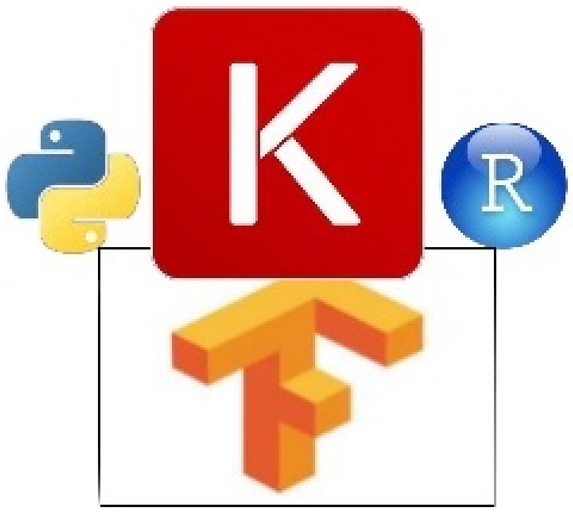
Image Classification & Natural Language Processing with Keras and TensorFlow in Python

Workshop



Dr. Shirin Glander

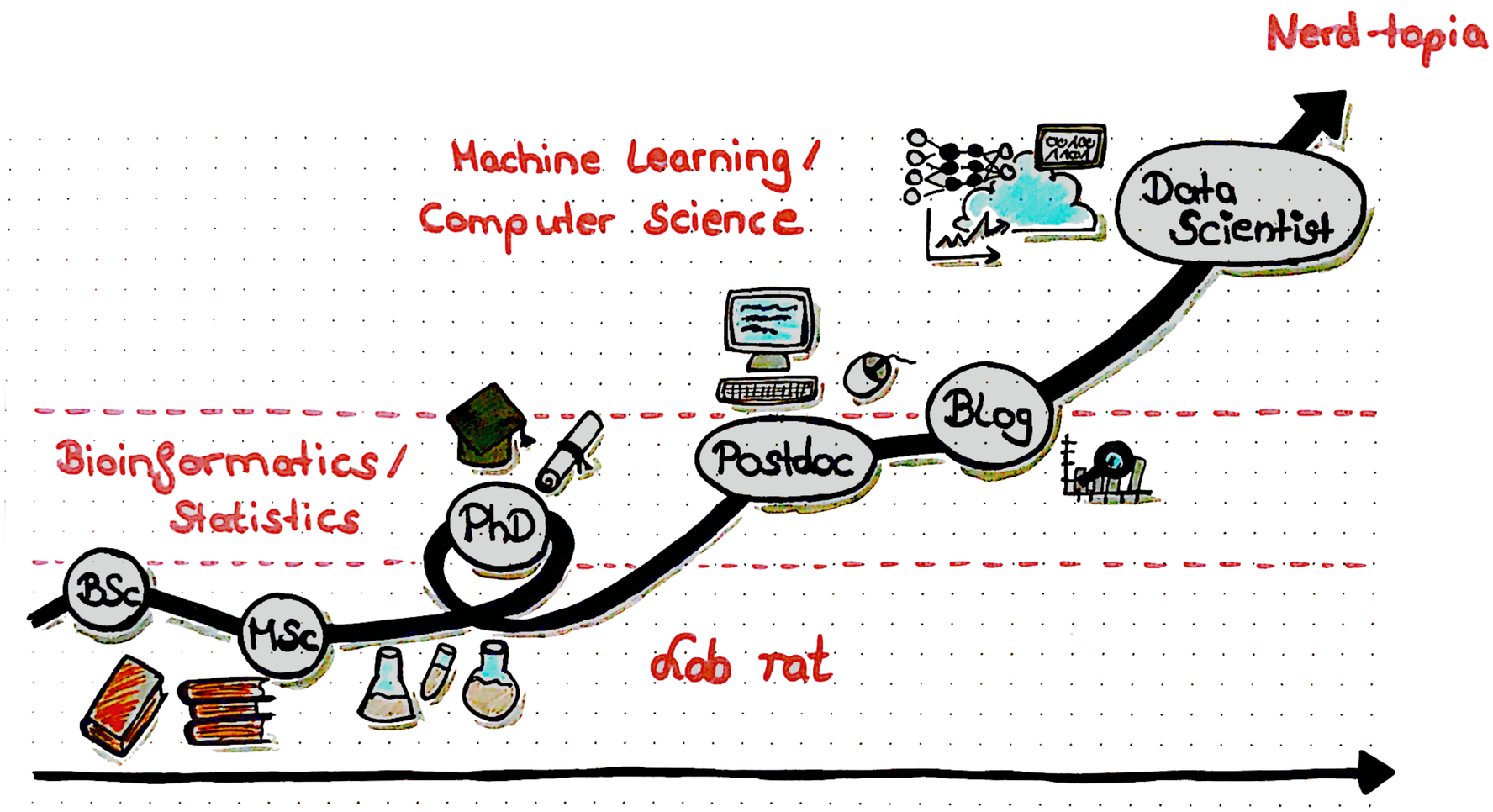
24.10.2018 - Bitmarck

About this Workshop

- Learn what neural nets are and how they can be used in computer vision & NLP.
- Understand how a computer learns to "see" ...
- ... and how a computer understands natural language.
- Apply pre-trained nets and modify them.
- Build a model from scratch that differentiates between fruits on images.
- Visualise convolutions and layers.
- Train a text classification model with CNNs ...
- ... and with LSTMs.
- (Bonus: Explaining our classifications with LIME)

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

About me


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

Why Keras + TensorFlow?

The Keras philosopy

- Rapid protoyping
- Very fast turnover from idea to model

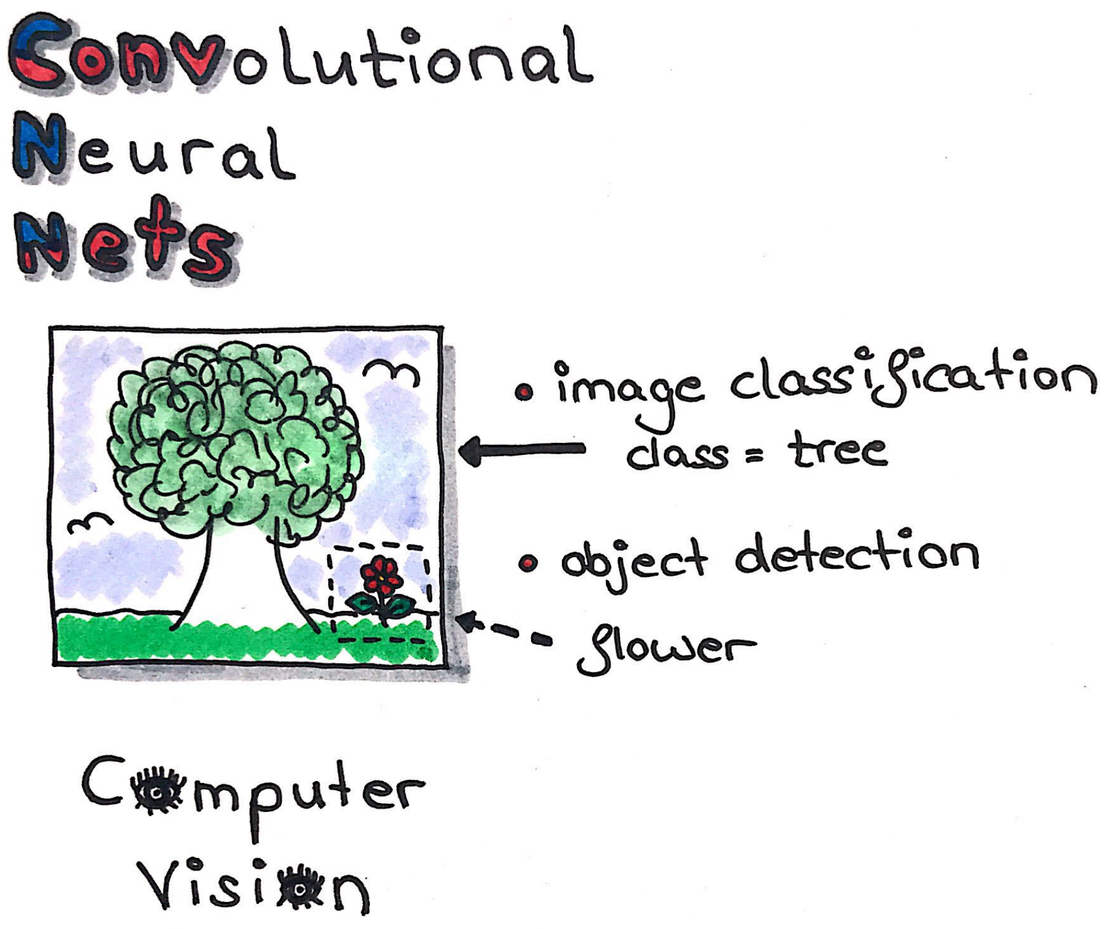
Keras is now the default TensorFlow API

- Keras is the first high-level library added to core TensorFlow at Google

Start simple & small

- Always start simple and small to generate a baseline model against which to compare!

