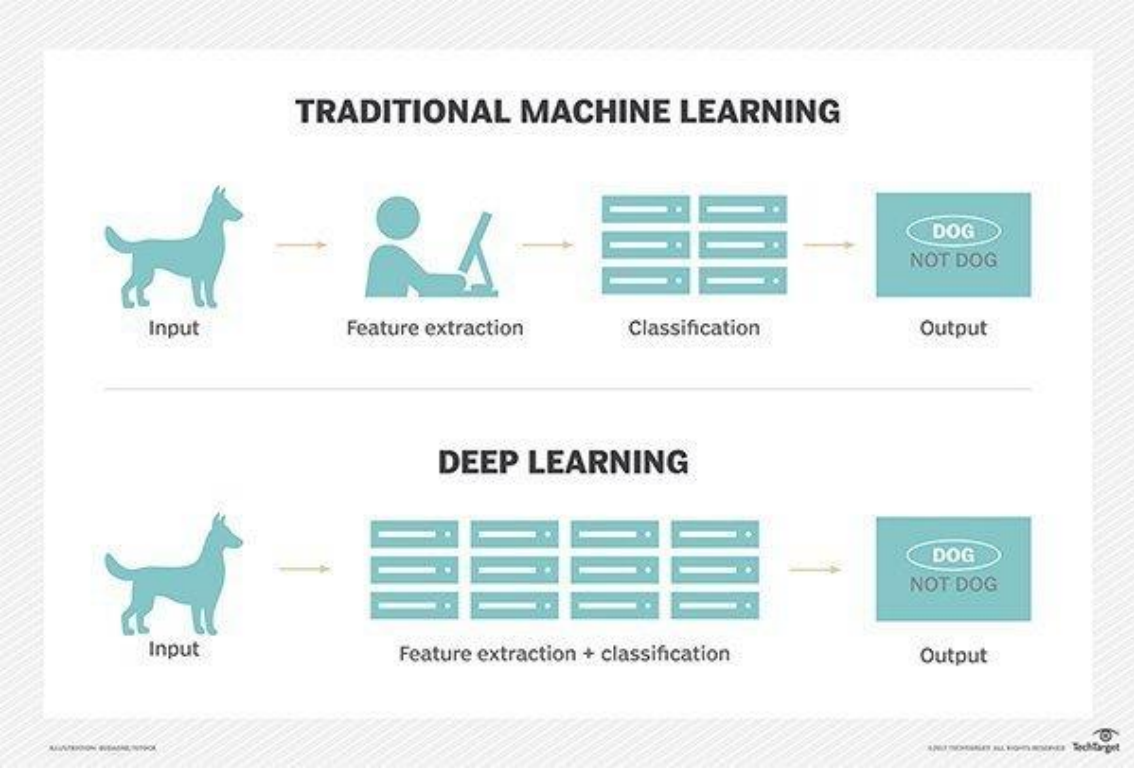


Prediction

universidad de león

Visual features

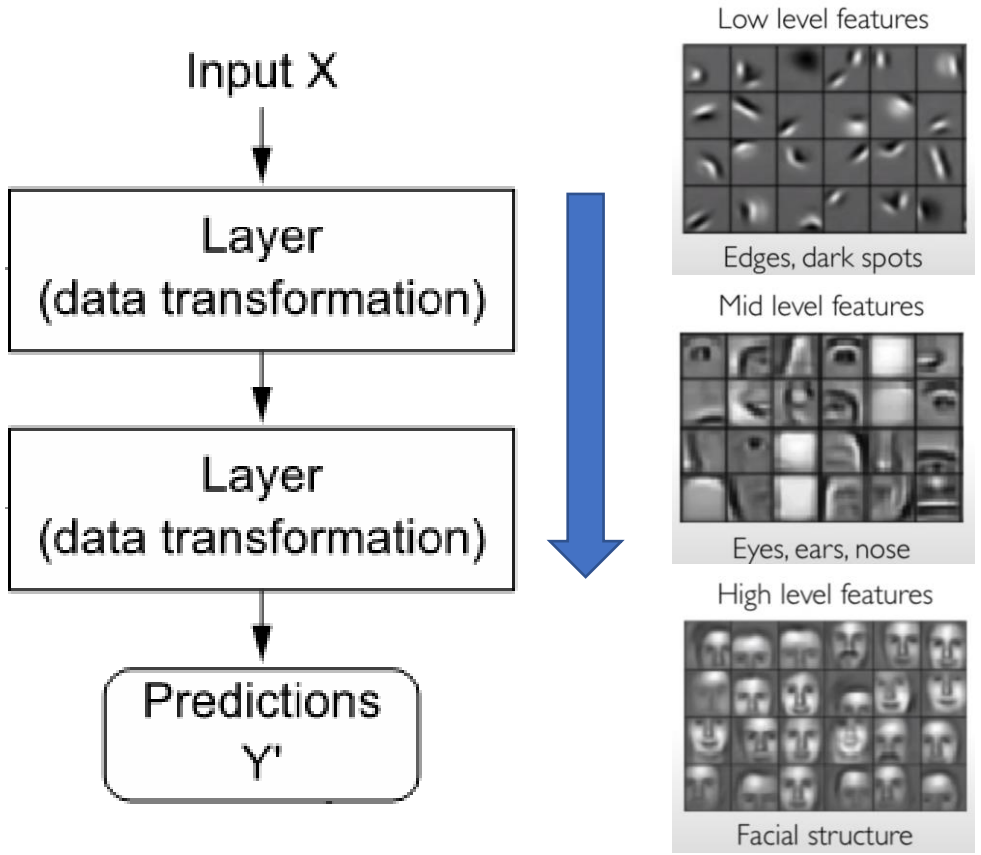
Deep Learning automates what used to be the crucial step in traditional Machine Learning (shallow learning) workflow: **Feature engineering**.



universidad de león

Visual features

- **Feature hierarchy**: Each **layer's input is the output of the previous layer's output**.
- The deeper you go in the network, the more complex the features become.
- All layers' weights are learnt *jointly*, at the same time, rather than in a succession.

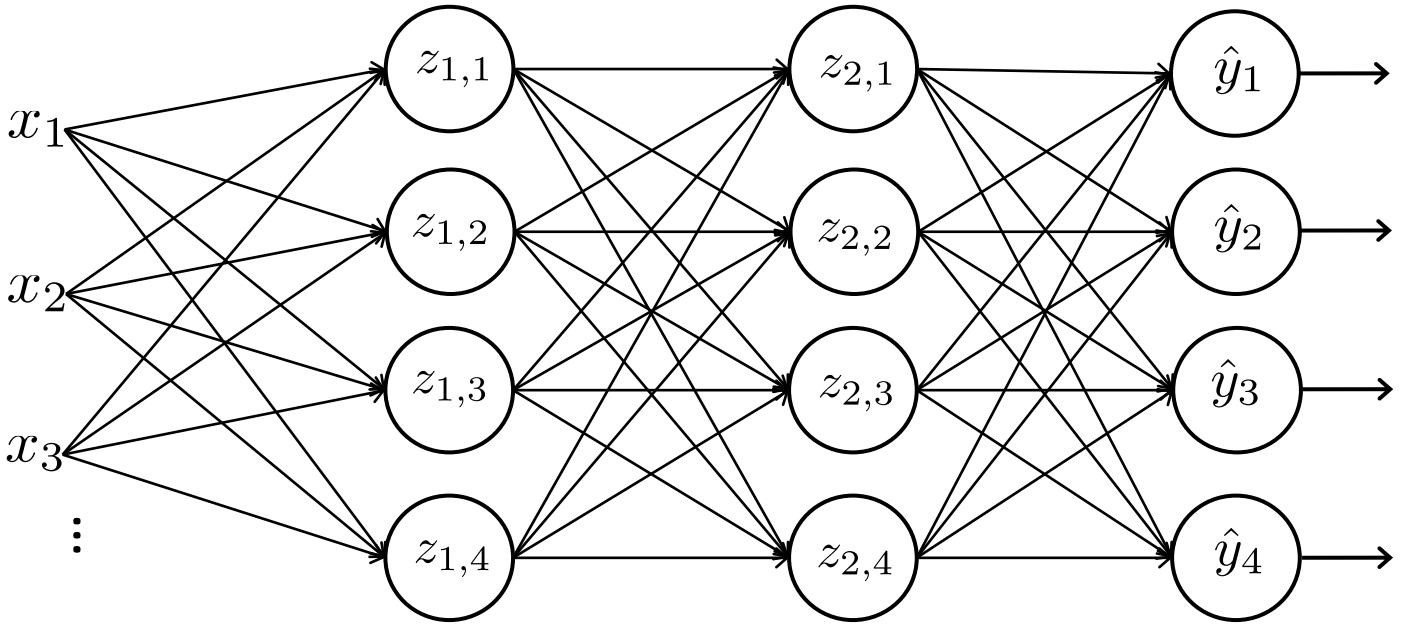


universidad de león

Visual features

What we know so far? Dense Layers

- **Fully connected Neural Network**
- Densely connected: every input is connected to every output

