
ML
SUMMIT
BILDKLASSIFIKATION LEICHT GEMACHT
– MIT KERAS UND TENSORFLOW
Dr. Shirin Glander
Data Scientist @ codecentric AG

About this Workshop

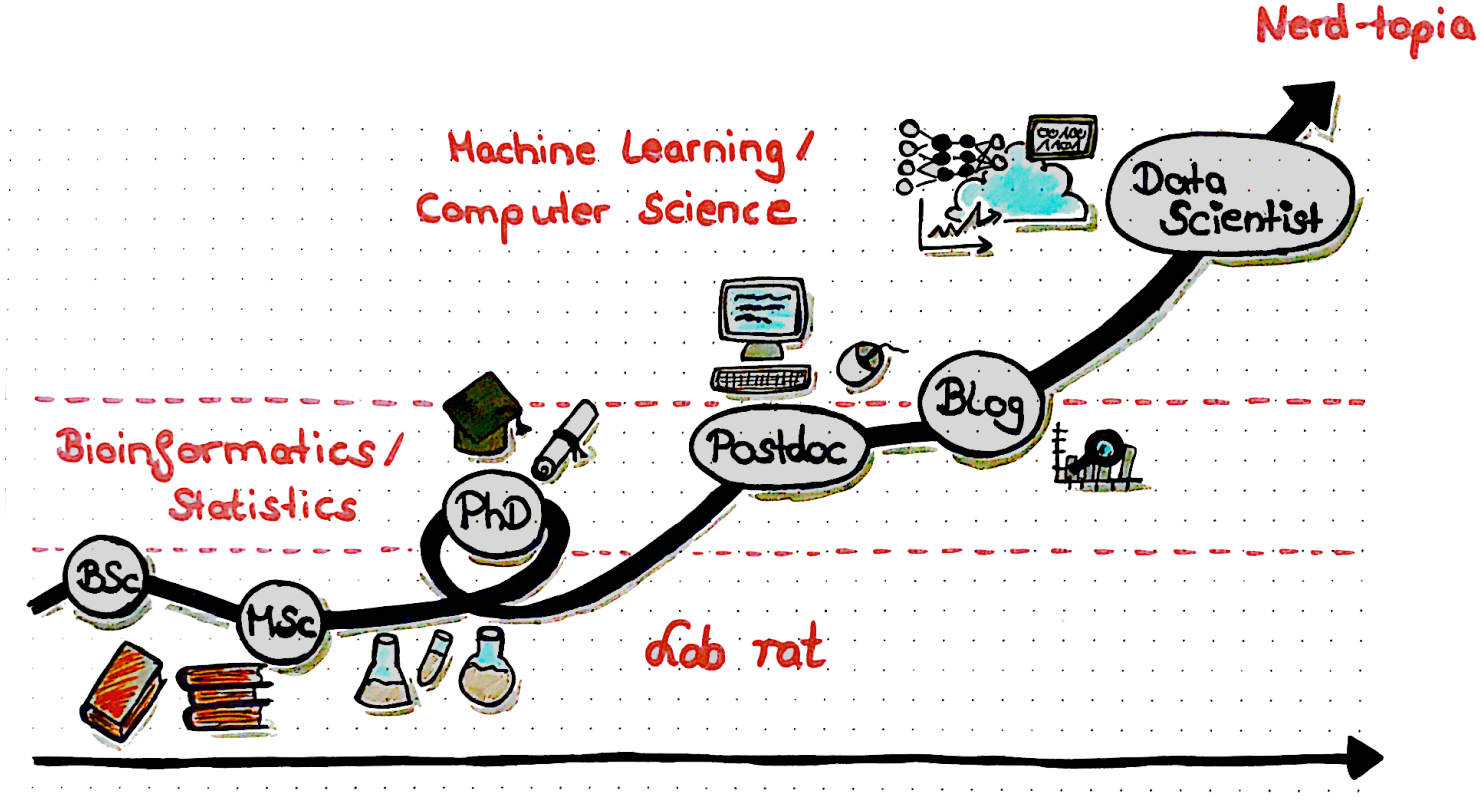
- Learn what neural nets are and how they can be used in computer vision.
- Understand how a computer learns to "see".
- Apply pre-trained nets and modify them.
- Build a model from scratch that differentiates between fruits on images.

- Visualise convolutions and layers.

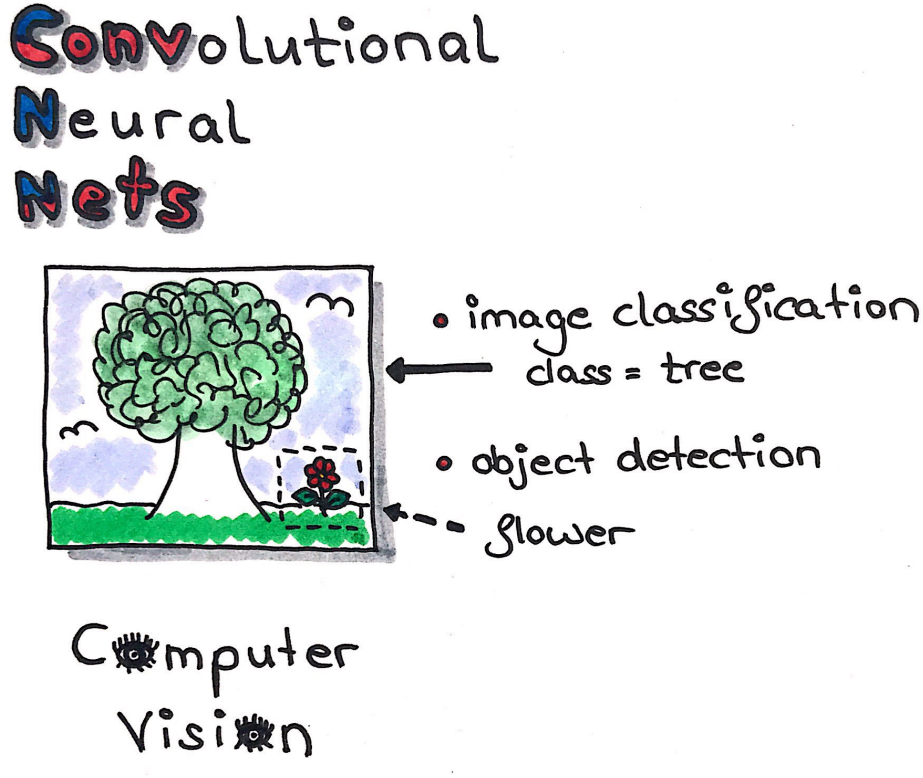
- (Bonus: Explaining our classifications with LIME)

Material on Github: https://github.com/ShirinG/image_classification_keras_tf (https://github.com/ShirinG/image_classification_keras_tf)

About me

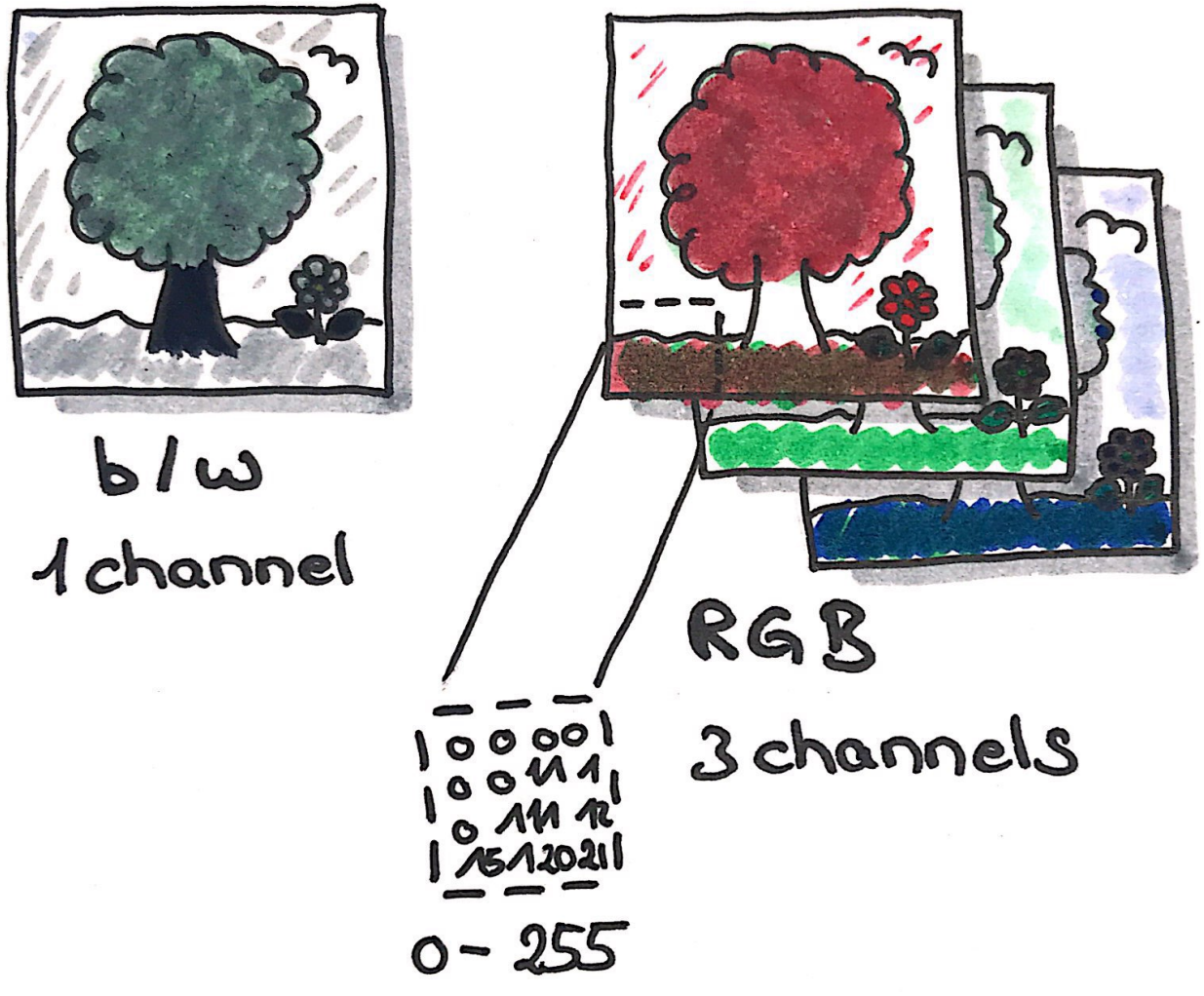

Nerd-topia
Machine Learning /
Computer Science
Data Scientist
Blog
Postdoc
PhD
MSc
BSc
Bioinformatics /
Statistics
Lab rat

How does a computer learn to "see"?