How does a computer learn to "see"?

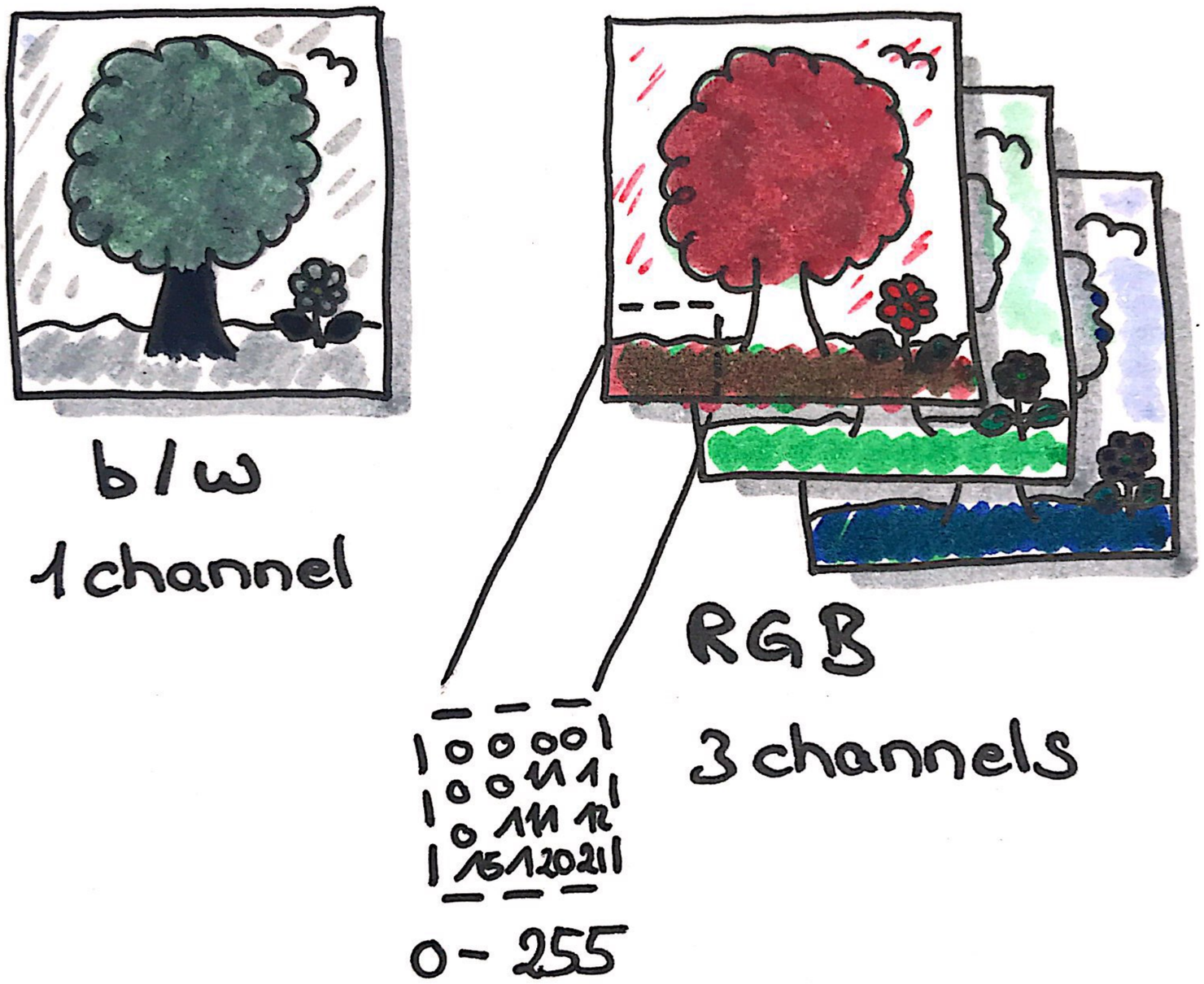


See also [this video from codecentric.AI](#) for more on Computer Vision Basics!

CNNs can also be used for text classification

Dies ist ein Blindtext. An ihm lässt sich vieles über die Schrift ablesen, in der er gesetzt ist. Auf den ersten Blick wird der Grauwert der Schriftfläche sichtbar. Dann kann man prüfen, wie gut die Schrift zu lesen ist und wie sie auf den Leser wirkt.
Dies ist ein Blindtext. An ihm lässt sich vieles über die Schrift ablesen, in der er gesetzt ist. Auf den ersten Blick wird der Grauwert der Schriftfläche sichtbar. Dann kann man prüfen, wie gut die Schrift zu lesen ist und wie sie auf den Leser wirkt.

How does a computer learn to "see"?



How does a computer understand natural language?

Document Term Matrix

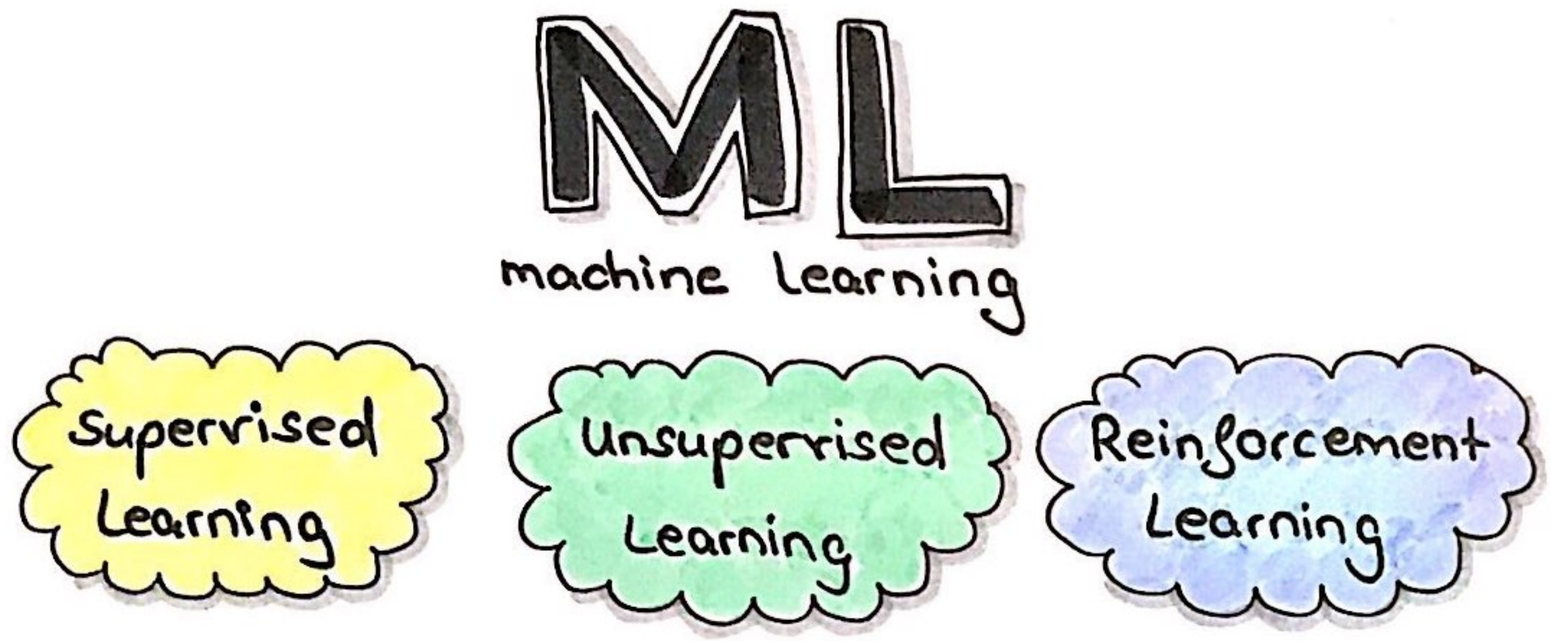
- Generate list of words in the complete corpus
- Count word frequency in each document
- Optional: normalize (e.g. tf/idf)
- Optional: remove stop words, etc.