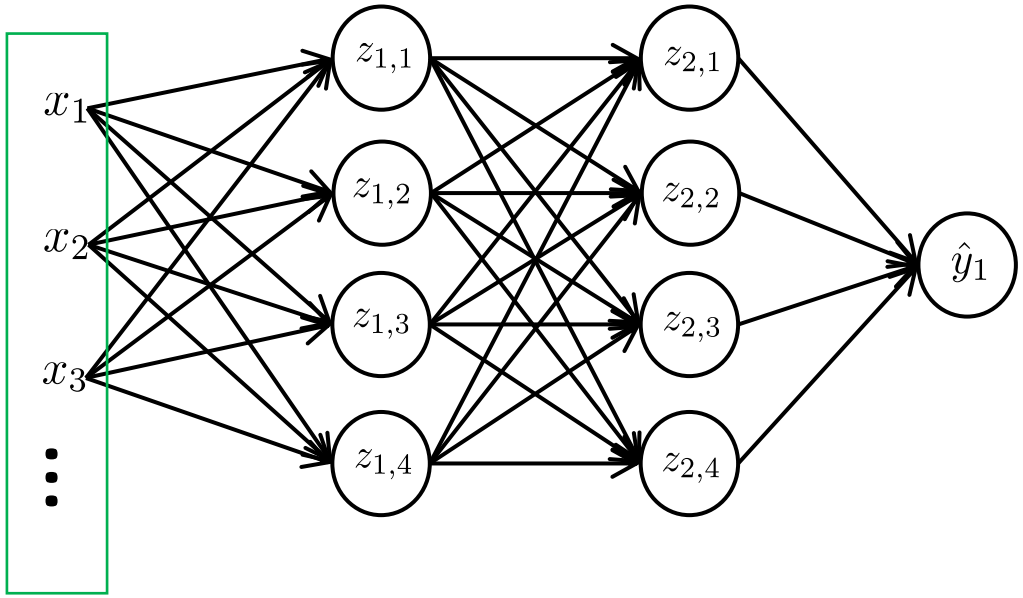


universidad de león

Visual features

Input:

- 2D Image we convert to a vector of pixel values



- We are losing spatial information!!
- Many parameters. Densely connected, because we are connecting every single pixel to every single neuron in our hidden layers.

How can we use spatial structure in the input to inform the architecture of the network? We should be able to use this prior knowledge we have…

Visual features: Image Convolution

How to keep spatial structure ➔ Connect **patches of the input** to neurons in the hidden layer of the neural network.

The neuron connected to the region of the input only "sees" this value. In other words, this neuron is only influenced by the patch connected to it.

Input:

- 2D Image
- Array of pixel values

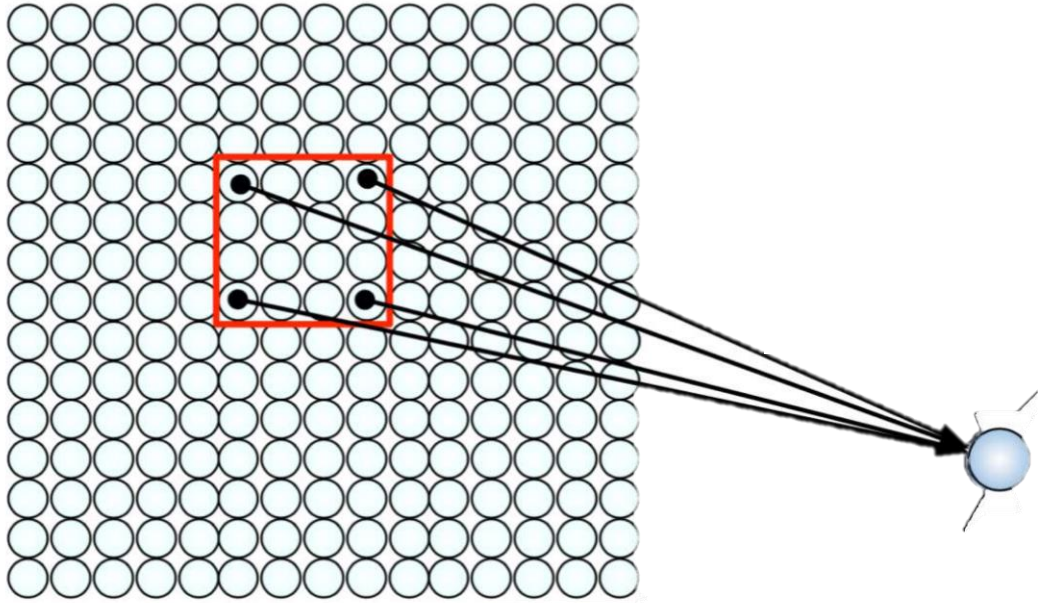
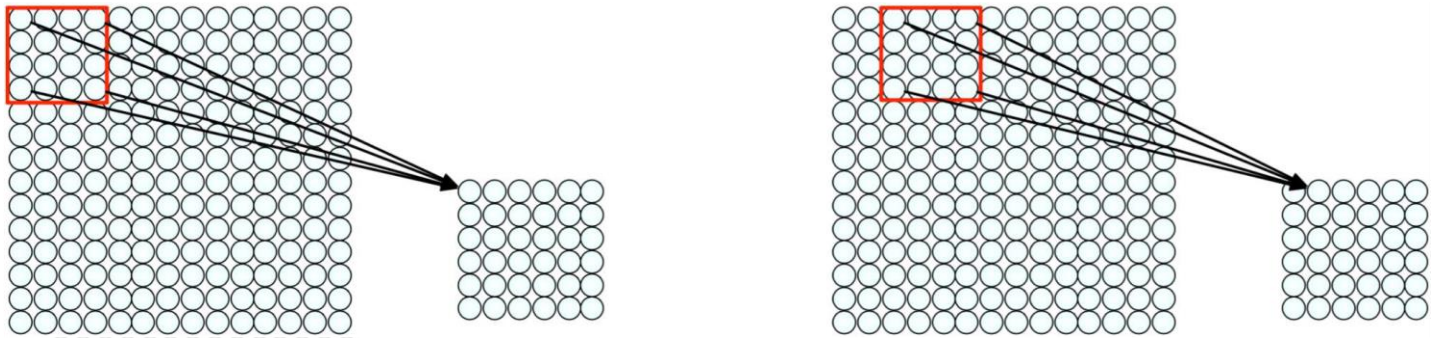


Image Convolution

Connect **patches of the input** to neurons in the hidden layer.

Use a sliding window to define connections.

How can we **weight** the patch to detect **particular** features? **Using filters!**

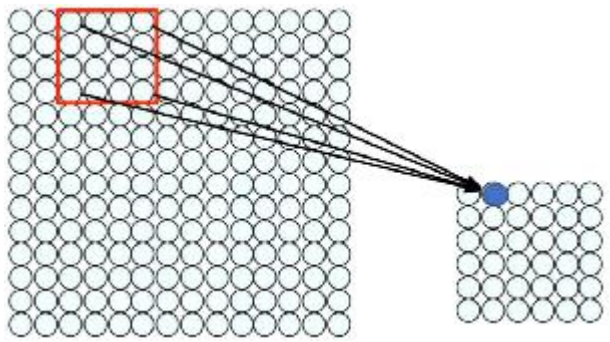


Instead of weighting single pixel values, we are going to weight these patches

Image Convolution

From the example:

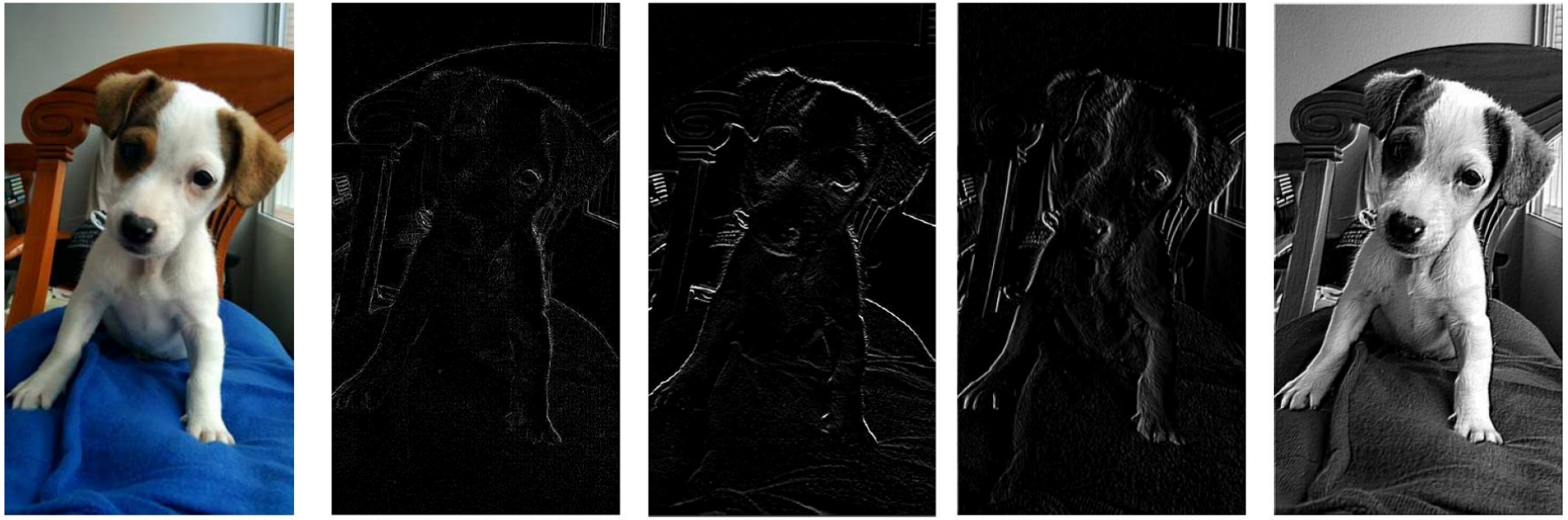
- Filter of size 4x4: 16 different weights
- Apply same filter to 4x4 patches to all image:
  we will use the result of this operation to define the neuron state in the next hidden layer
- Shift by 2 pixels for next patch



This “patchy” operation is **convolution** ☺

1. A patch (2D-array, matrix) with a set of weights is a **filter**.
2. Each **filter** highlights a different type of **local feature**
   - For example, horizontal/vertical edges
3. Applying many different filters, you would get different local features.
4. Spatially share parameters of each filter (i.e., features that matter in one part of the input should matter elsewhere).

Image Convolution



Laplacian

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Sobel H

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Sobel V

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Emboss

$$\begin{bmatrix} -2 & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

Adapted from CVBLAB

https://miro.medium.com/max/1640/0*faJQRB08AFT4C3Z5.gif

Image Convolution

Convolution operation

Operation between two functions, that give a third function **that expresses how the shape of one function is modified by the other**

$$g(x,y) = f(x,y) * h(x,y)$$

Instead of working with functions , we work with images. Matrixes with Image pixel values

$$f(x,y) \rightarrow I(x,y)$$

Operation between two **images**, that give a third **image** that expresses how the shape of one **image** is modified by the other

$$resulting_{image} = original_{image} * kernel_{image}$$

$$g(x,y) = f(x,y) * h(x,y)$$

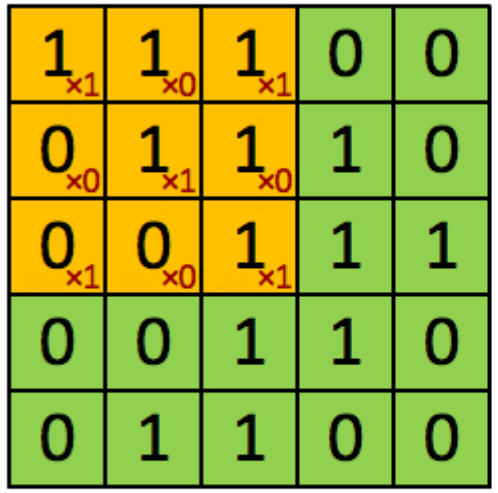
1	0	1
0	1	0
1	0	1

Operation between **images** of the same size. Summation of the multiplication of those elements that occupy the same position in the matrix.

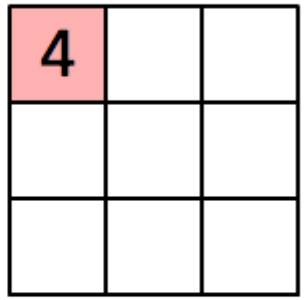
https://miro.medium.com/max/1640/0*faJQRB08AFT4C3Z5.gif

Image Convolution

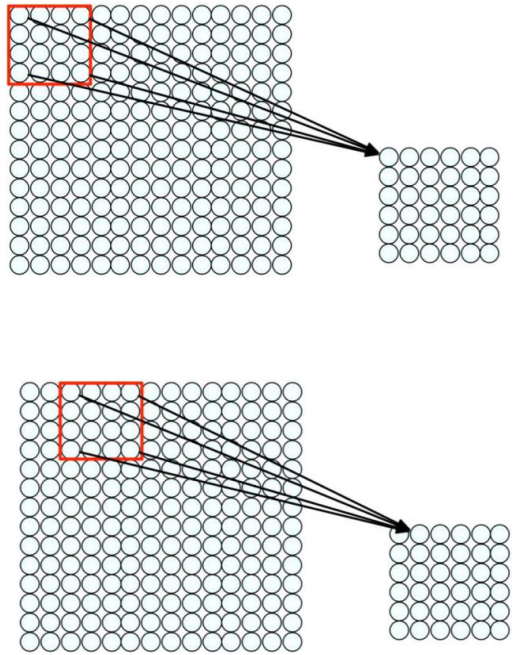
Convolution operation



Image



Convolved Feature



universidad de león

Table of contents

universidad de león

REGINNA 4.0

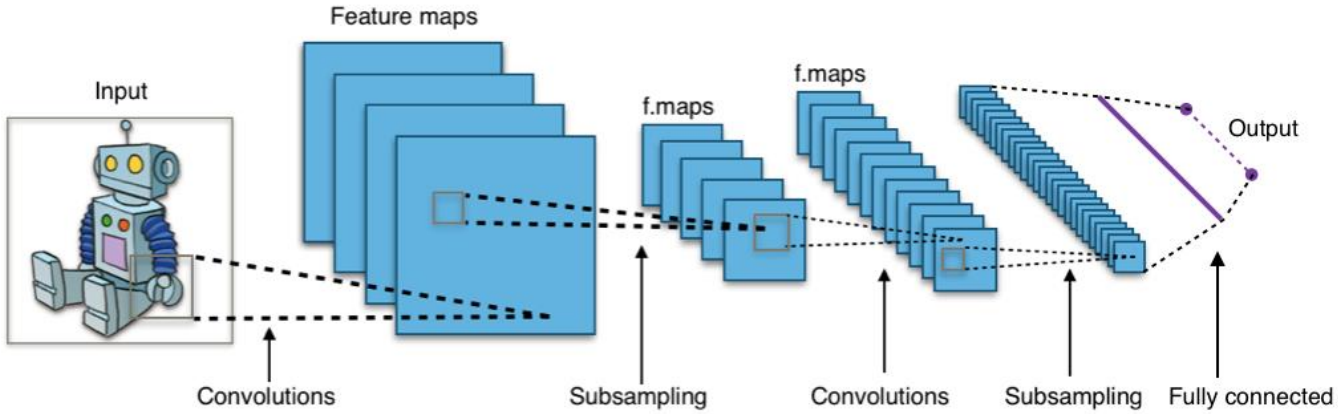
Liu, Z., Mao, H., Wu, C. Y., Feichtenhofer, C., Darrell, T., & Xie, S. (2022). A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11976-11986).

Convolutional Neural Networks

Convolutional Neural Networks (CNN): type of NN architecture with three types of layers:

- Convolutional layers: Convolve the input data. Extract feature maps.
  - Non-linearity
- Pooling layers: Keeps the most important features (subsampling).
- Fully connected layers: Classification layers.