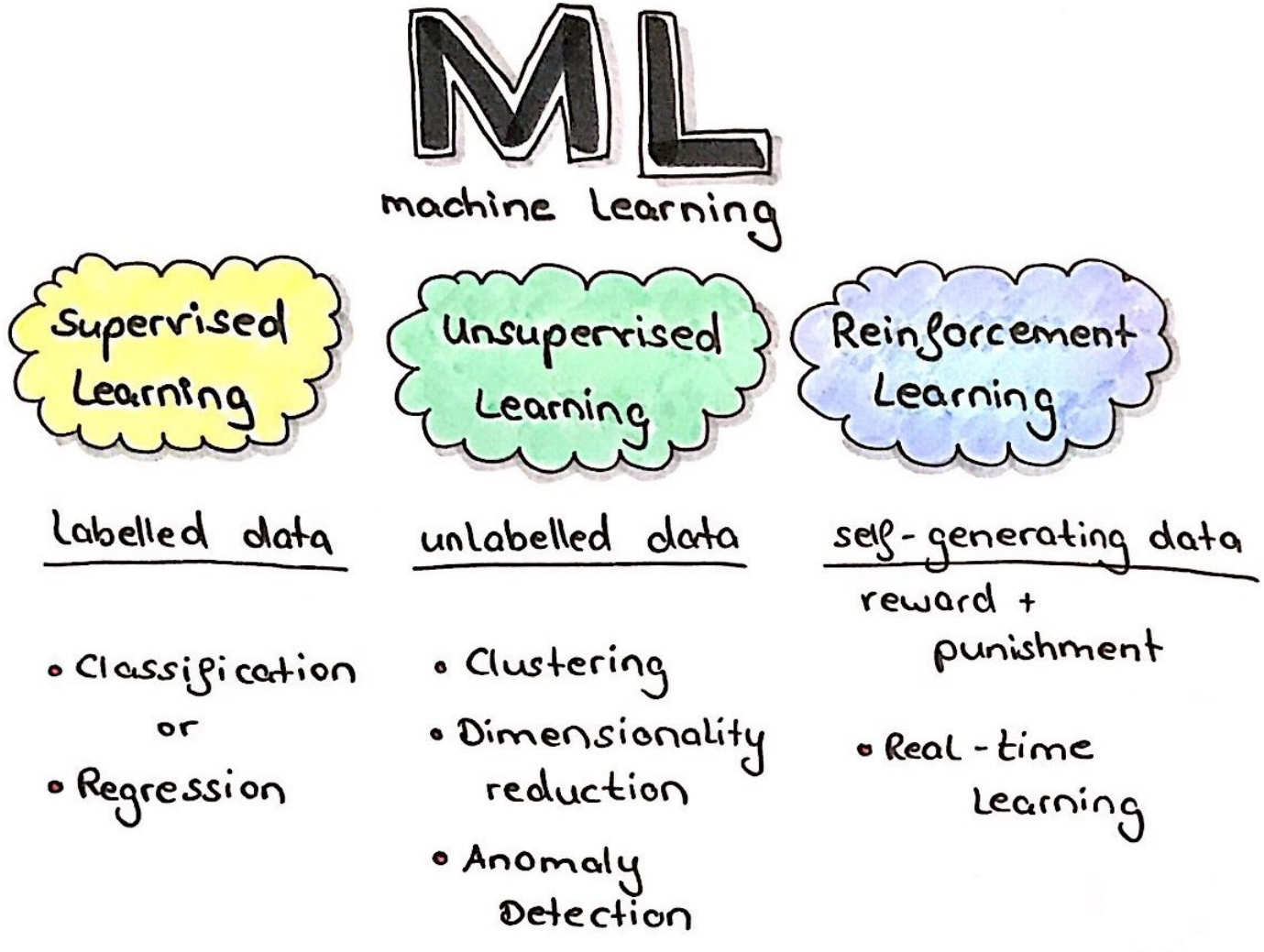


See also this video from codecentric.AI (https://youtu.be/h7eQrBbM45Q) for more on Computer Vision Basics!

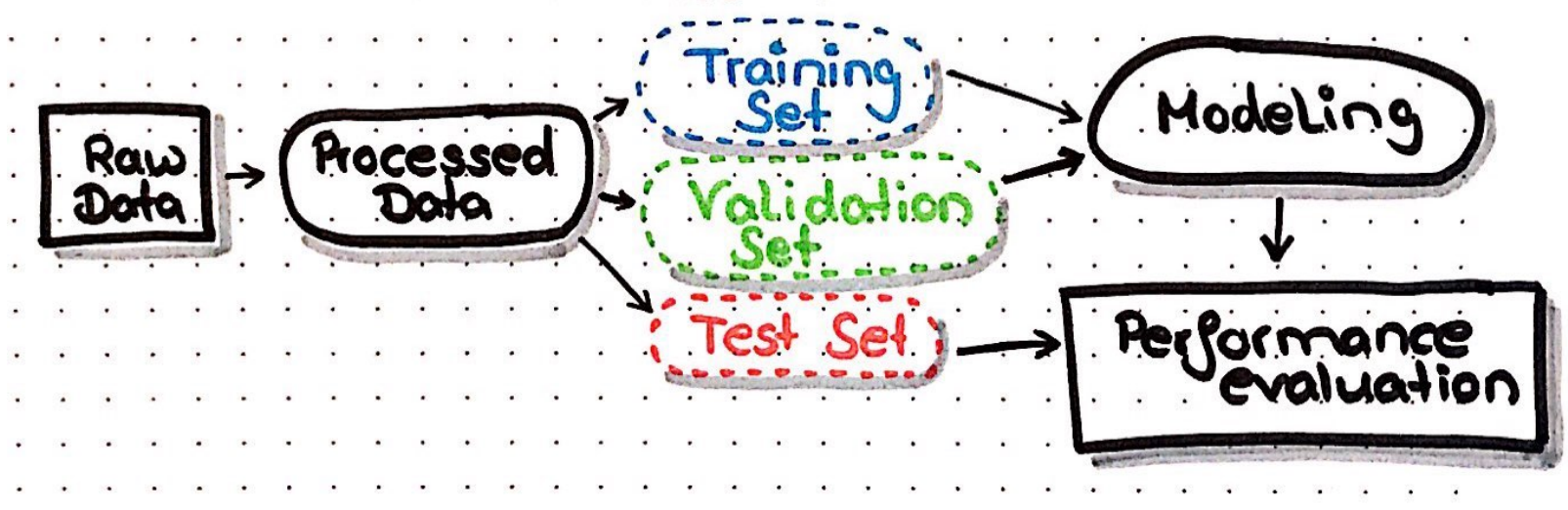
How does a computer learn to "see"?



What is Machine Learning (ML)

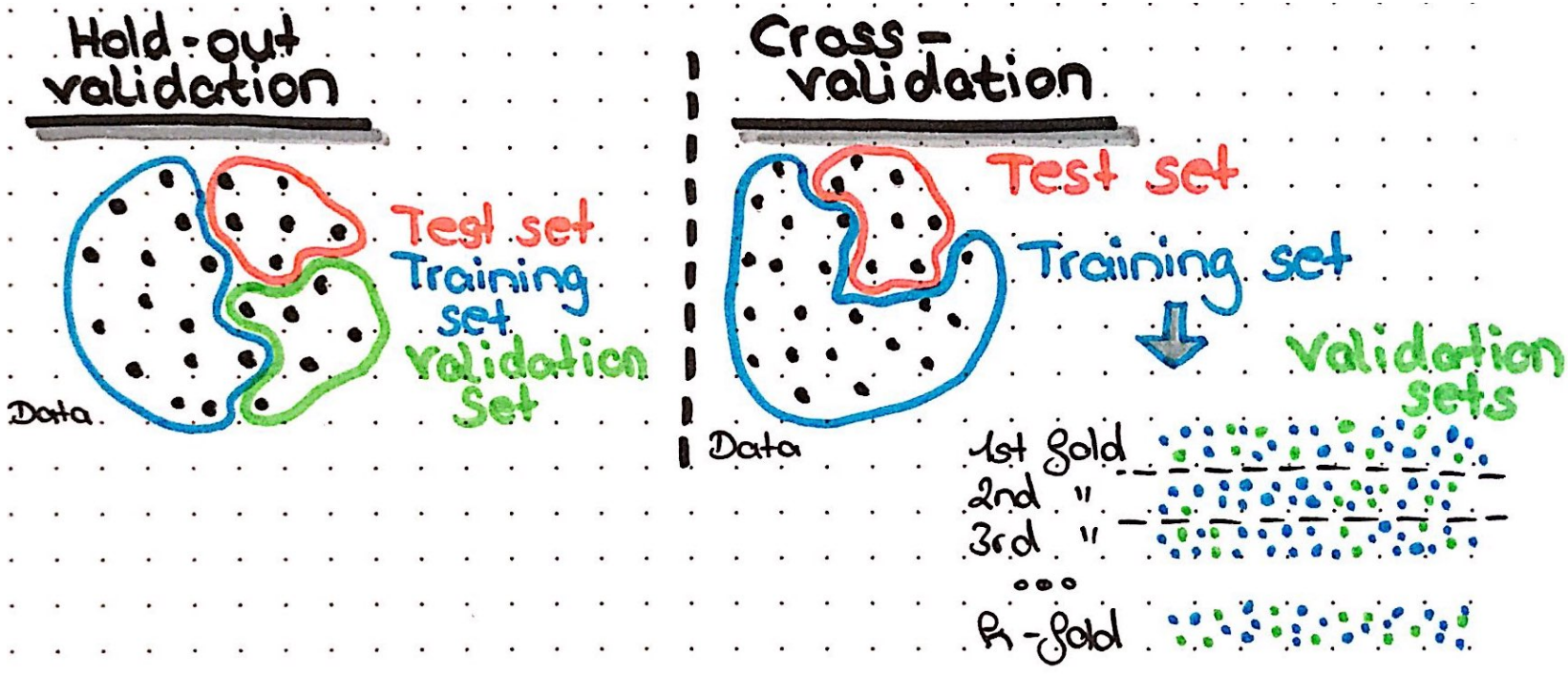


A typical ML Workflow

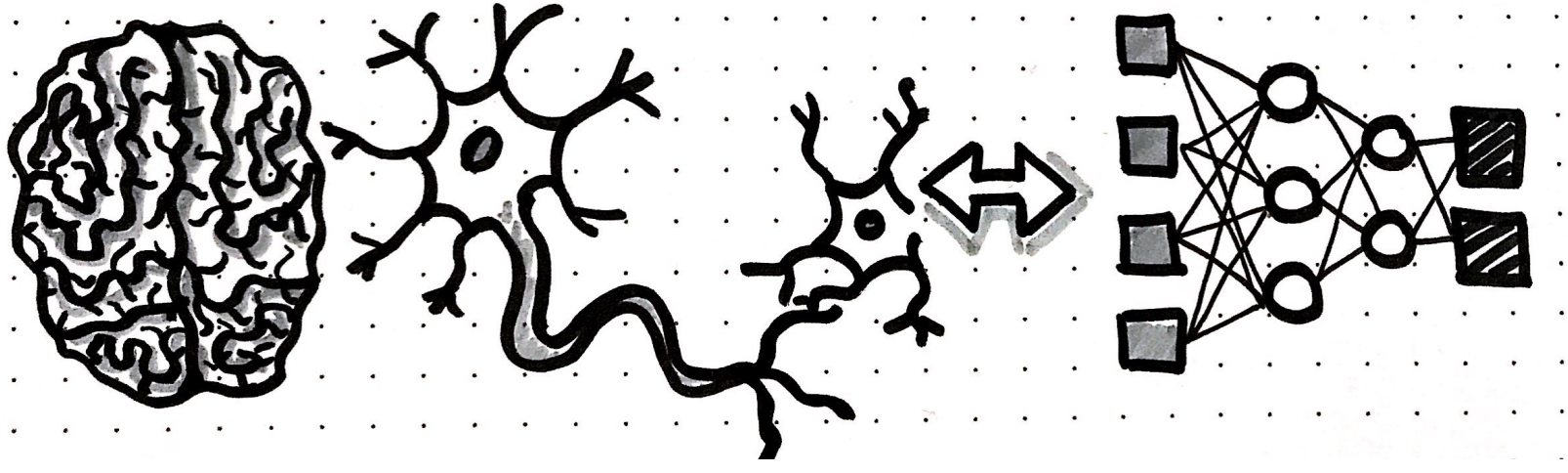


Why use validation and test data?

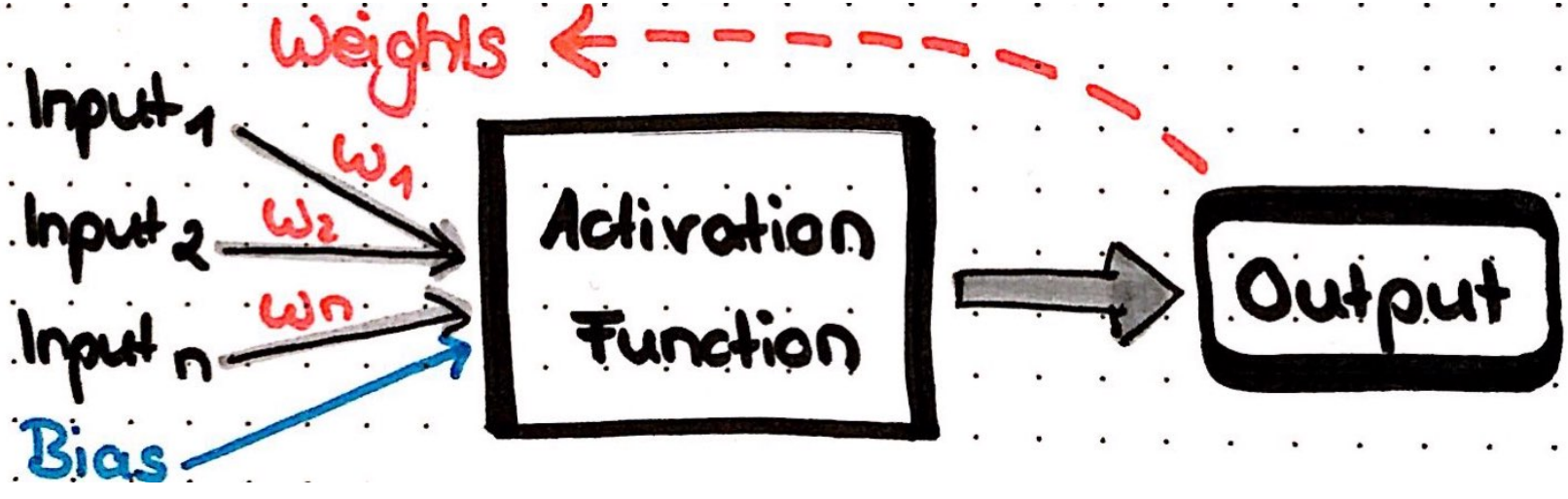
Validation methods



What are neural nets?