Word embedding matrices

Word embeddings

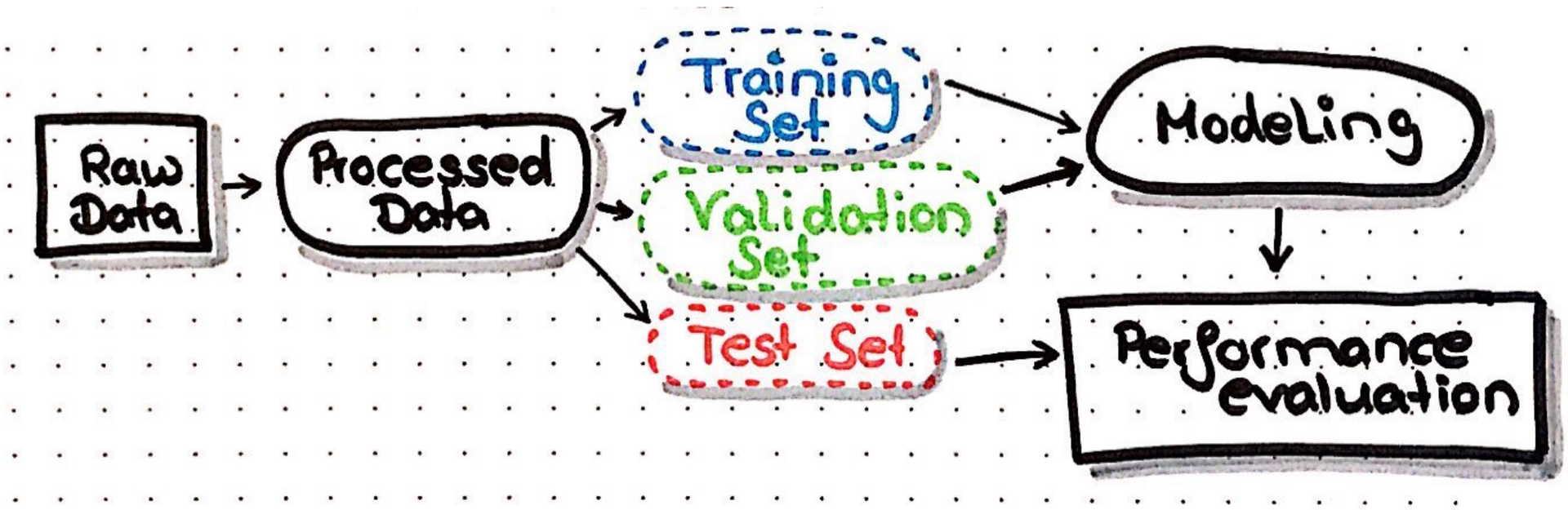
- Words/phrases are mapped to vectors
- Can be created with neural nets, PCA/t-SNE/UMAP, co-occurrence matrices
- Vectors represent context
- e.g. word2vec

What is Machine Learning (ML)



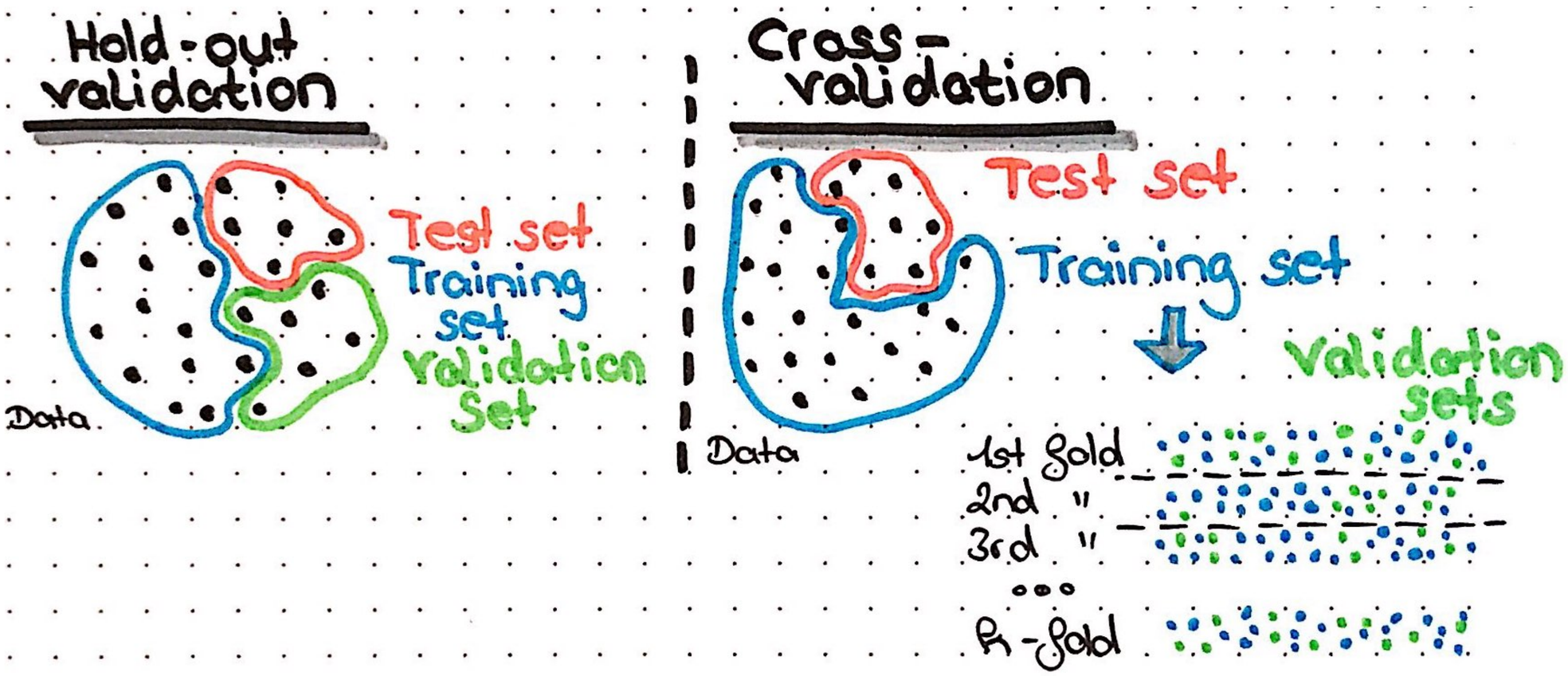
Labelled data	unlabelled data	self-generating data
• Classification or • Regression	• Clustering • Dimensionality reduction • Anomaly Detection	reward + punishment • Real-time Learning

A typical ML Workflow

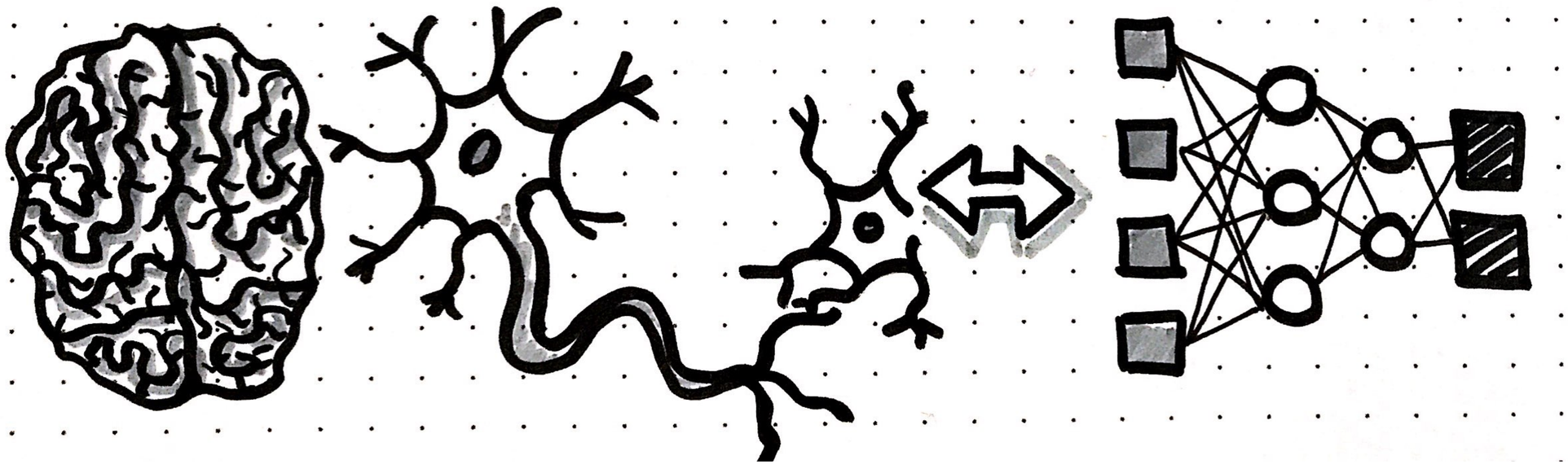


Why use validation and test data?