Some examples: Inception v3, ResNet, VGG16, …, EfficientNets, **ConvNeXts***

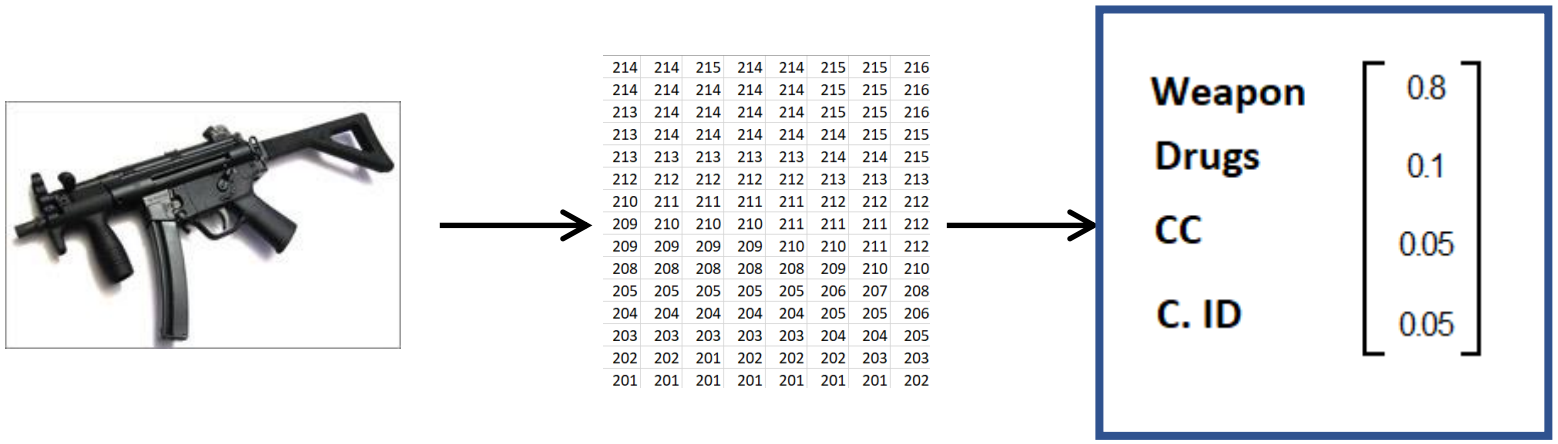


Adapted from CVBLAB

Convolutional Neural Networks

Definition of CNN

- A Convolutional Neural Network (CNN) takes images
- The first layers of these networks are a set of **convolutional layers** and **pooling layers**. **That make possible to learn meaningful features directly from the images instead of using *handcrafted* features.**
- They are usually followed by a set of **fully connected layers** that carry out the classification using the patterns learnt before.



universidad de león

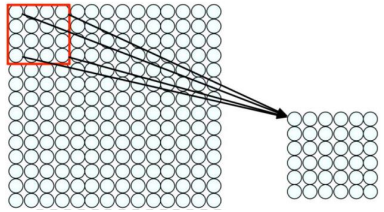
https://keras.io/api/layers/convolution_layers/

Convolutional Neural Networks

CNN (Convolutional Layer)

```
layers.Conv2D(16, 3, padding='same', activation='relu')
```

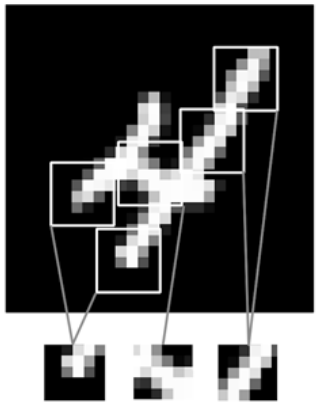
- Fundamental difference with fully connected layers:
  - Dense layers learn global patterns in the input feature space.
  - **Objective. Convolutional layers learn local patterns found in small 2D windows**.



4x4 filter: matrix of weights w_{ij}

$$\sum_{i=1}^{4}\sum_{j=1}^{4} w_{ij}\, x_{i+p,j+q} + b$$

for neuron (p,q) in hidden layer



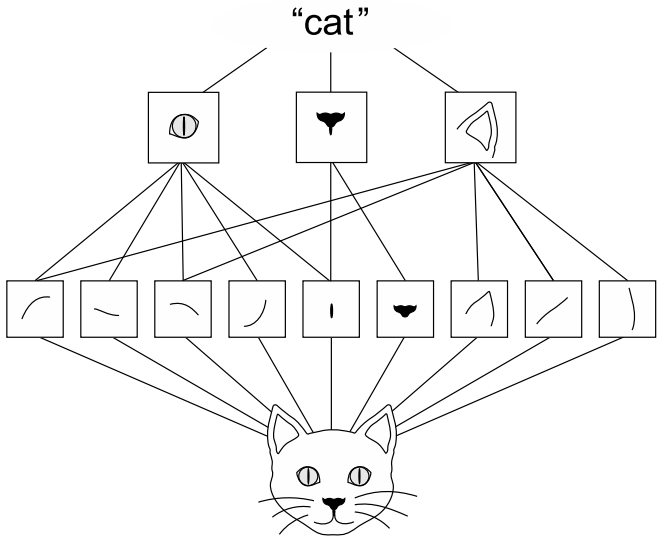
- Convolutional layers **can** carry out **convolutions** to the input images with multiple filters (or multiple features to extract).
- Each filter extracts different **local features** from the image.
- Filters (kernel values) are **learnt during the training process**.

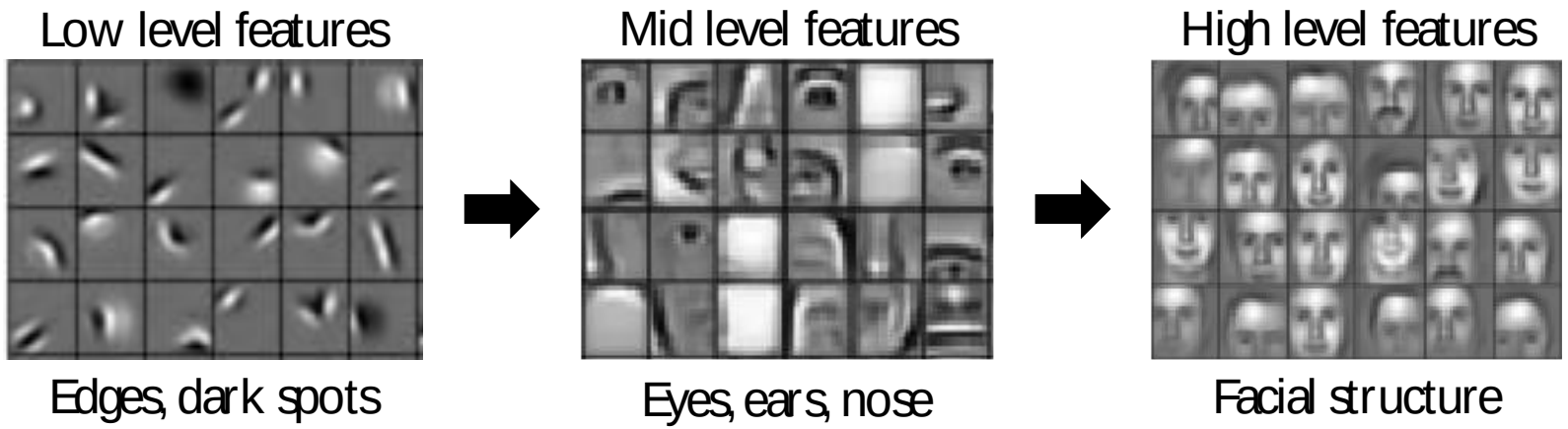
universidad de león

Convolutional Neural Networks

CNN (Convolutional Layer)

- Convnets learn **translation-invariant patterns**: After learning a pattern in one area of the image, it can be recognized anywhere.
- **Hierarchy**: The output of one convolutional layer is the input image of the next convolutional layer.
- They can learn **spatial hierarchies of patterns**: Convnets can increasingly complex and abstract visual concepts.





universidad de león

REGINA 4.0

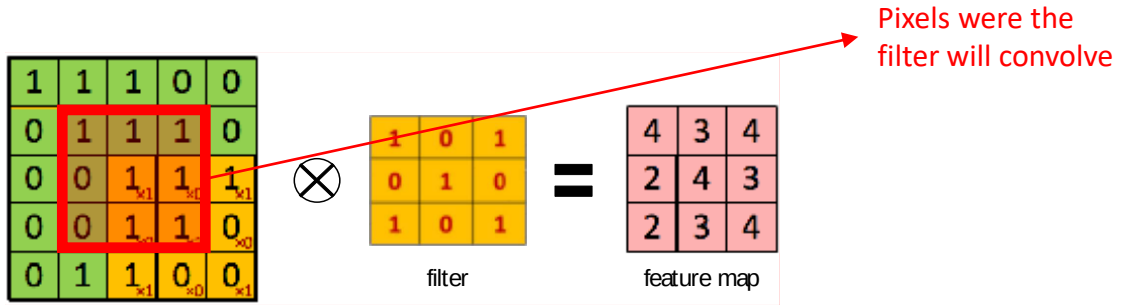
Convolutional Neural Networks

CNN (Convolutional Layer)

layers.**Conv2D**(filters, kernel_size, strides)

Layer dimension (on what depends)

- **Border effect**: When applying filters with size *n* x *n*, the output feature map will shrink some rows and columns in the borders, because of the positions where the filter will not fit.