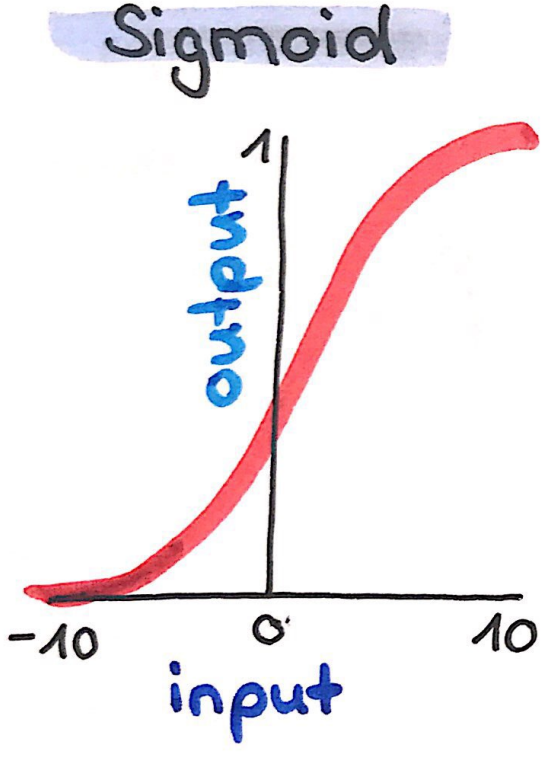


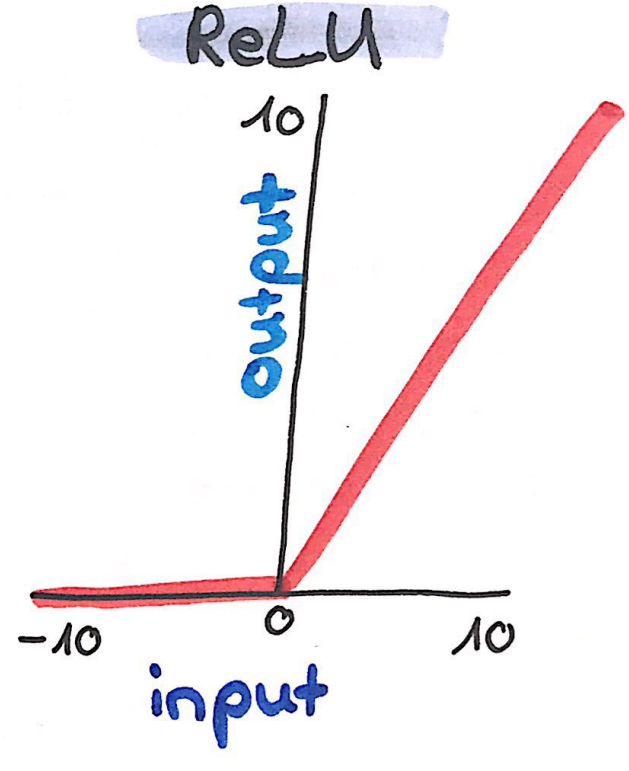
Perceptrons



Activation functions normalise input

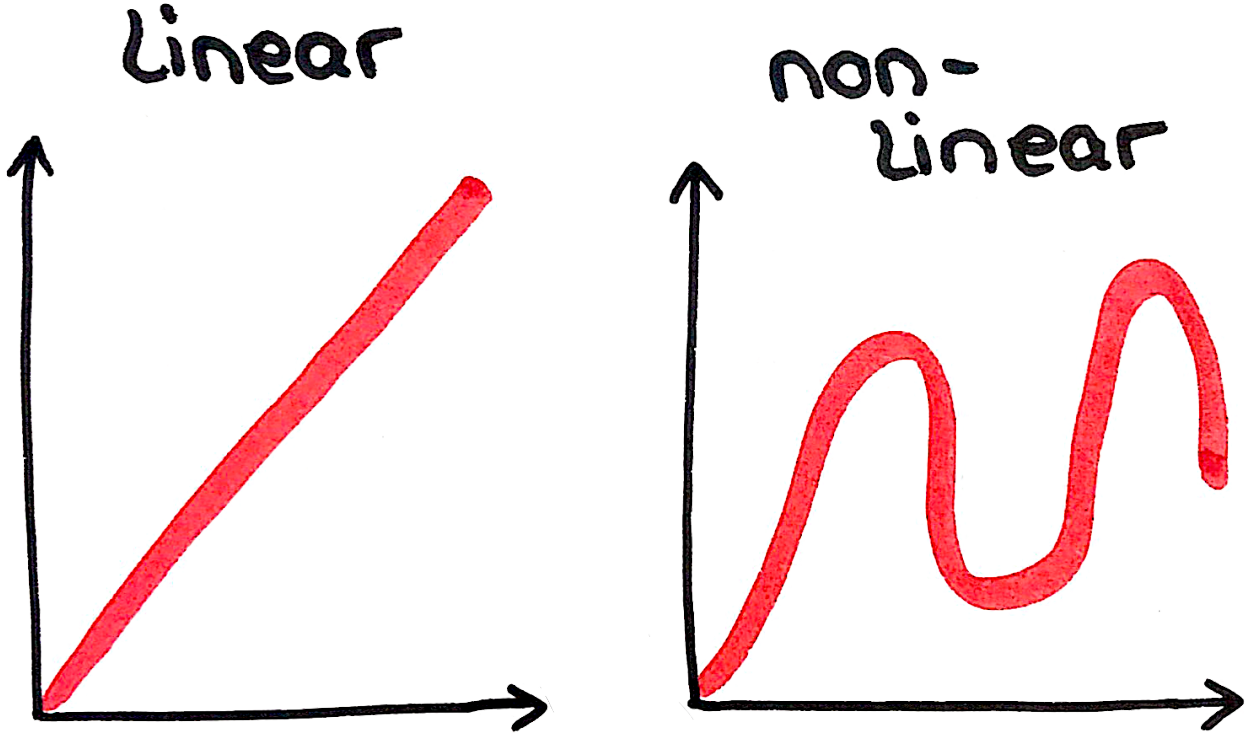
- every problem can be described with a mathematical function