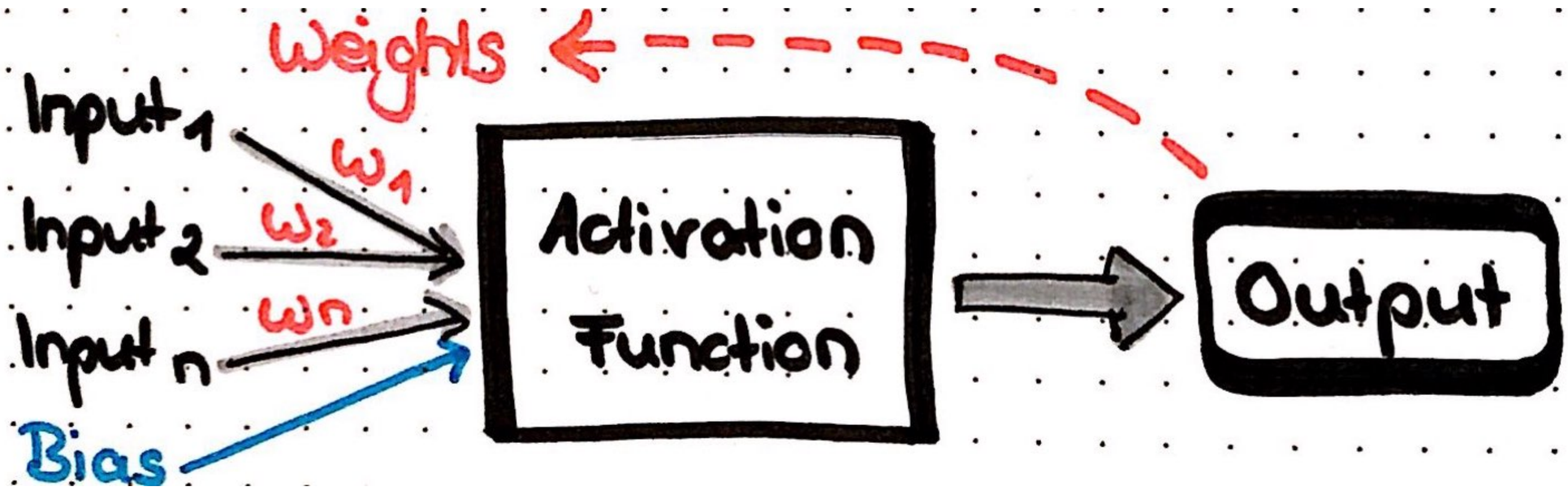
Validation methods



What are neural nets?

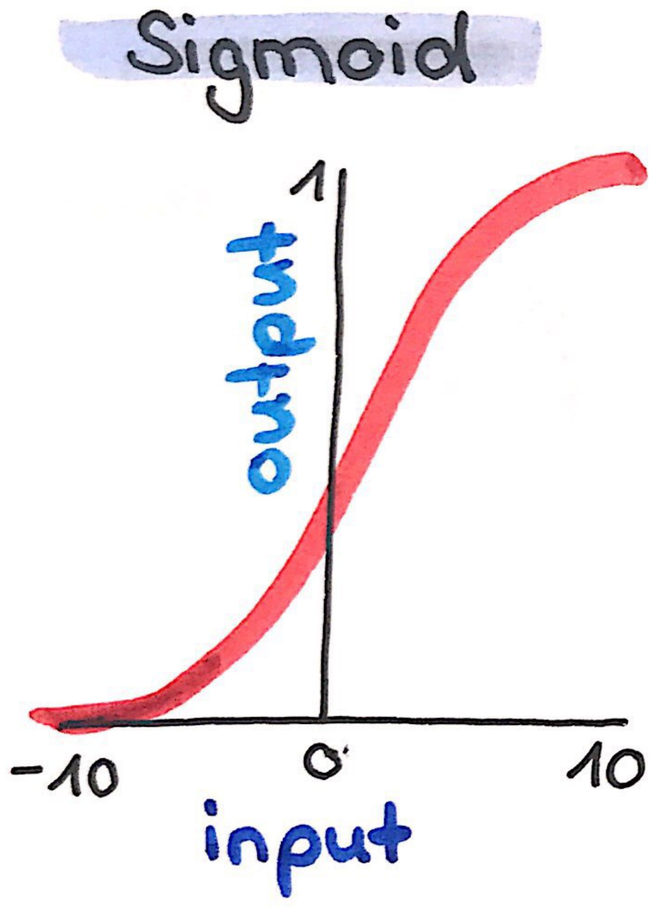


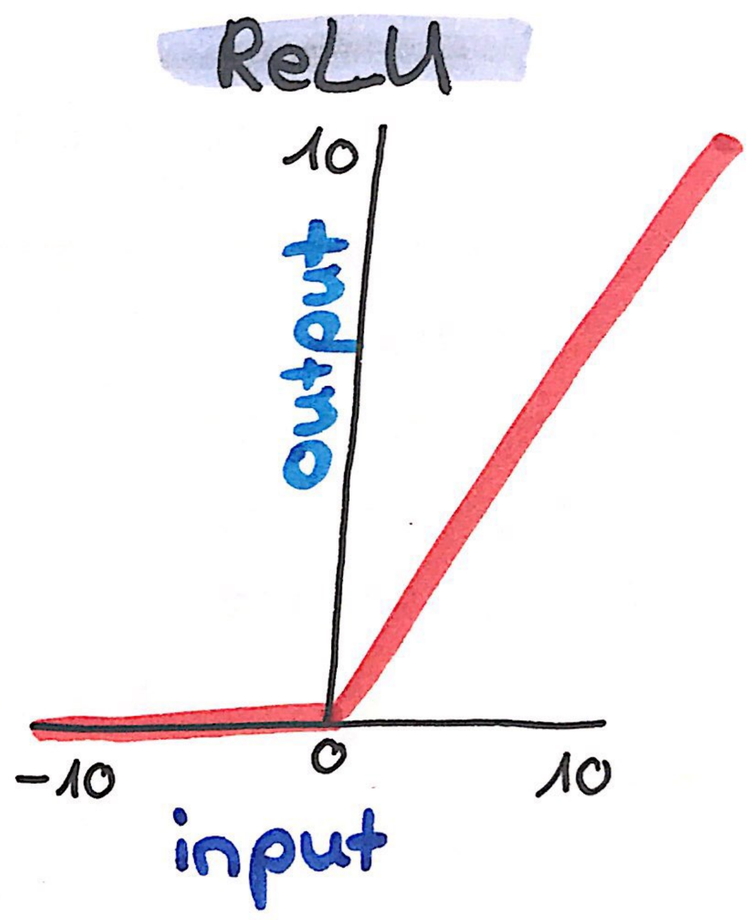
Perceptrons



Activation functions normalise input

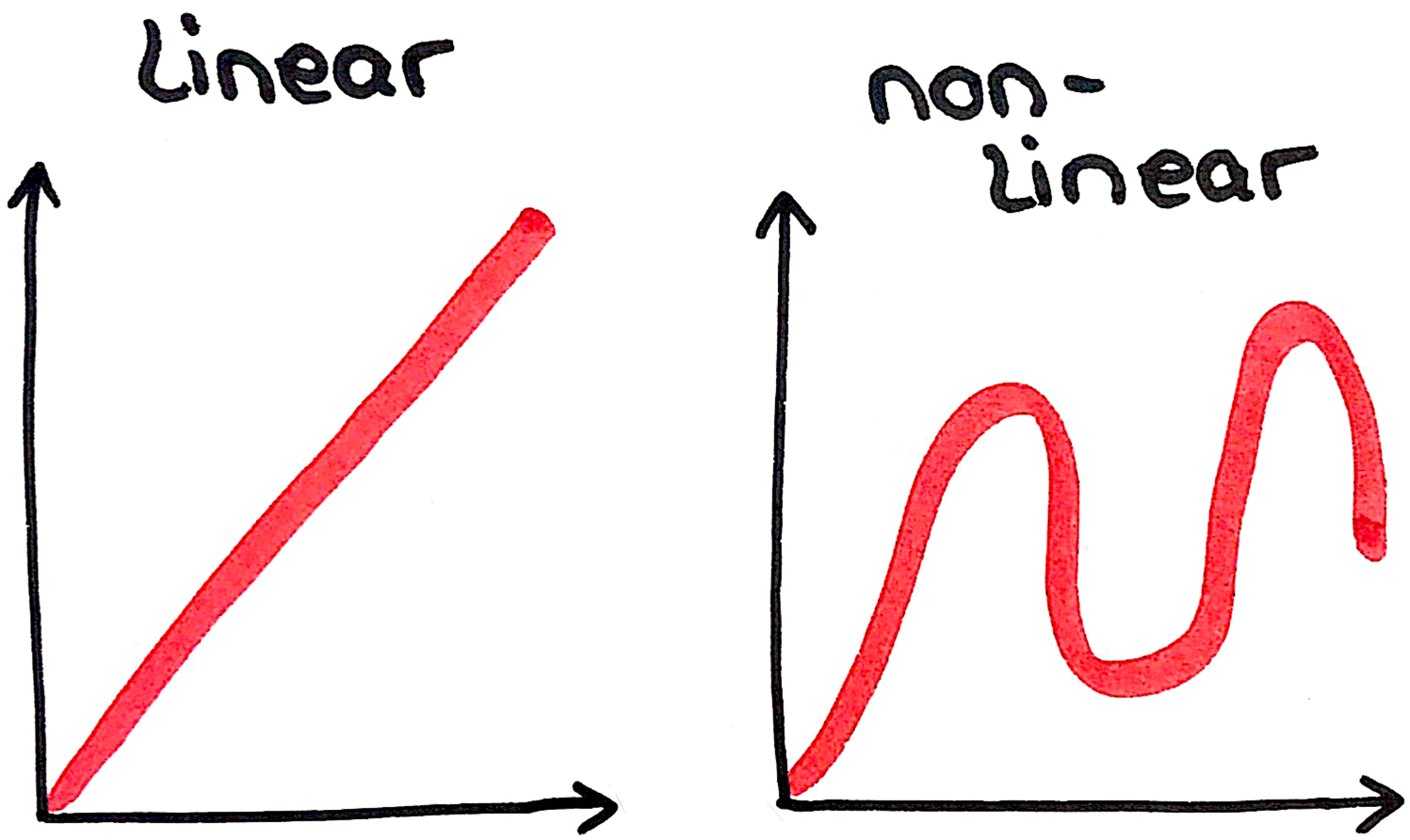
- every problem can be described with a mathematical function