- **Stride**: Number of pixels the filter will shift at convolution

universidad de león

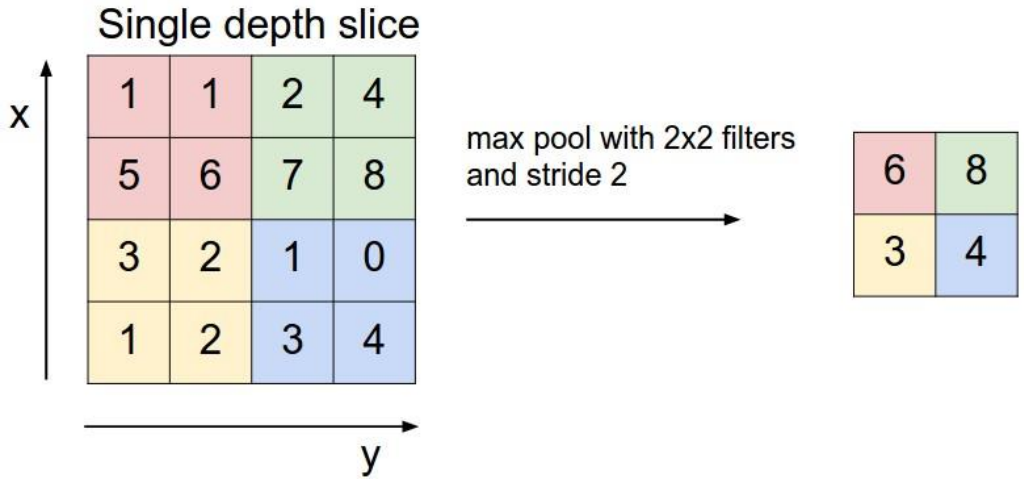
https://keras.io/api/layers/pooling_layers/

Convolutional Neural Networks

CNN (Pooling layers)

layers.MaxPooling2D()

- Common practice to insert a **pooling layer** in-between successive convolutional layers.
- **Function**: Down sample the output data from the previous conv layer.
- Useful because:
  - Reduces the number of parameters and, therefore, computation.
  - Controls overfitting..
  - Induce spatial filter hierarchies by making successive convolution filters to look at increasingly large windows.
- Normally use **max pooling**: Max operation on 2x2 windows with stride 2



universidad de león

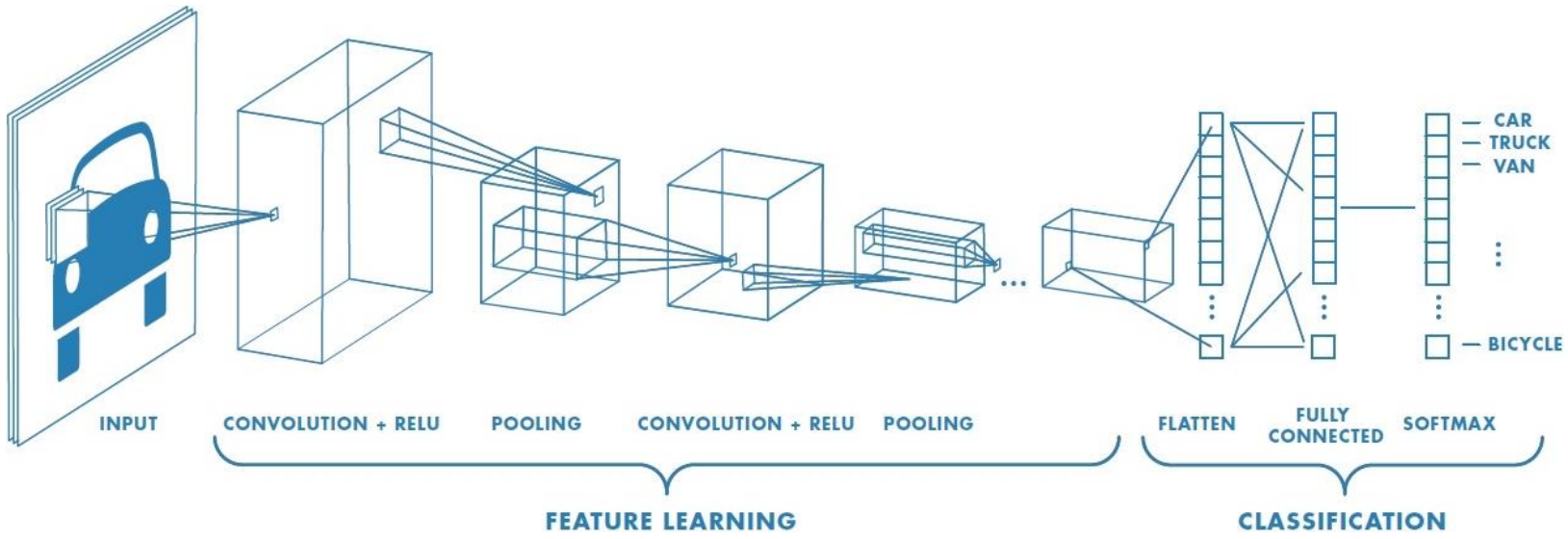
Convolutional Neural Networks

CNN (Fully Connected layers)

```
layers.Dense(128, activation='relu'),
layers.Dense(num_classes)
```

Finally, convolutional and pooling layers are connected to **fully connected layers** to make the final predictions.

Therefore, the **output** of the convolutions and down samplings is **flattened** (i.e., converted into a feature vector) and used as an input for fully connected layers, which will carry out the classification.



universidad de león

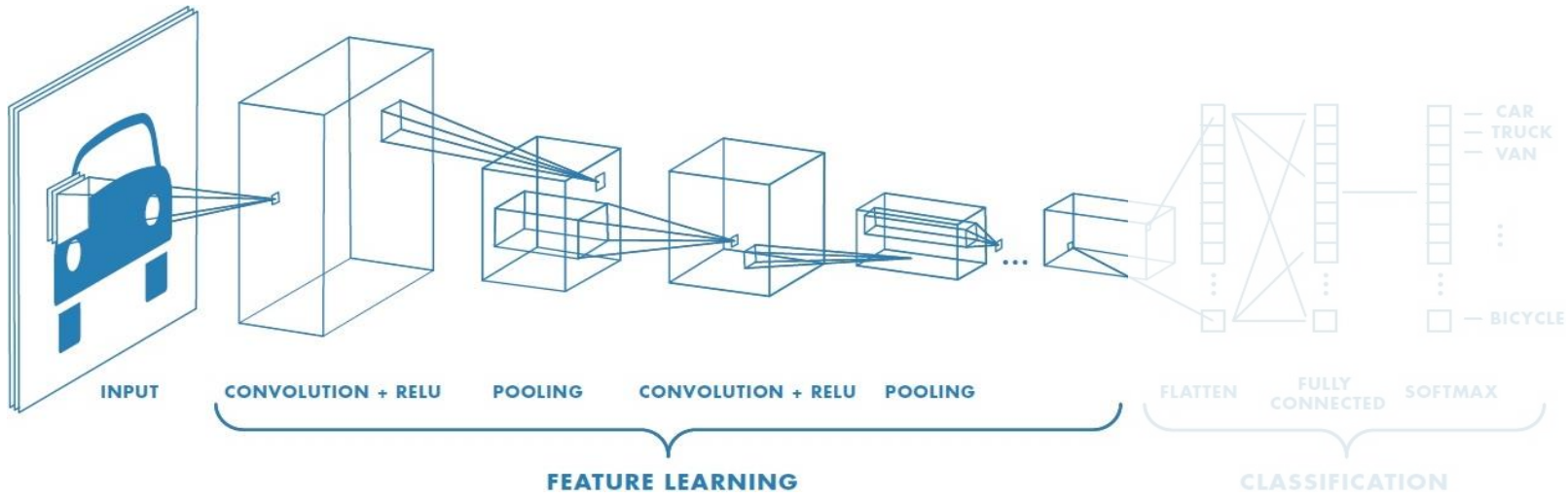
REGINNA 4.0

Convolutional Neural Networks

CNN (Summary)

In summary, given an input image, the **feature learning part** (a.k.a. Convolutional base):

1. Learns features in input image through **convolution**.
2. Introduce **non-linearity** through **activation function** (e.g. ReLU)
3. Reduce dimensionality and preserve spatial invariance with **pooling**.