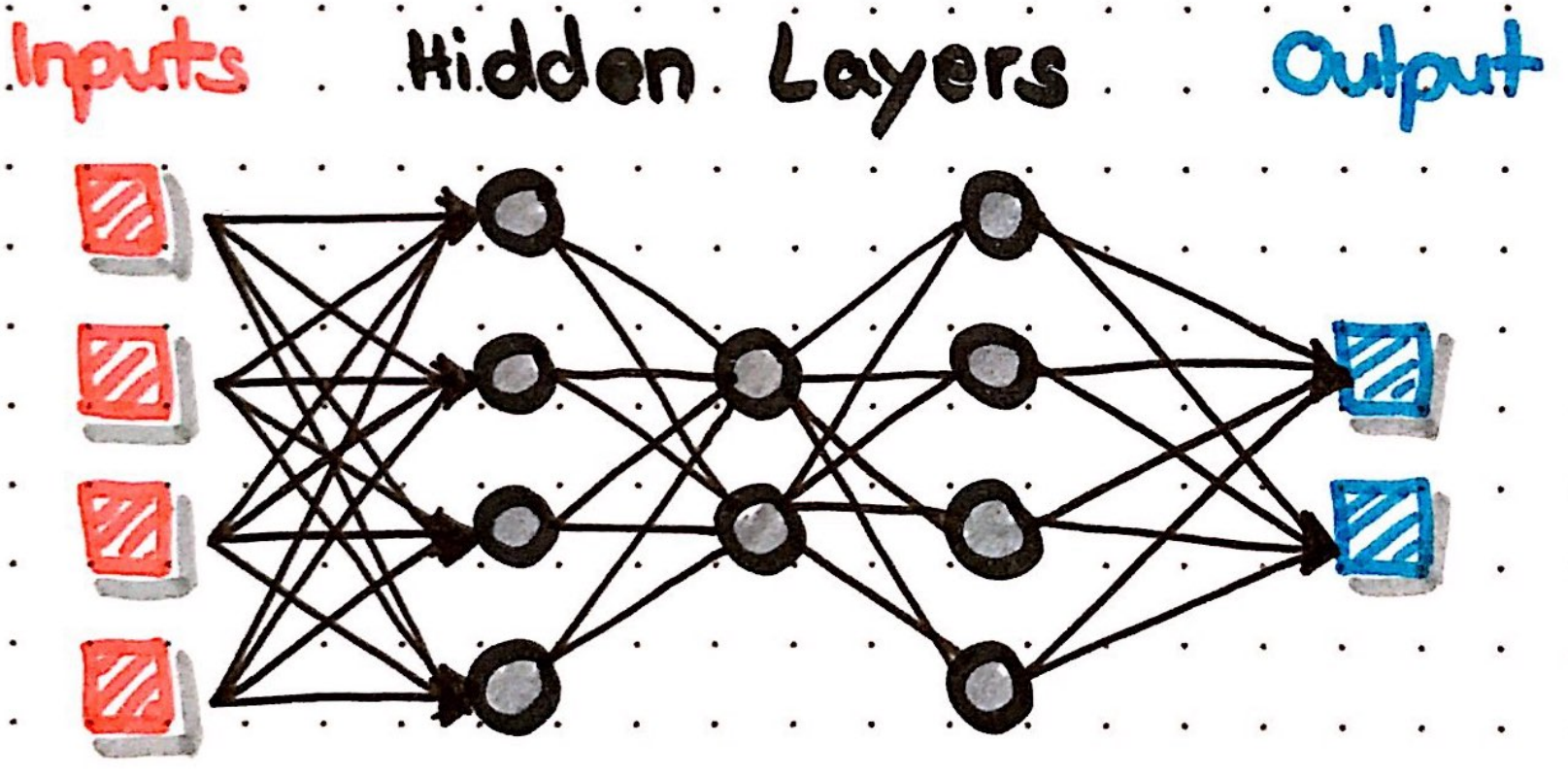


Activation functions make non-linearity possible

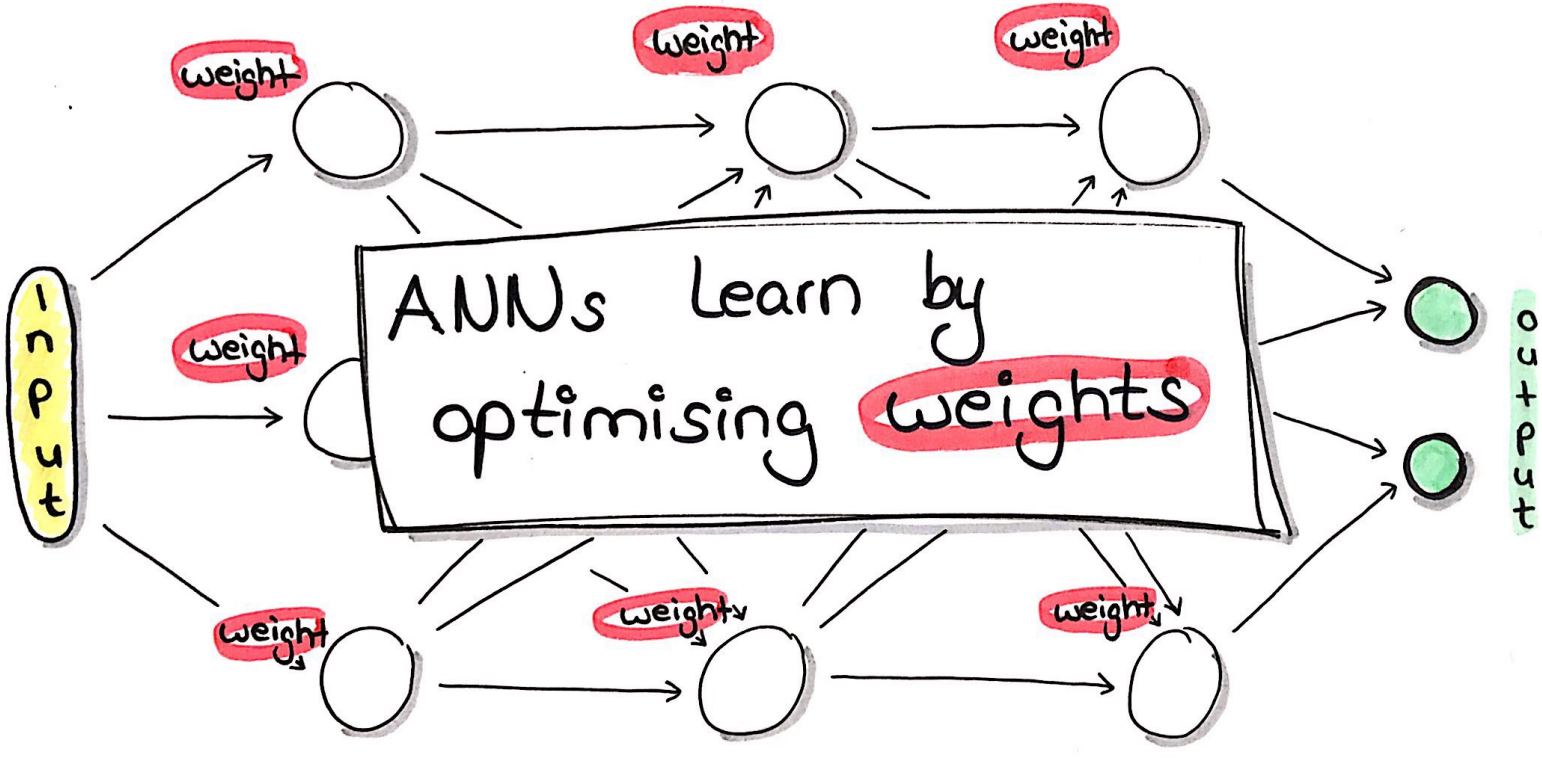
- non-linear activation functions allow us to approximate ANY mathematical formula with neural nets



Multi-Layer Perceptrons

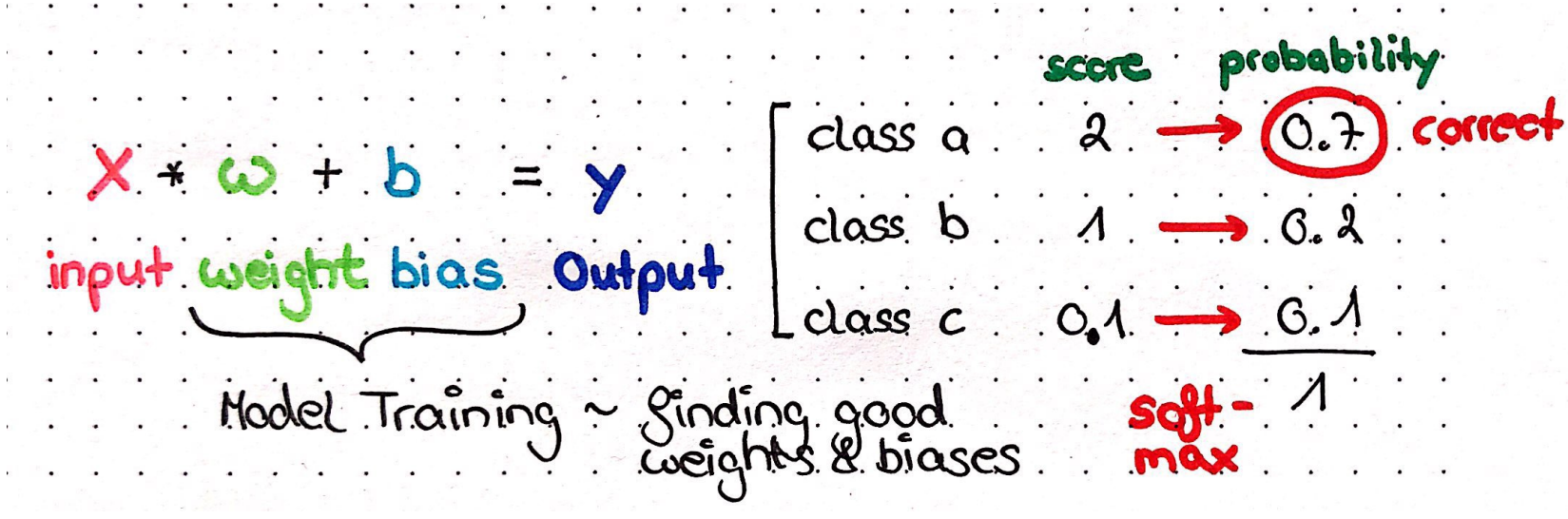


How does a neural net learn?



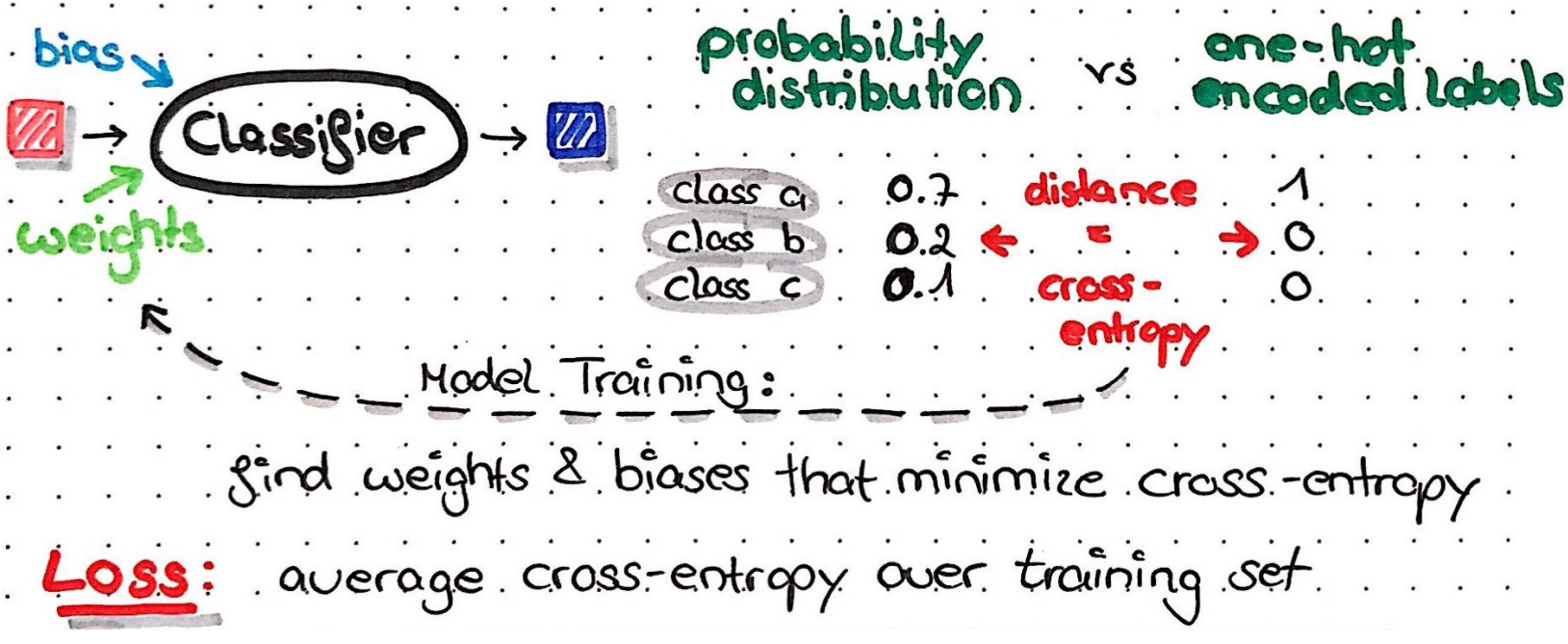
How does a neural net learn?

The Softmax function



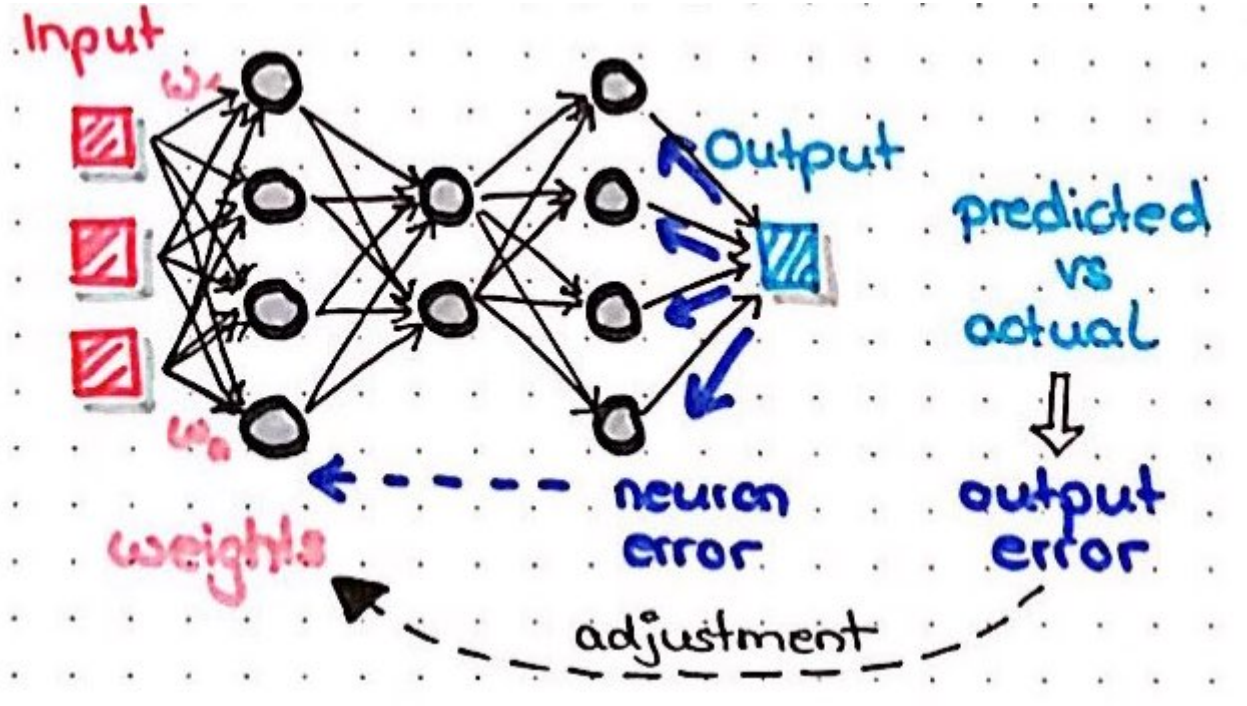
How does a neural net learn?