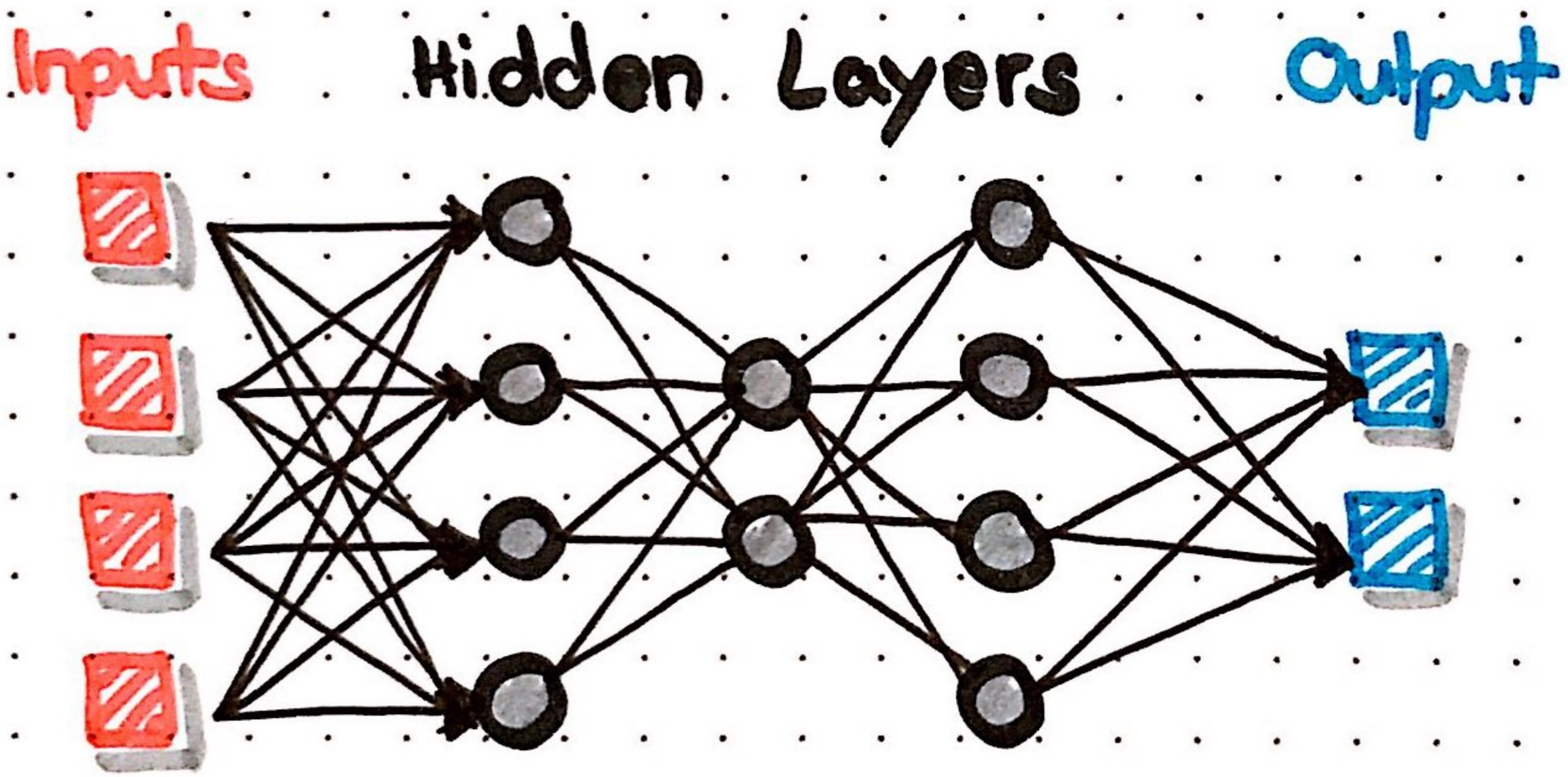


Activation functions make non-linearity possible

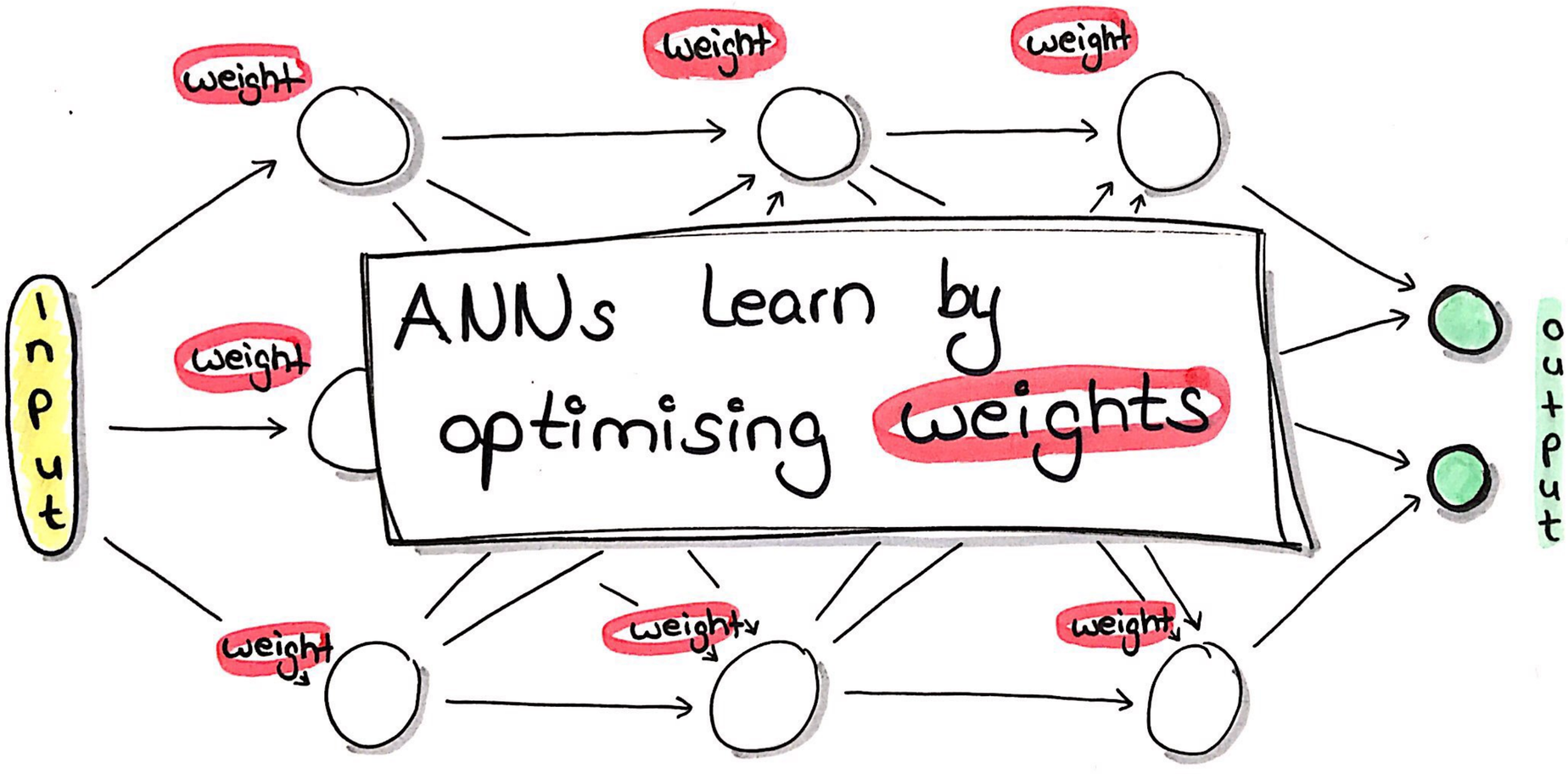
- non-linear activation functions allow us to approximate ANY mathematical formula with neural nets