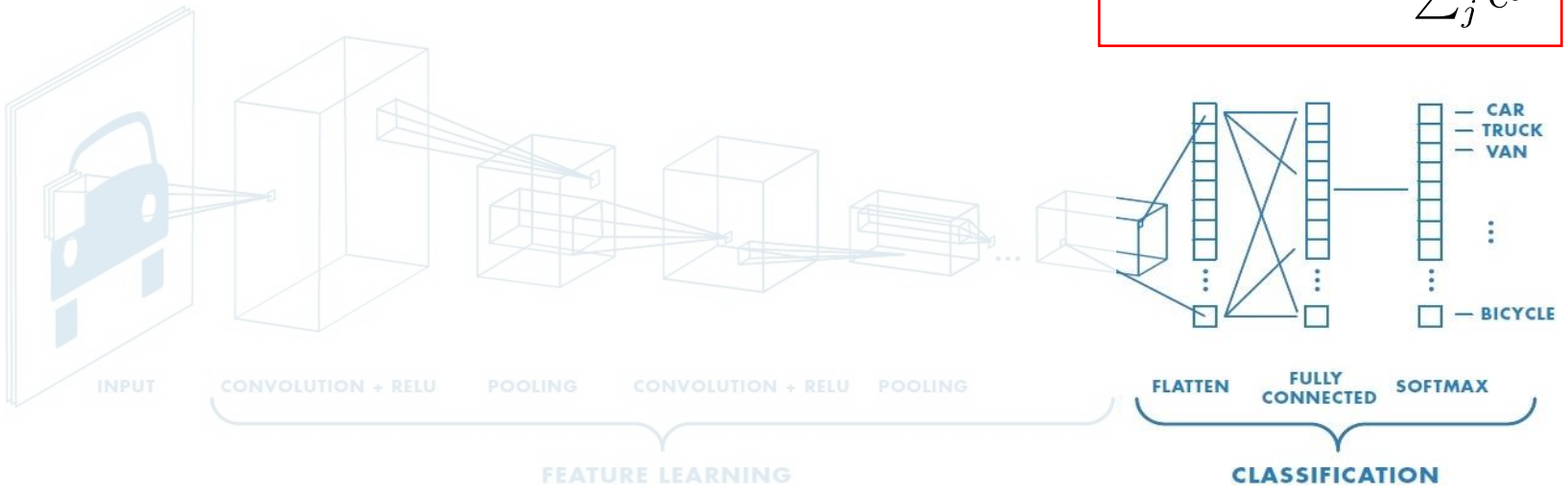


Convolutional Neural Networks

In the classification part:

- Features are flattened
- And given as an inputs to fully connected layers to make predictions

$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_i}}$$



universidad de león

Convolutional Neural Networks

Architectures

Lee-Net (LeCun et al. **1998**)

…

…

…

AlexNet (Krizhevsky et al. **2012**)

…

VGGNet (Simonyan and Zisserman, **2014**)

…

GoogLeNet (Szegedy et al. **2014**)

…

ResNet (He et al. **2015**)

…

FractalNet (Larsson et al. **2017**)

MobileNets (Howard et al. 2017)

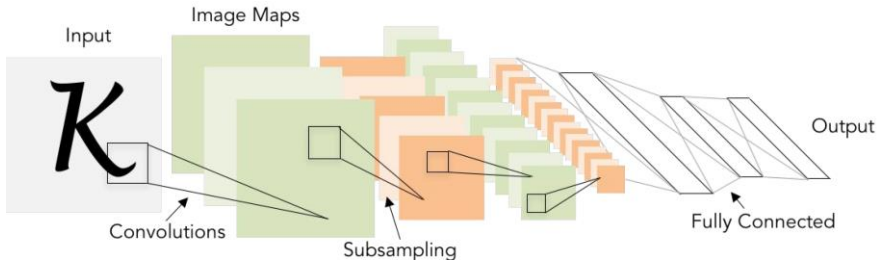
…

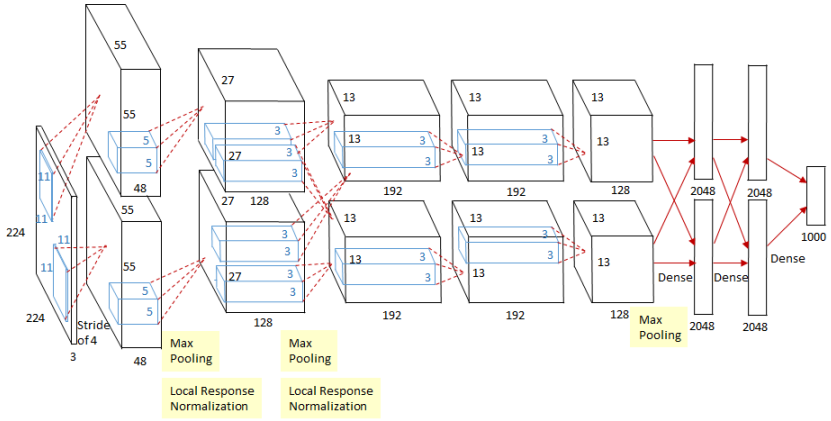
…

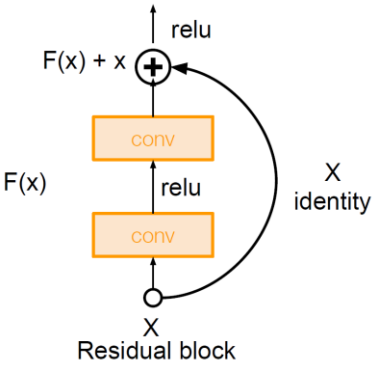
EfficientNet (Tan and Le, **2019**)

….

ConvNeXt (Liu et al. **2022**)







universidad de León

Table of contents

universidad de león

Convolutional Neural Networks

Issues

- Deep learning can find features in the training data on its own when an amount of data large enough is available. Especially in the case of high dimensional data.
- **What is "large enough"?** In the context of ConvNets, a few thousands (depending on number of classes, etc.) of images is considered a small dataset.
- *Case study: TOIC dataset. Images from Tor darknet*

universidad de león

Convolutional Neural Networks

Issues

Basic approaches

- Training a new model from scratch + **data augmentation**
- **Transfer Learning**: feature extraction with a pre-trained network
- **Fine-tuning** a pre-trained network

Convolutional Neural Networks

Training a new model from scratch + data augmentation

Working with a small amount of data: the network cannot properly learn all the possibilities -> Overfitting

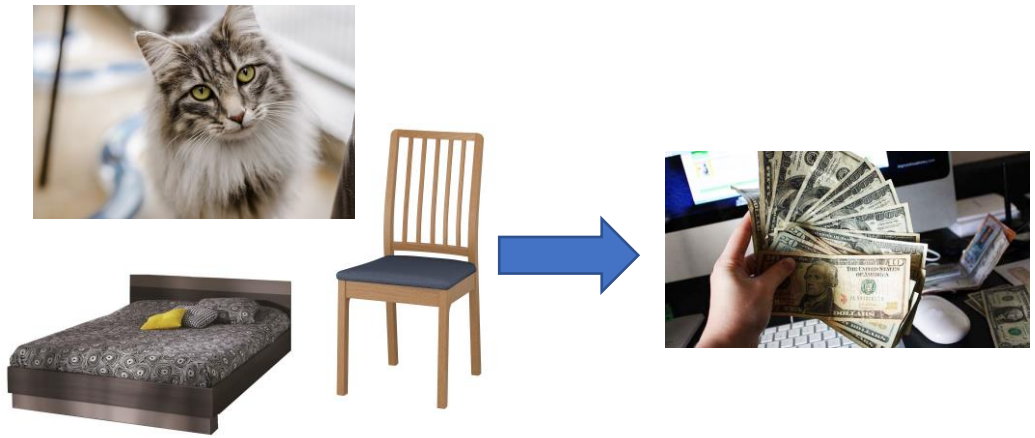
Data augmentation generates more training data from existing training samples.

Data is augmented via random transformations that yield believable-looking images. It helps expose the model to more aspects of the data and generalize better.

Convolutional Neural Networks

Transfer Learning: feature extraction with a pretrained network

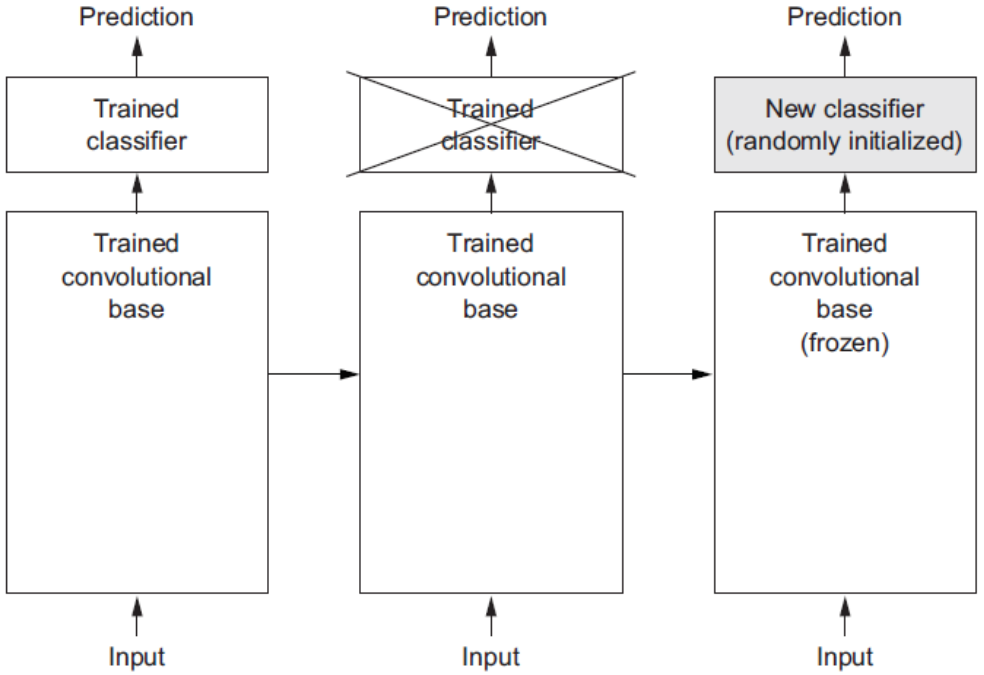
- **Pretrained network**: Saved network that was previously trained on a large dataset.
  - Typically, trained on a large image-classification task such as ImageNet (1.4M images, 1000 classes: animals, objects, etc.).
- Such general network **can be repurposed for different problems**.
- Pretrained networks publicly available (e.g. in the module `keras.applications`).
- Examples:
  - Xception.
  - **VGG16**
  - VGG19
  - ResNet50
  - InceptionV3
  - …