Cross-entropy



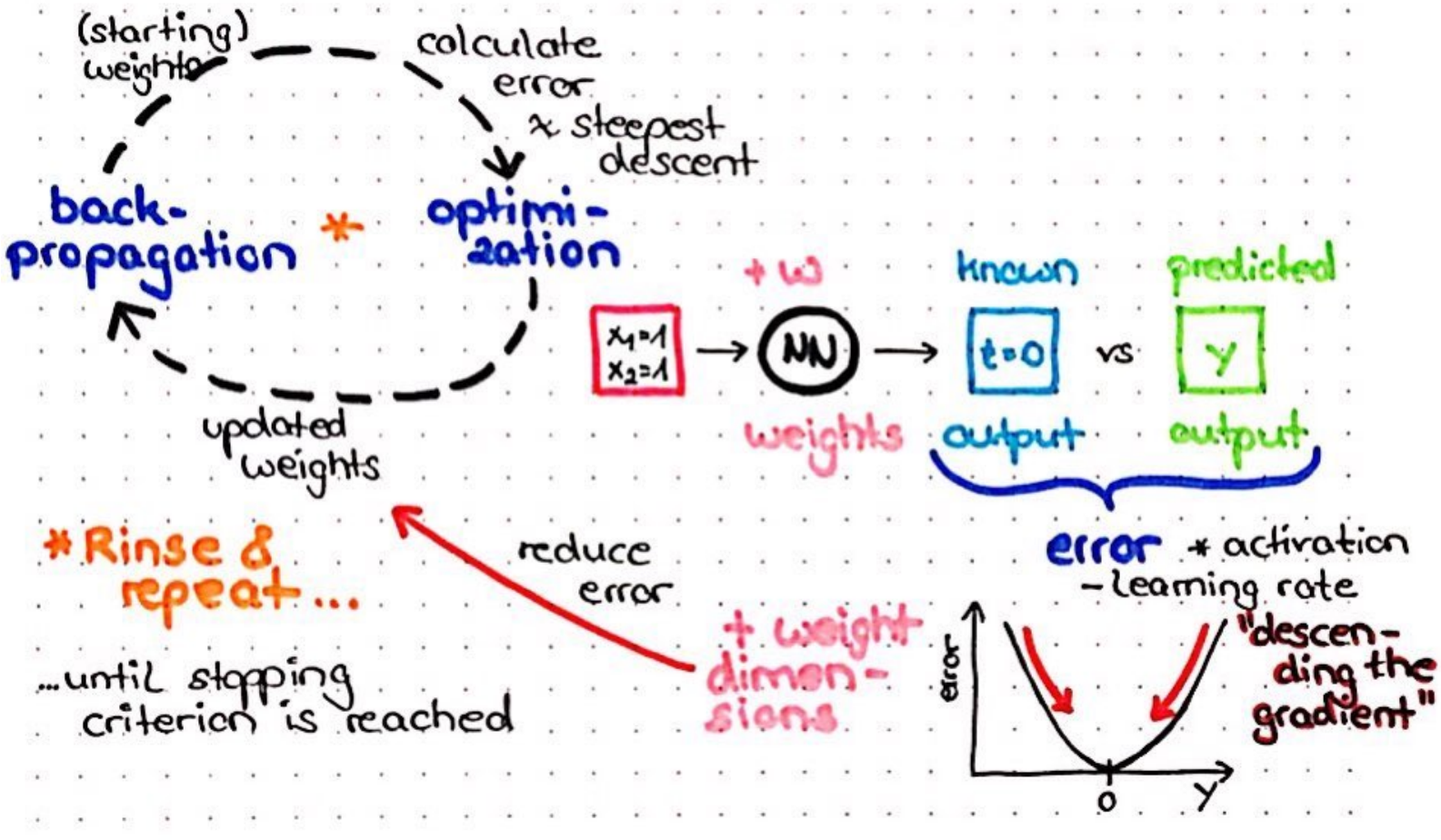
How does a neural net learn?

Backpropagation

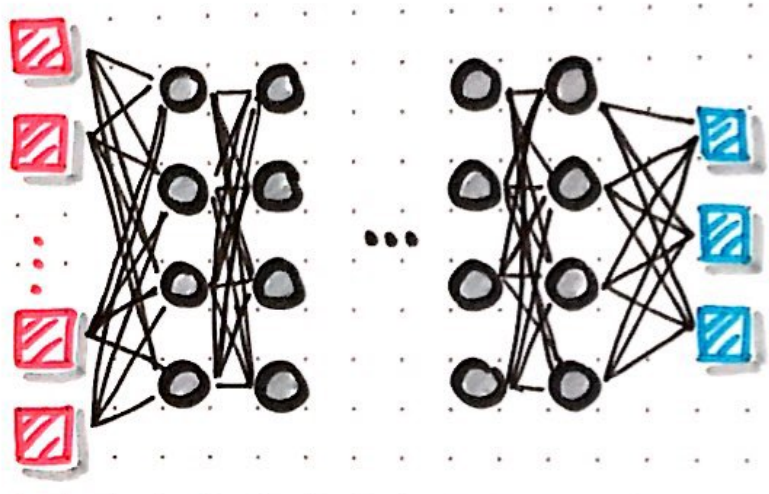


How does a neural net learn?

Gradient descent optimization



Deep Learning

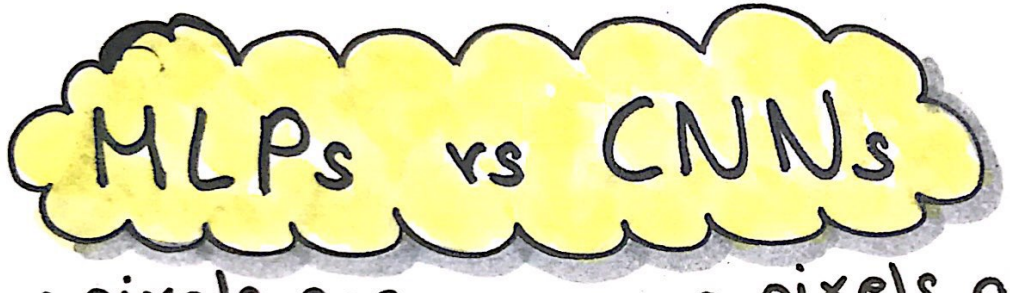


- supervised, semi-supervised or unsupervised

- NLP, speech recognition
- image recognition
- object classification
- recommender systems

etc.

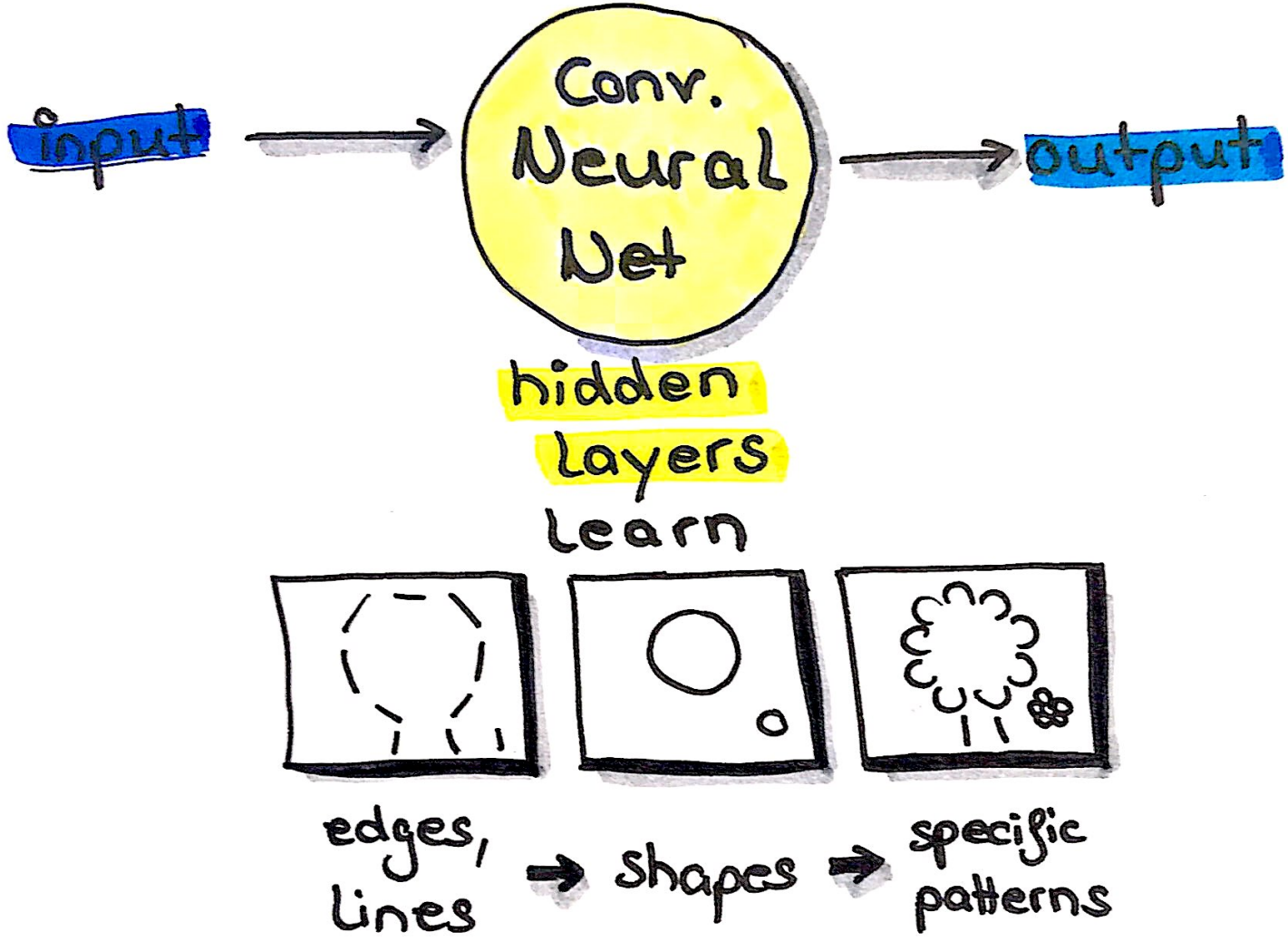
Convolutional Neural Nets



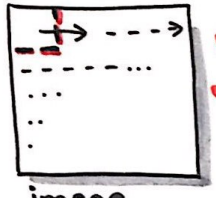
- pixels are considered independent
- computationally faster
- Learned : weights

- pixels are considered as groups of connected information (context)
- analysed as chunks (= windows)
- Learned : filters

ConvNets


input
Conv.
Neural
Net
output
hidden
layers
learn
edges,
lines
shapes
specific
patterns

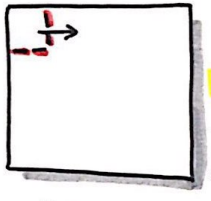
ConvNets



sliding window

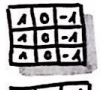
↳ window size (e.g. 3x3 pixels)

image

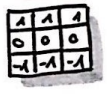


apply filters to every sliding window

e.g.



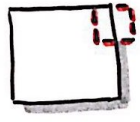
vertical lines



horizontal lines

- **Filters**
  detect shapes & patterns in window chunks
- multiple filters are combined
- filters are learned