Multi-Layer Perceptrons

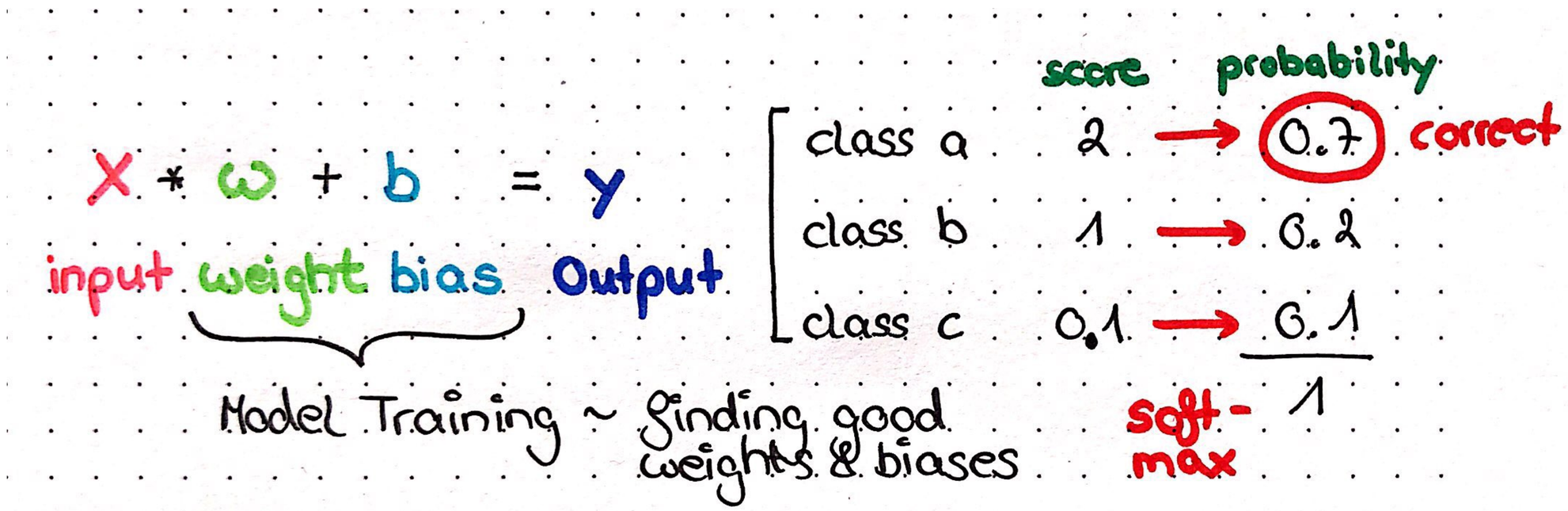


How does a neural net learn?



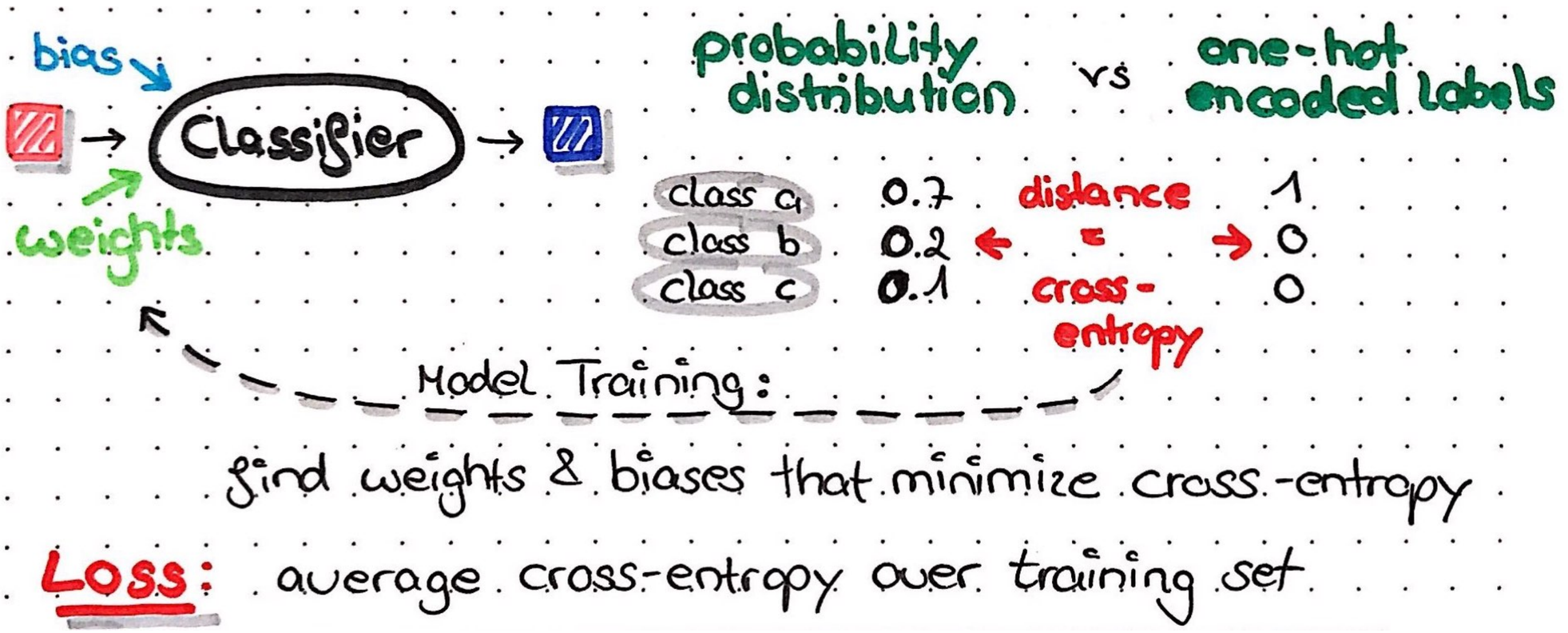
How does a neural net learn?

The Softmax function



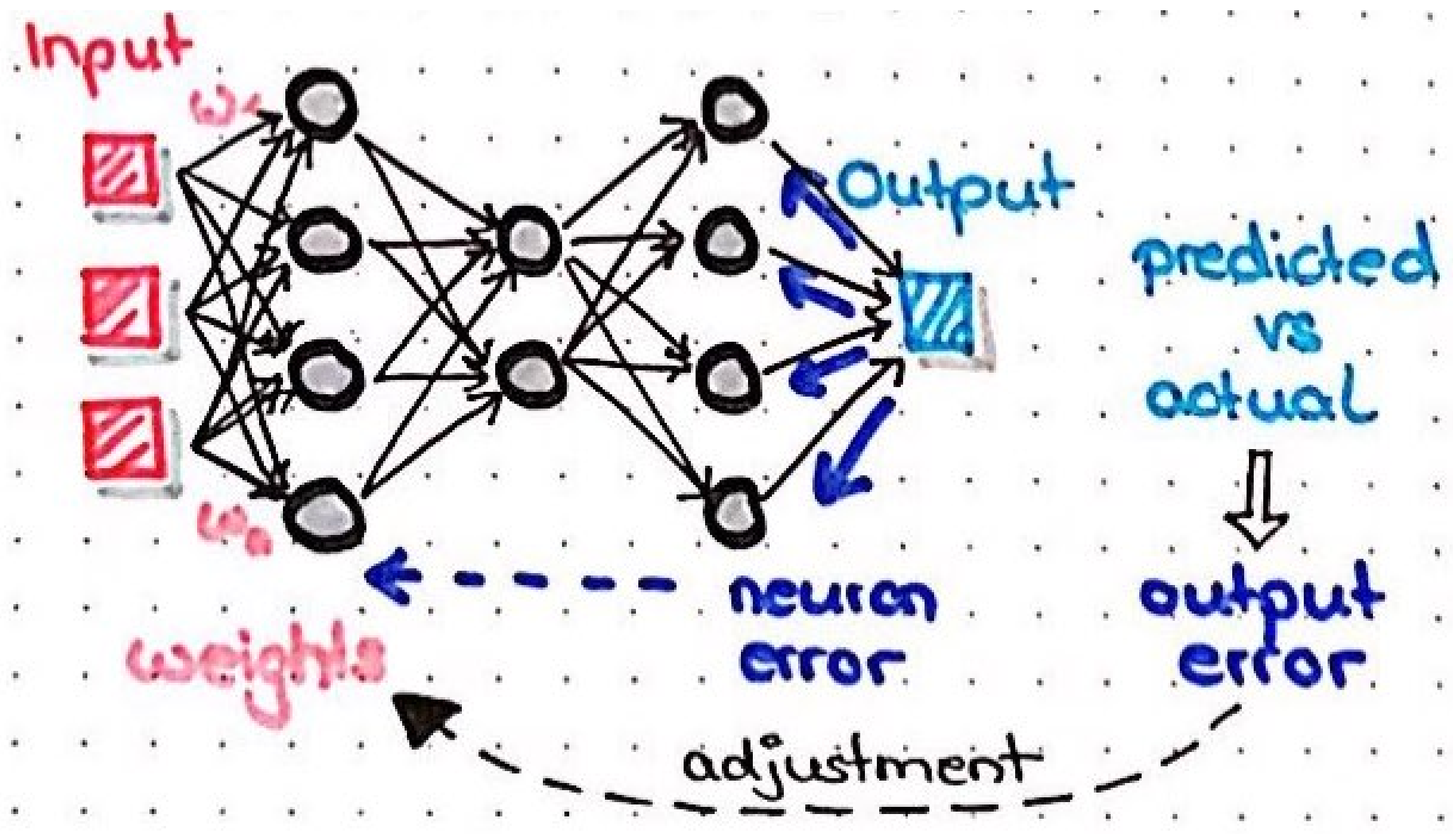
How does a neural net learn?