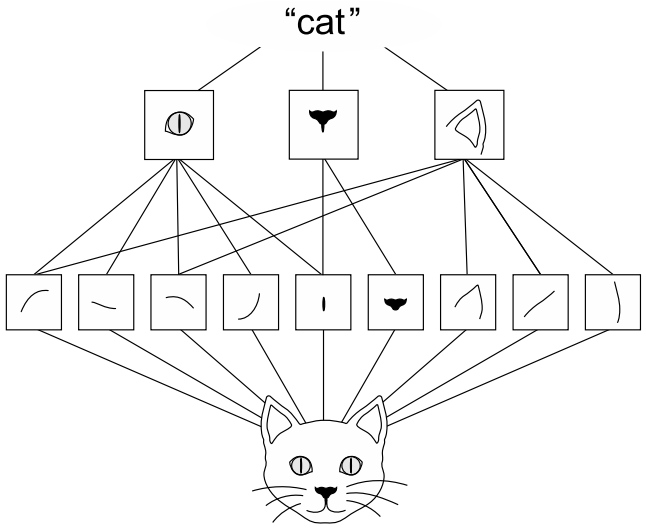


Convolutional Neural Networks

Transfer Learning: feature extraction with a pretrained network

- Consists of using a pretrained network to extract features from new images using its *convolutional base*.
- These features are then run through a new classifier, which is trained from scratch.
- Alternative: Add fully connected layers or freeze the convolutional base



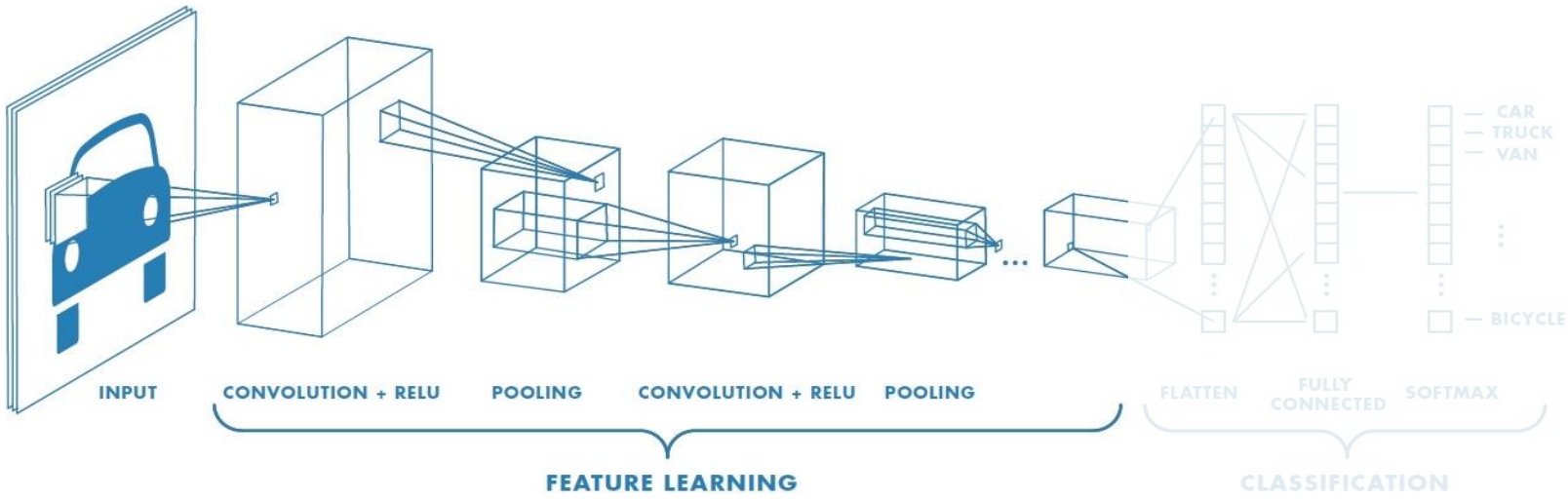


universidad de león

Convolutional Neural Networks

Transfer Learning: feature extraction with a pretrained network

- Consists of using a pretrained network to extract features from new images using its *convolutional base*.
- These features are then run through a new classifier, which is trained from scratch.
- Alternative: Add fully connected layers or freeze the convolutional base

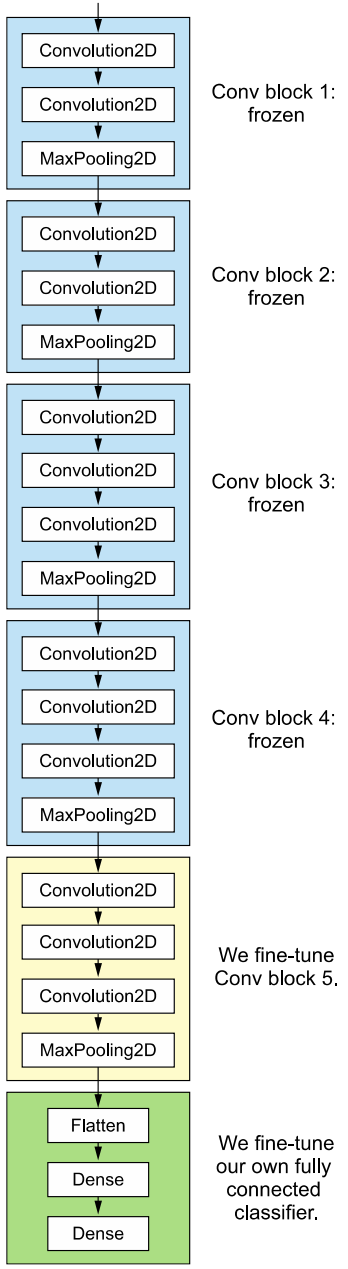


universidad de león

Convolutional Neural Networks

Fine-tuning a pre-trained network

- Consists on freezing the convolutional base except a few layers on top of it, and jointly training this non-frozen part and the fully connected layers added on top of it.
- Only fine-tune the top layers of the convolutional base once the classifier on top has been trained.
- Steps:
  1. Add dense network on top of an pretrained network.
  2. Freeze the convolutional base.
  3. Train the added part.
  4. Unfreeze some top layers in the convolutional base.
  5. Jointly train both these layers and the Dense layers added before.

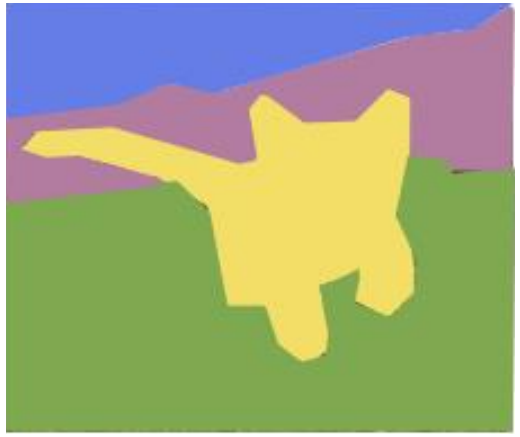


universidad de león

Some applications

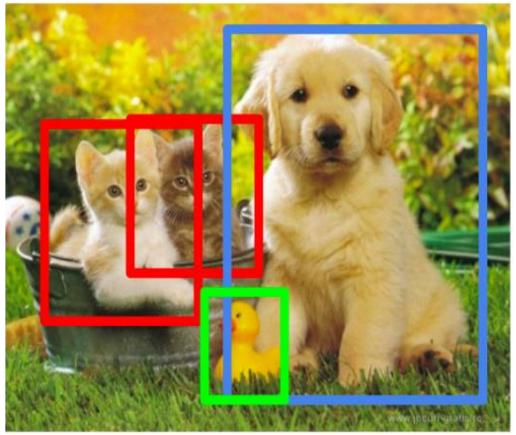
Beyond classification

Semantic Segmentation



CAT

Object Detection



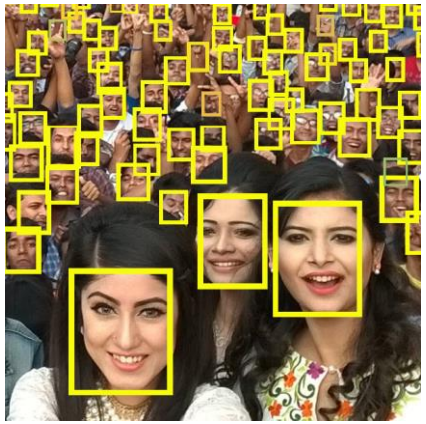
CAT, DOG, DUCK

Image Captioning

The cat is in the grass.