padding



- fake pixels are created at the edge of images to incorporate border pixels

stride

- how much overlap to have in sliding window

Convolutional Layers

apply filters

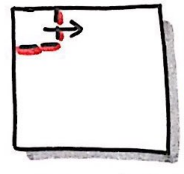
output

=

feature maps

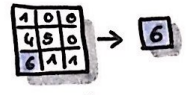
stacks of

feature maps



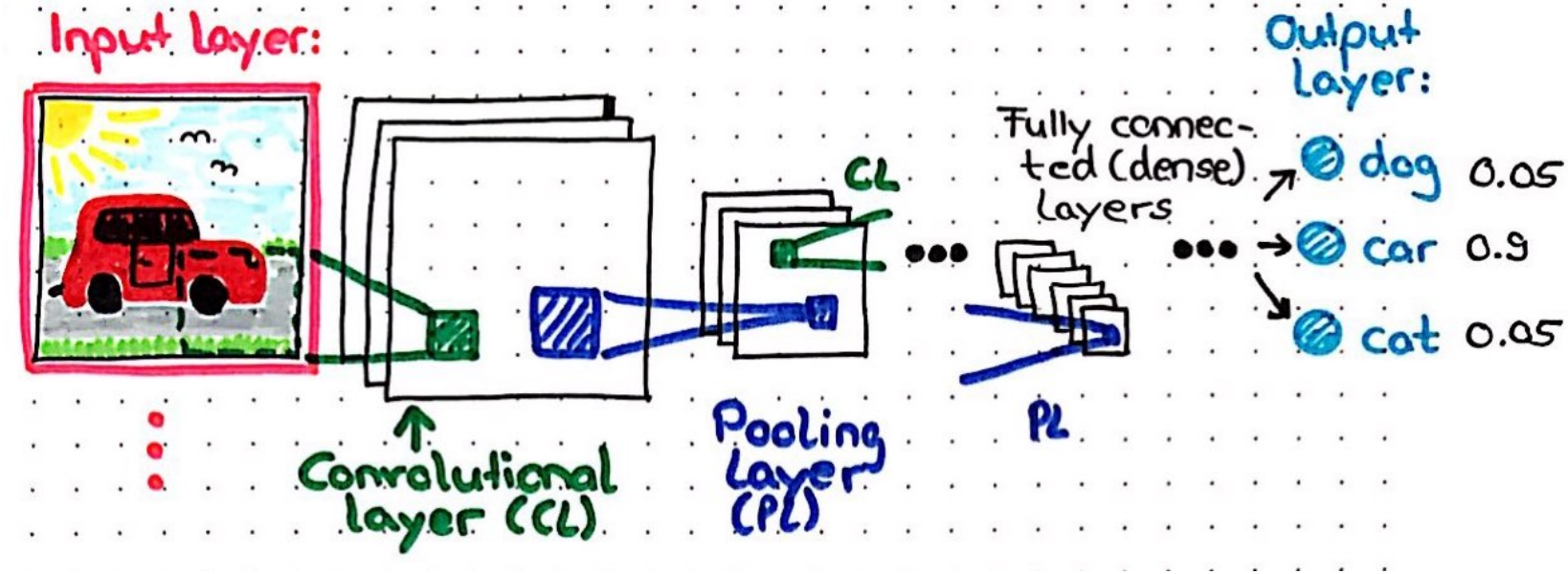
Pooling Layers

e.g. max pooling

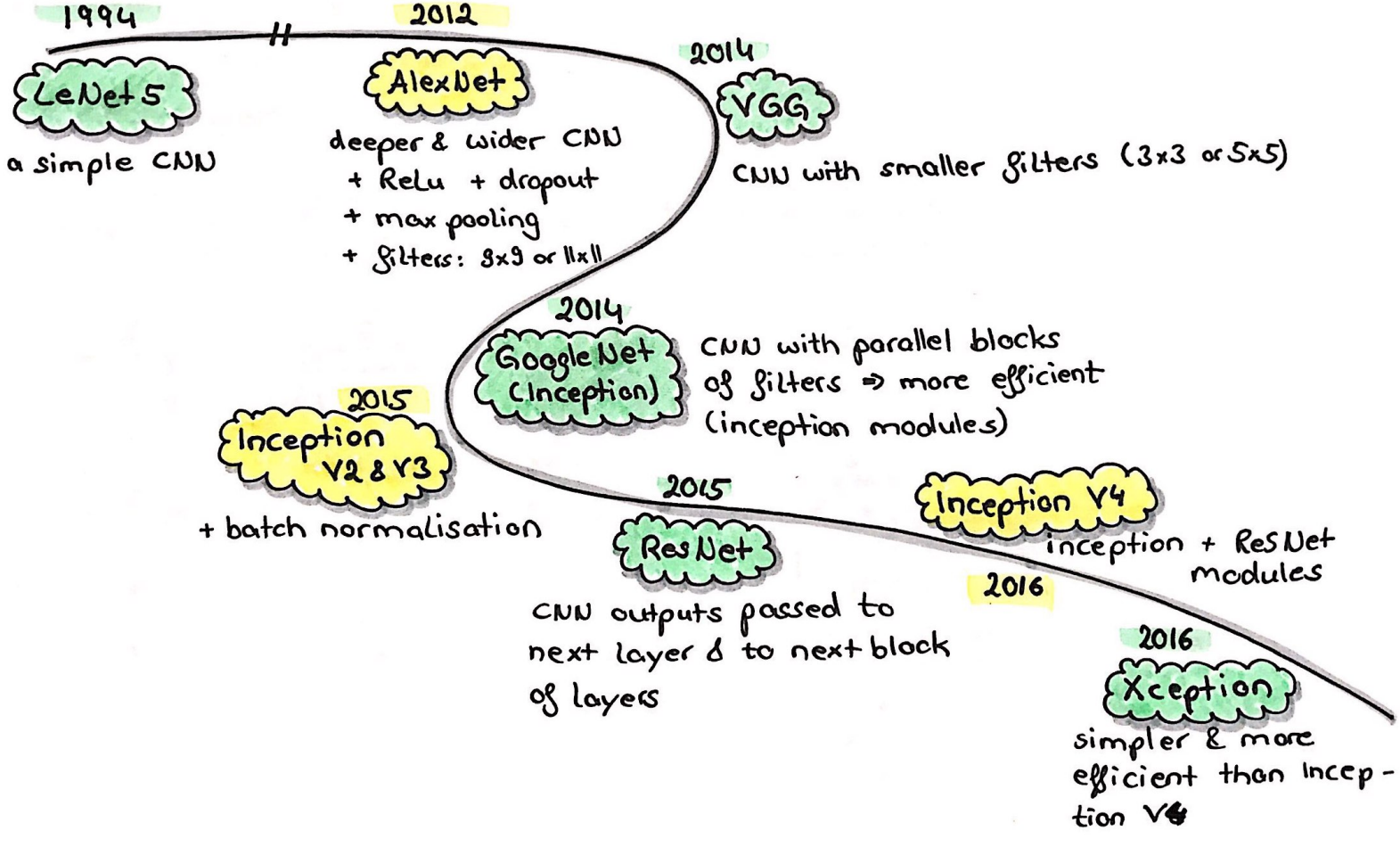


- reduces compute time
- boils information down

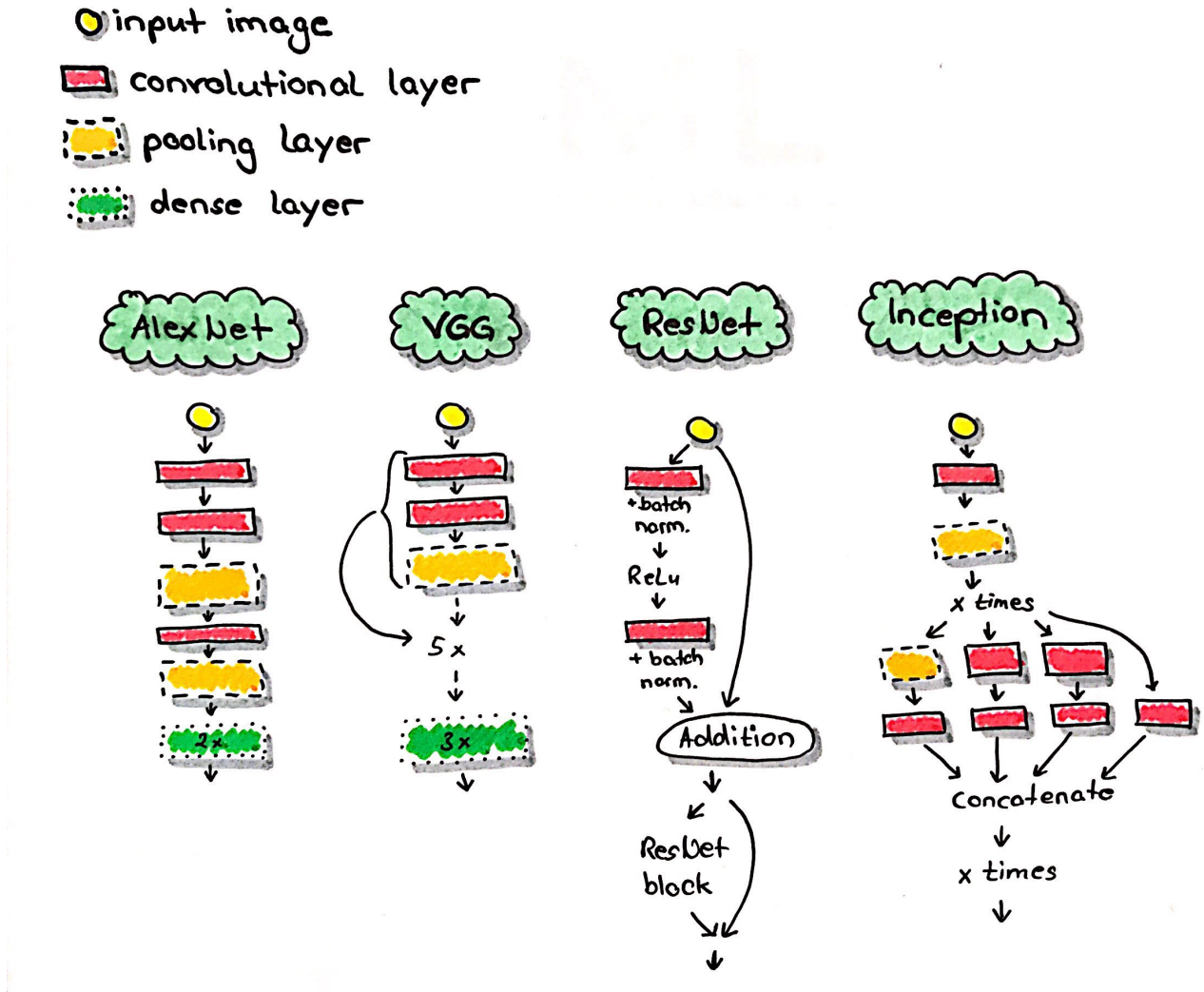
ConvNets


Input layer:
Convolutional layer (CL)
CL
Pooling layer (PL)
PL
Fully connected (dense) layers
Output layer:
dog 0.05
car 0.9
cat 0.05

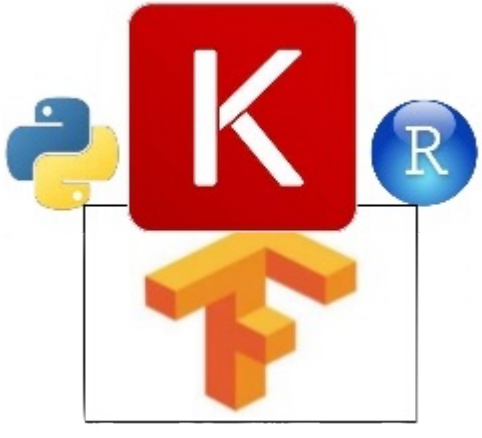
Evolution of neural nets for image recognition



CNN architectures

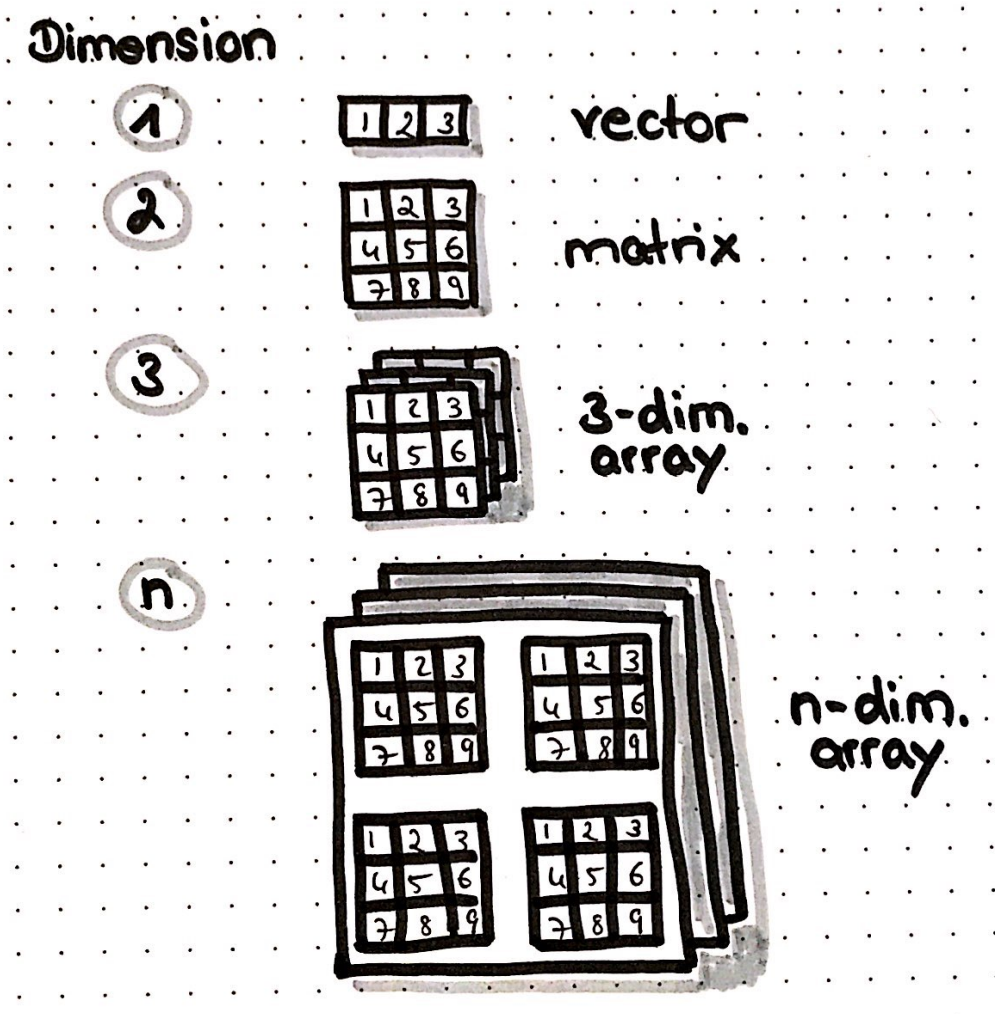


Introduction to TensorFlow



What are tensors?