Cross-entropy



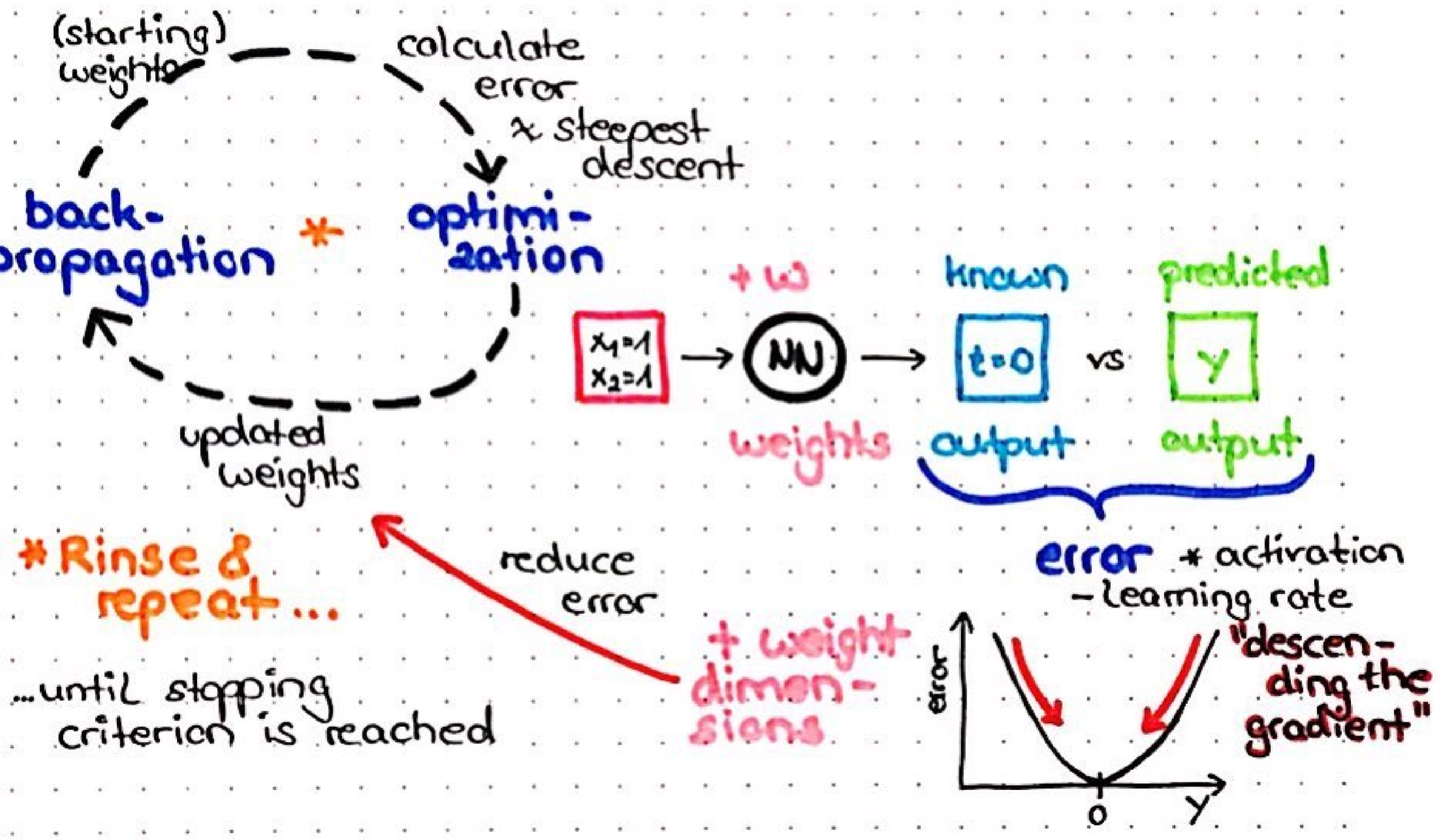
How does a neural net learn?

Backpropagation

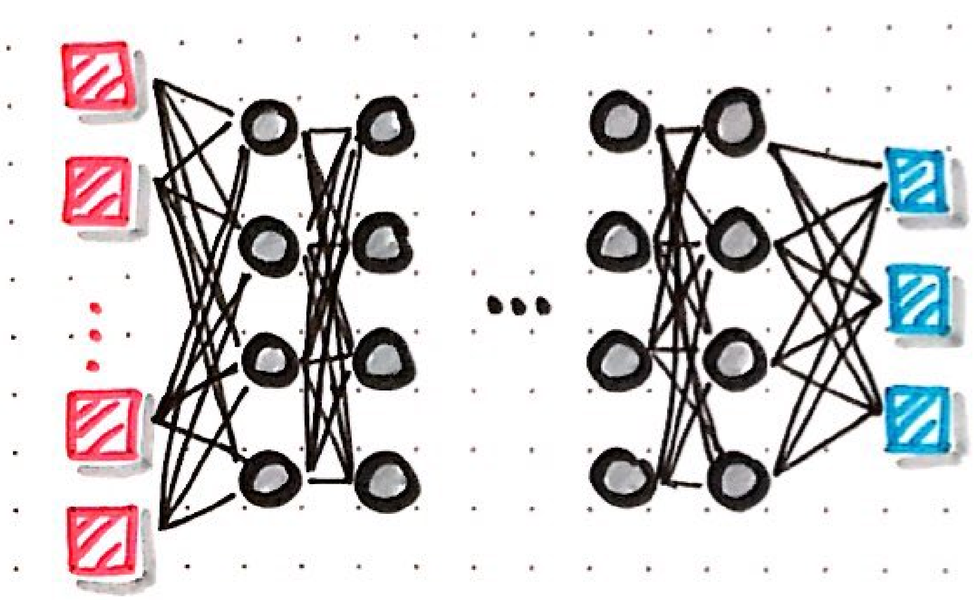


How does a neural net learn?