Instance Segmentation

CAT, DOG, DUCK



Cybersecurity

Botnet Detection

Spam detection & classification

Malware detection

Ransomware Detection

Phishing

Cyberattacks

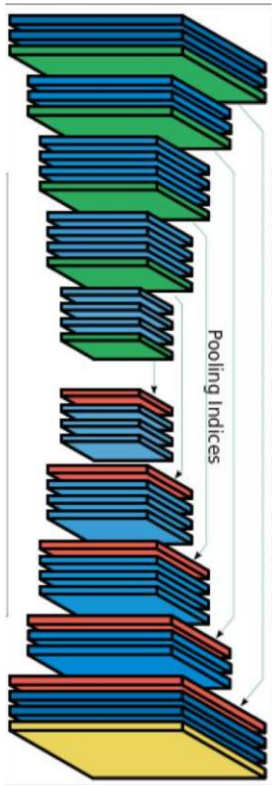
Fraudulent website detection

Adult Porn Detection (or CSEM)

universidad de león

REGINNA 4.0

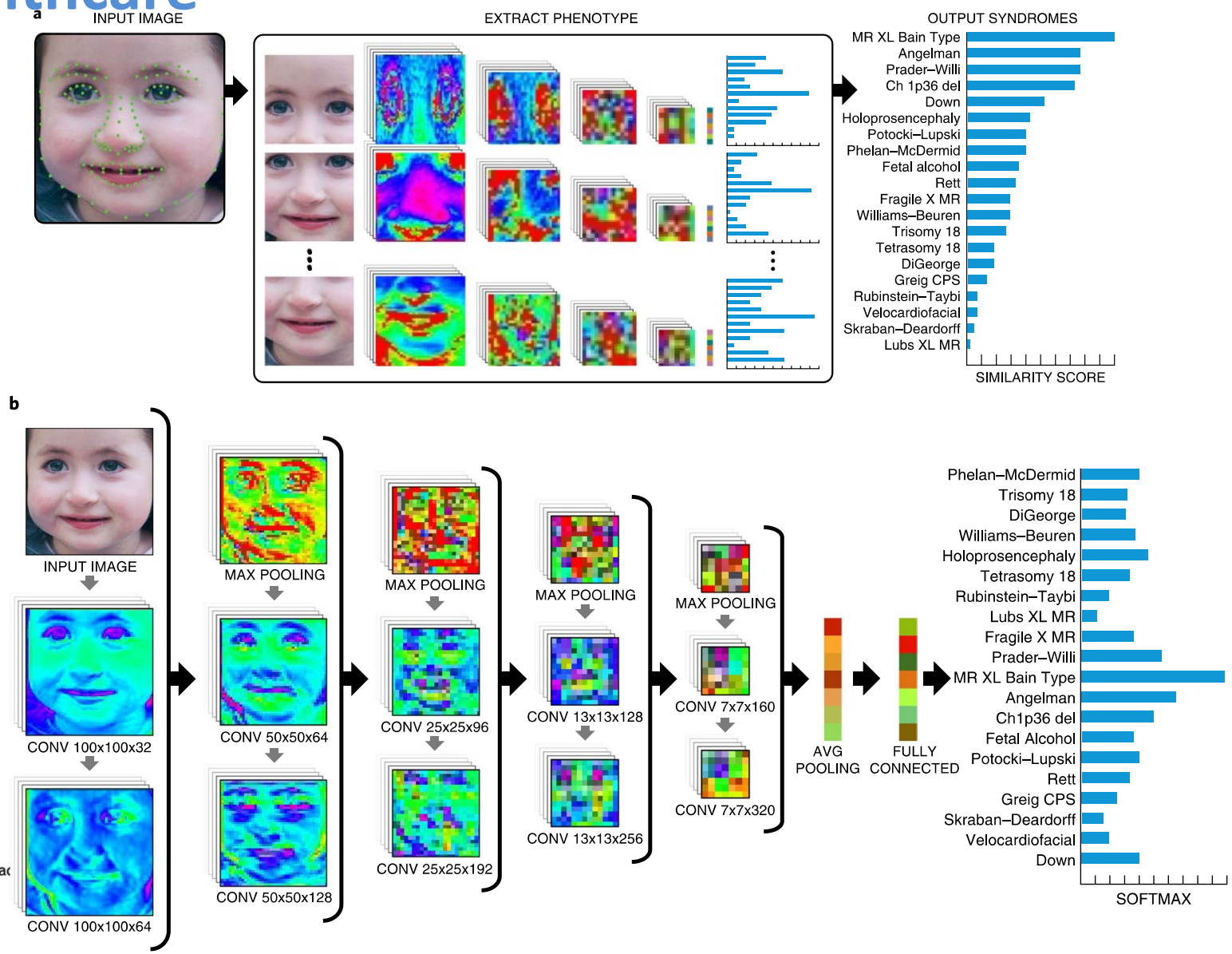
Self driving cars



universidad de león

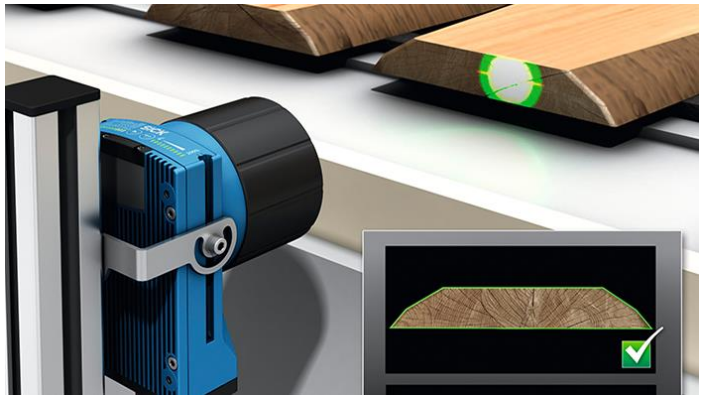
REGINNA 4.0

Healthcare



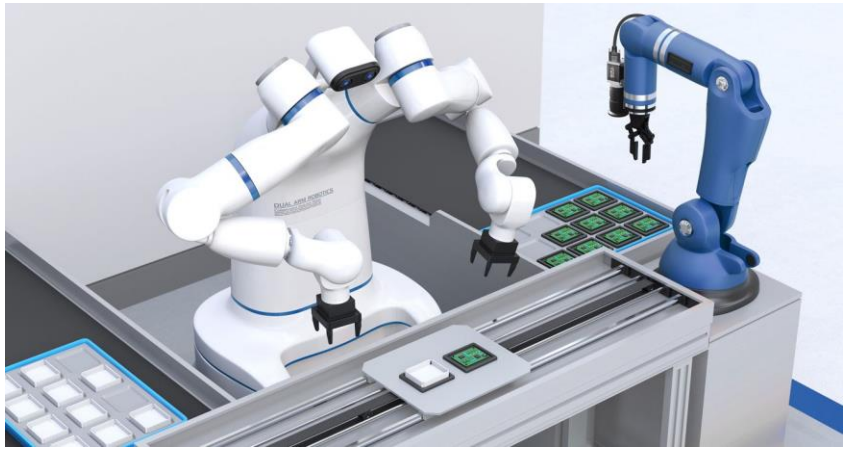
universidad de león

Industry 4.0



Inspection

Indoor augmented reality



Robotics

Figure from https://www.sick.com/at/en/deep-learning-as-motor-for-industry-40/w/blog-deep-learning/
https://insidernavigation.com/ar-indoor-navigation/
https://tanhungha.com.vn/ung-dung-cua-camera-vision-cho-vision-guided-robotics-n290.html
https://www.anybotics.com/computer-vision-and-synthetic-data-are-key-to-training-autonomous-robots/

universidad de león

REGINNA 4.0

Hands on: Automatic classification of inserts using image classification with CNN

REGINNA 4.0

Eduardo Fidalgo Fernández
Universidad de León (Spain)
eduardo.fidalgo@unileon.es

Supported by
eit Manufacturing
Funded by the European Union
INNOVATION CAPACITY BUILDING FOR HIGHER EDUCATION
PART OF EIT HEI Initiative
eit-hei.eu

www.reginna4-0.eu