Tensors = multidimensional arrays

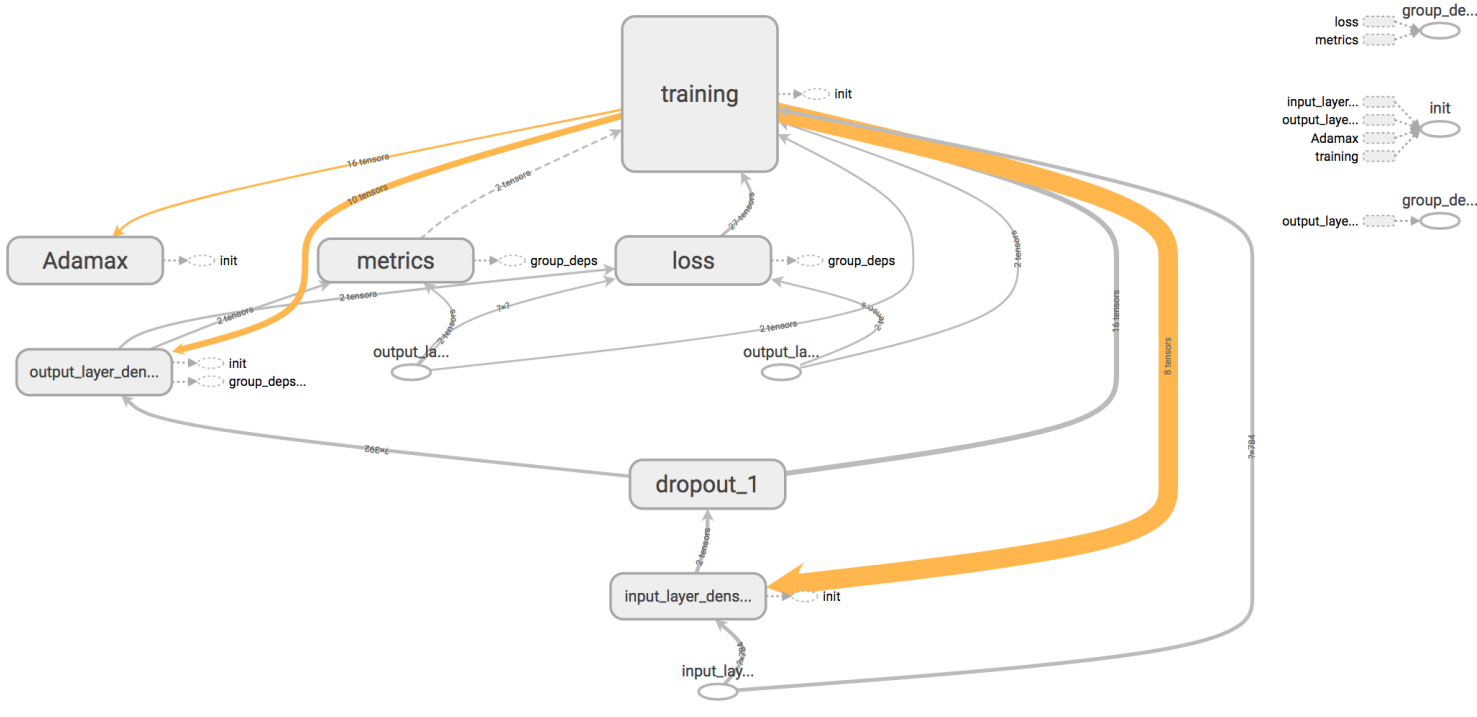


Tensor "Flow"

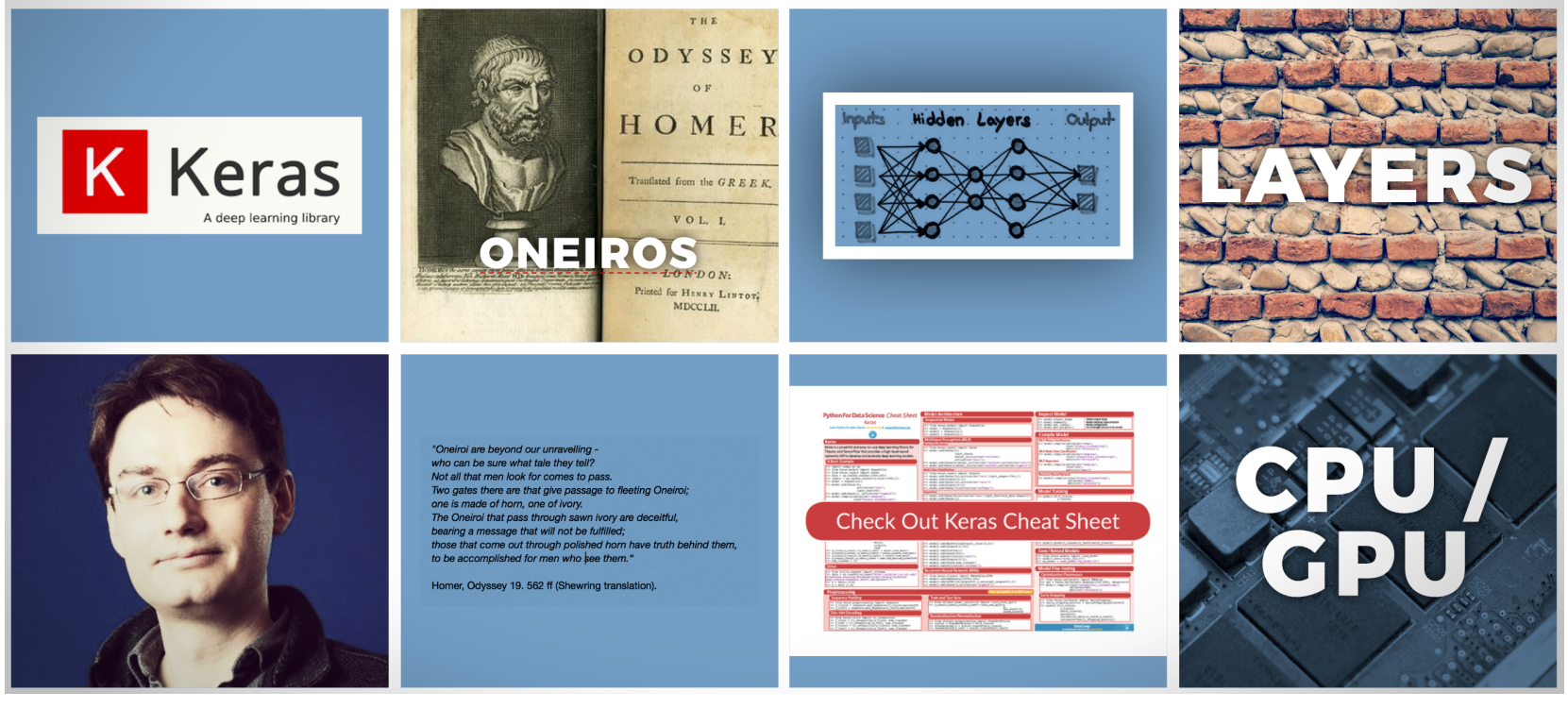
Graphs in TensorBoard

Main Graph

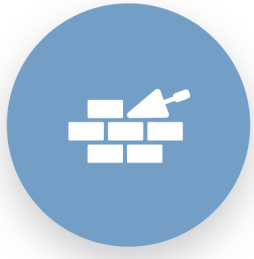
Auxiliary Nodes



Keras High-Level API for TensorFlow



Keras APIs



SEQUENTIAL MODELS

simple

suitable for most cases

linear order of layers

only one direction from input to output

FUNCTIONAL API

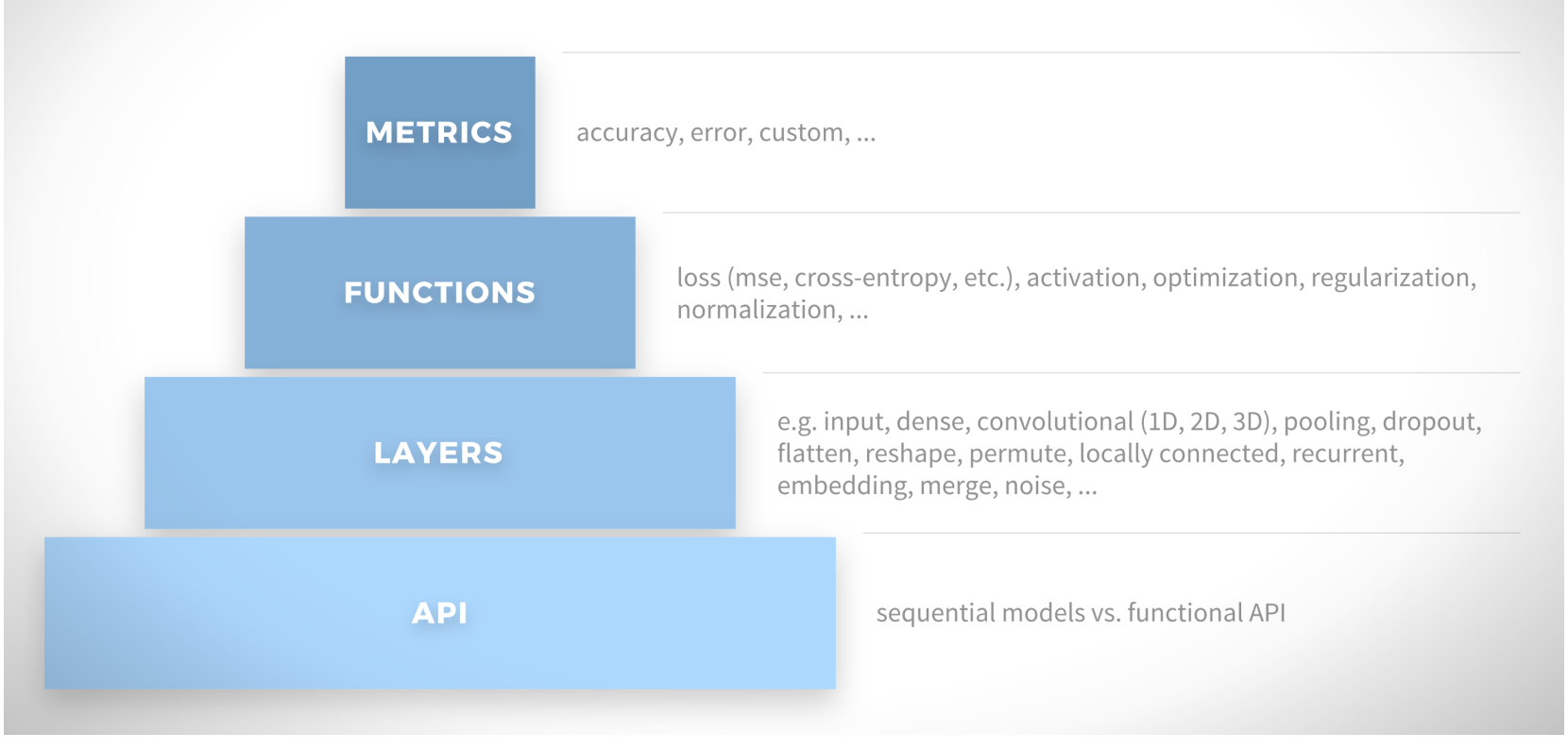
more complex

suitable for complex models

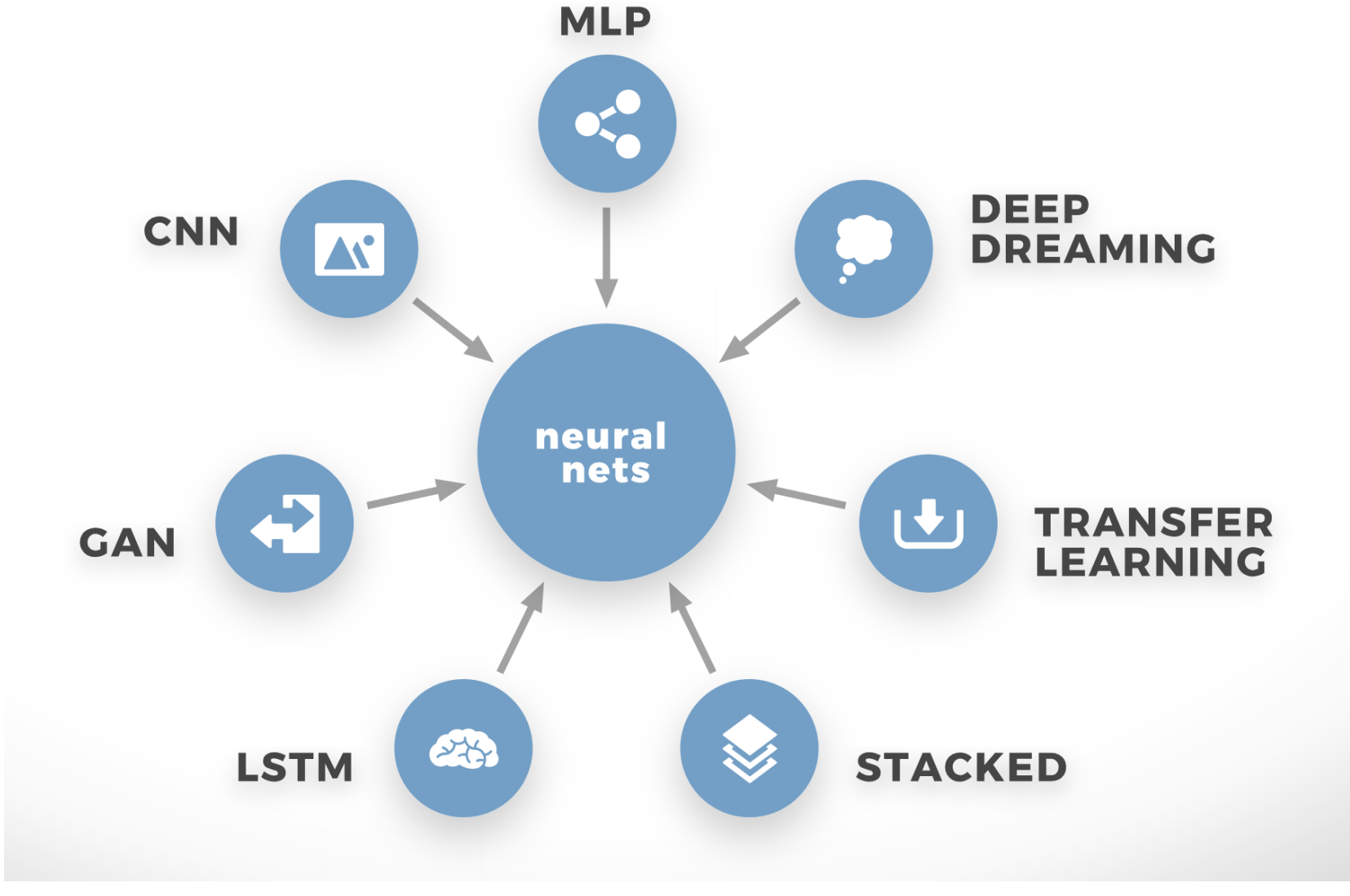
can have multiple in- or outputs

layers can be non-sequential, e.g. LSTM

Keras layers



Endless possibilities

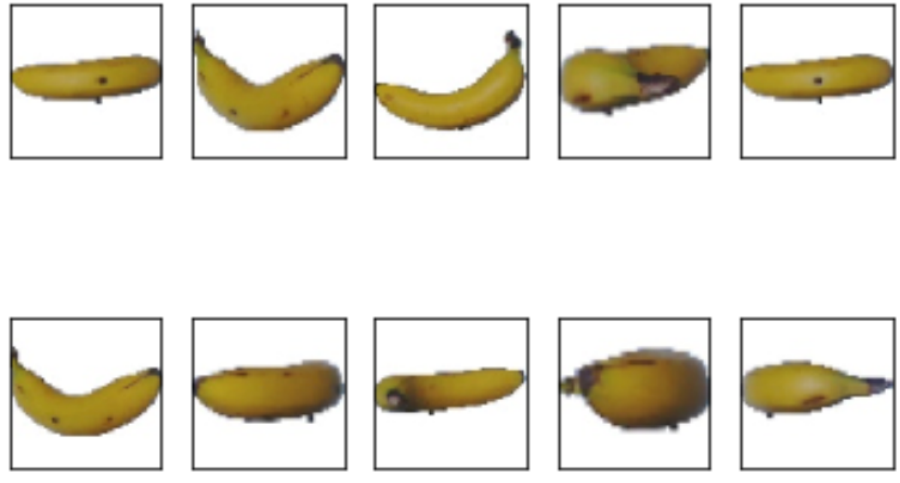


Let's get our hands dirty!

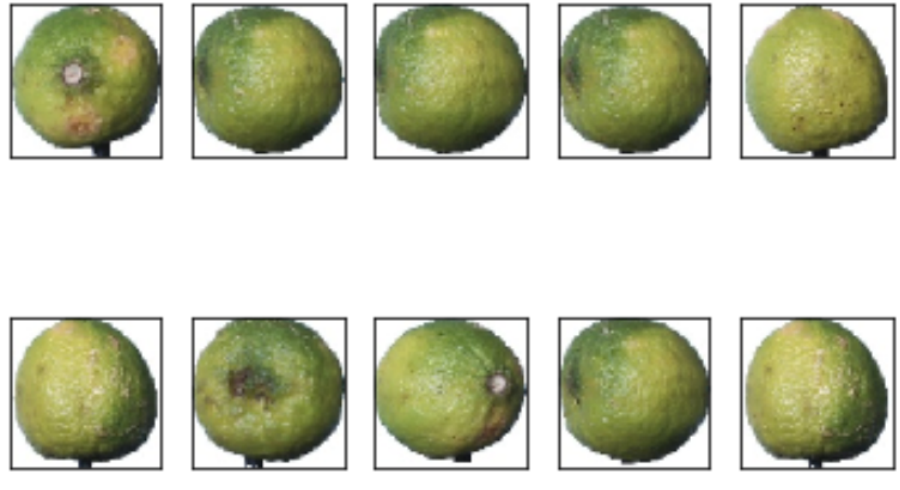
EDA

https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/01-eda.ipynb (https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/01-eda.ipynb)

In [10]:
```
plot_images(imgs=train_images[:10], rows=2, columns=5)
```



In [11]:
```
plot_images(imgs=train_images[1000:1010], rows=2, columns=5)
```



In [12]:
```
plot_images(imgs=train_images[2000:2010], rows=2, columns=5)
```

Apply pretrained nets

- VGG16

https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/02-pretrained-1.ipynb (https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/02-pretrained-1.ipynb)

- Xception

https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/02-pretrained-2.ipynb (https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/02-pretrained-2.ipynb)

- Try out:
  ResNet50

Use VGG16

https://keras.io/applications/#usage-examples-for-image-classification-models

In [10]:
```
model_vgg = VGG16(weights="imagenet")
```

In [11]:
```
model_vgg.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
```

Modify pretrained nets

https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/03-modify-pretrained.ipynb (https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/03-modify-pretrained.ipynb)

Modify VGG16

- transfer learning (freeze all but the penultimate layer and re-train the last Dense layer) and
- fine tuning (un-freeze the lower convolutional layers and retrain more layers)

Validation set: fit_generator has no option validation_split

https://keras.io/applications/#usage-examples-for-image-classification-models

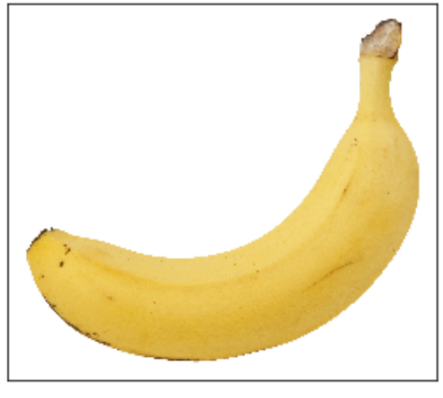
In [6]:
```
# important: exclude top layers
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(75, 75, 3))
#base_model.summary()
```

In [7]:
```
# Freeze the layers except the last 4 layers
for layer in base_model.layers[:-4]:
    layer.trainable = False
```

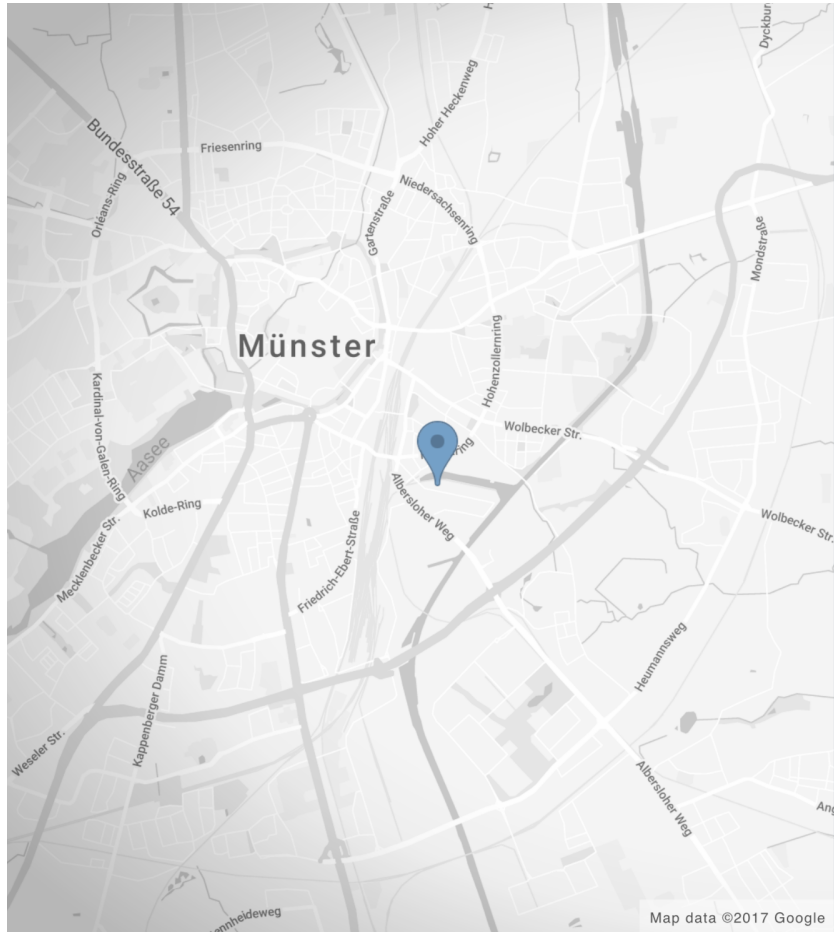
Build your own CNN

https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/04-fruits-cnn.ipynb (https://github.com/ShirinG/image_classification_keras_tf/blob/master/notebooks/04-fruits-cnn.ipynb)

```
In [16]: classify_image_model(test_images[0])
```



```
Predicted class: Banana with probability 99.99985694885254%
```



Thank you! And stay connected

Am Mittelhafen 14, Münster

ShirinGlander

shirin.glander@codecentric.de

www.codecentric.ai
https://www.youtube.com/codecentricAI
www.shirin-glander.de