Gradient descent optimization



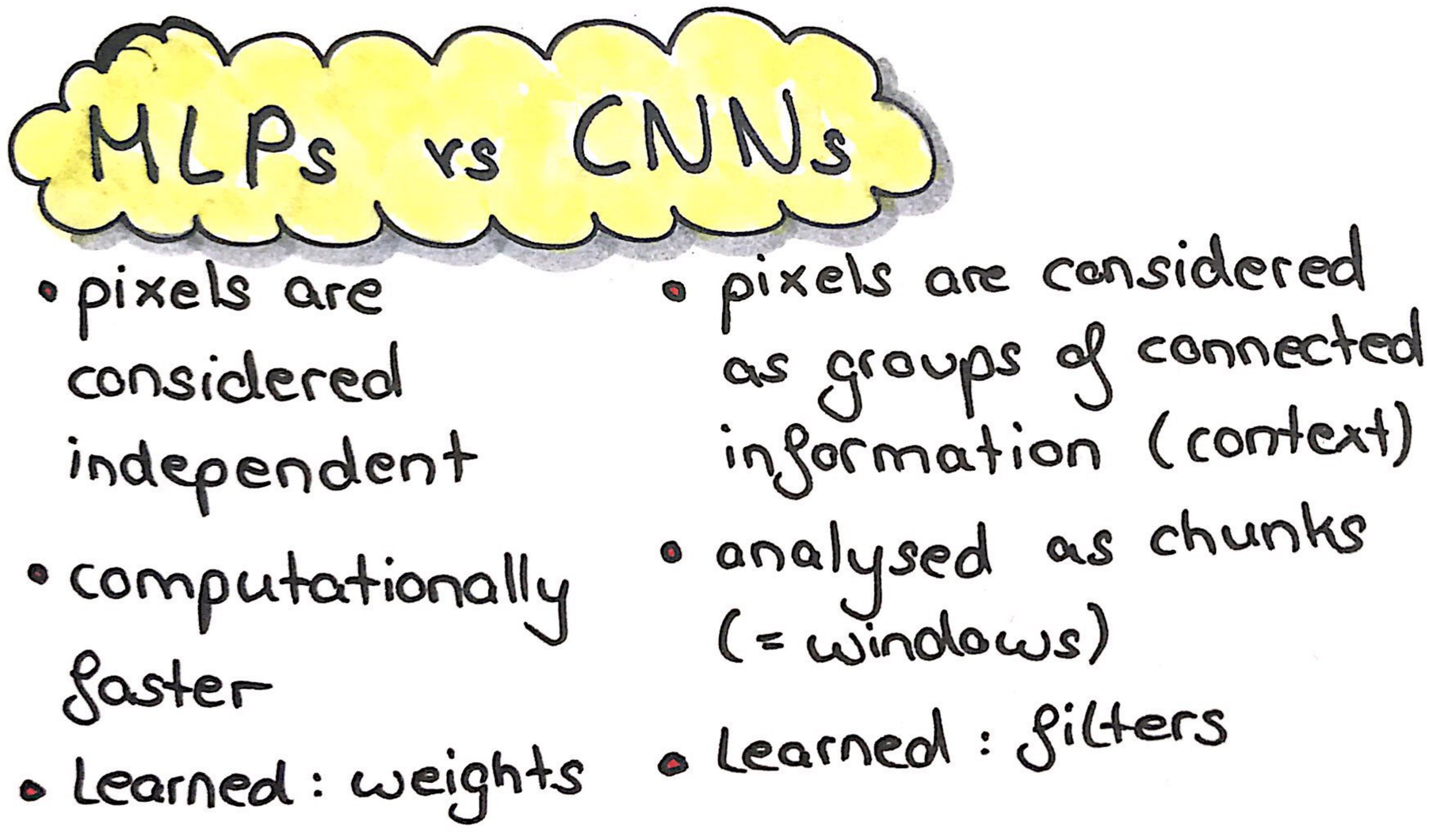
Deep Learning



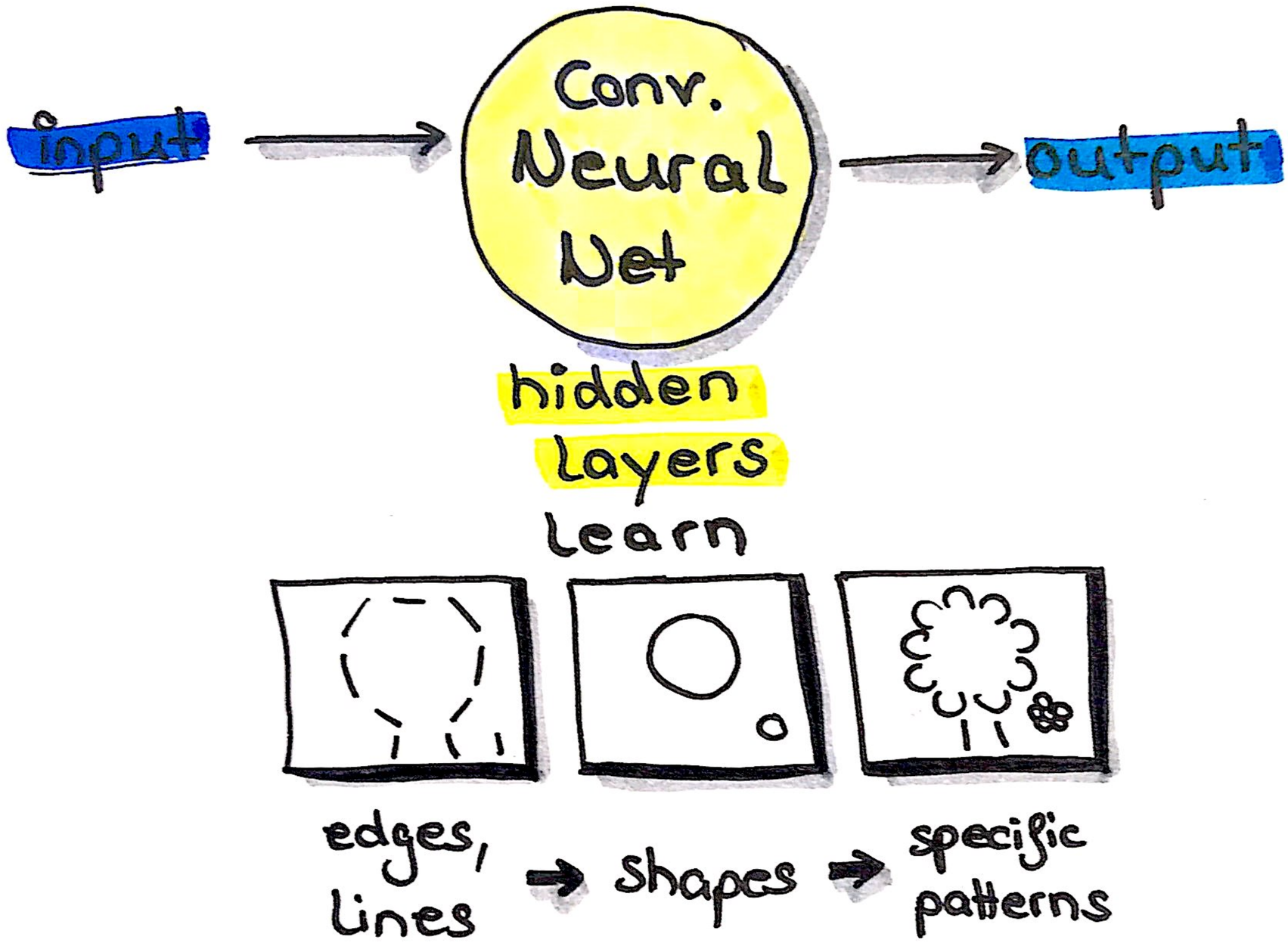
- supervised, semi-supervised or unsupervised
- NLP, speech recognition
- image recognition
- object classification
- recommender systems

etc.

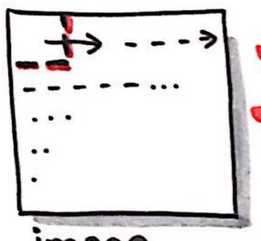
Convolutional Neural Nets



ConvNets


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

ConvNets

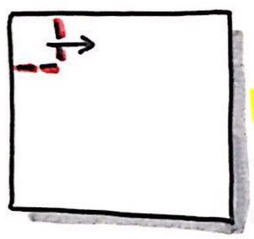


sliding window

↳ window **size** (e.g. 3×3 pixels)

image

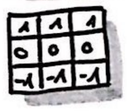
↓



apply **filters** to every sliding window

e.g.

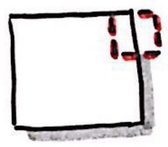
vertical lines



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

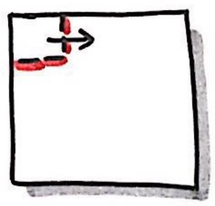
=

feature maps

↓

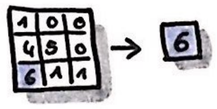
stacks of **feature maps**

→



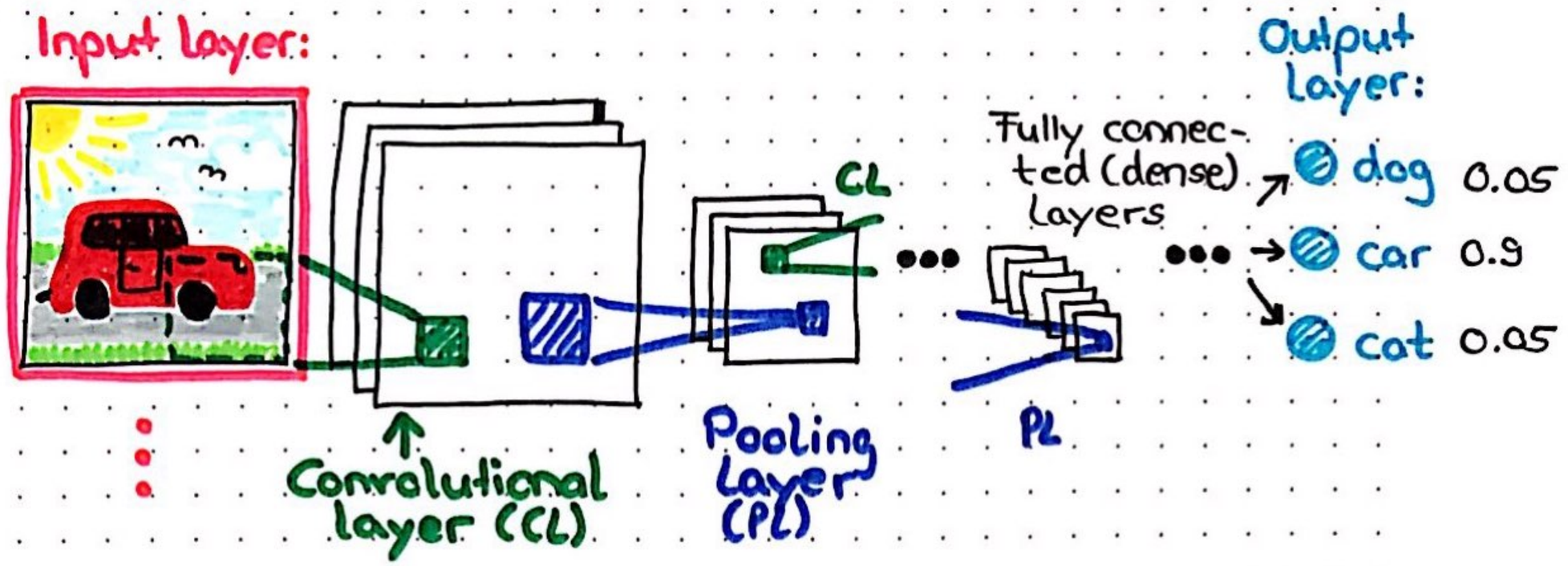
Pooling Layers

e.g. max pooling

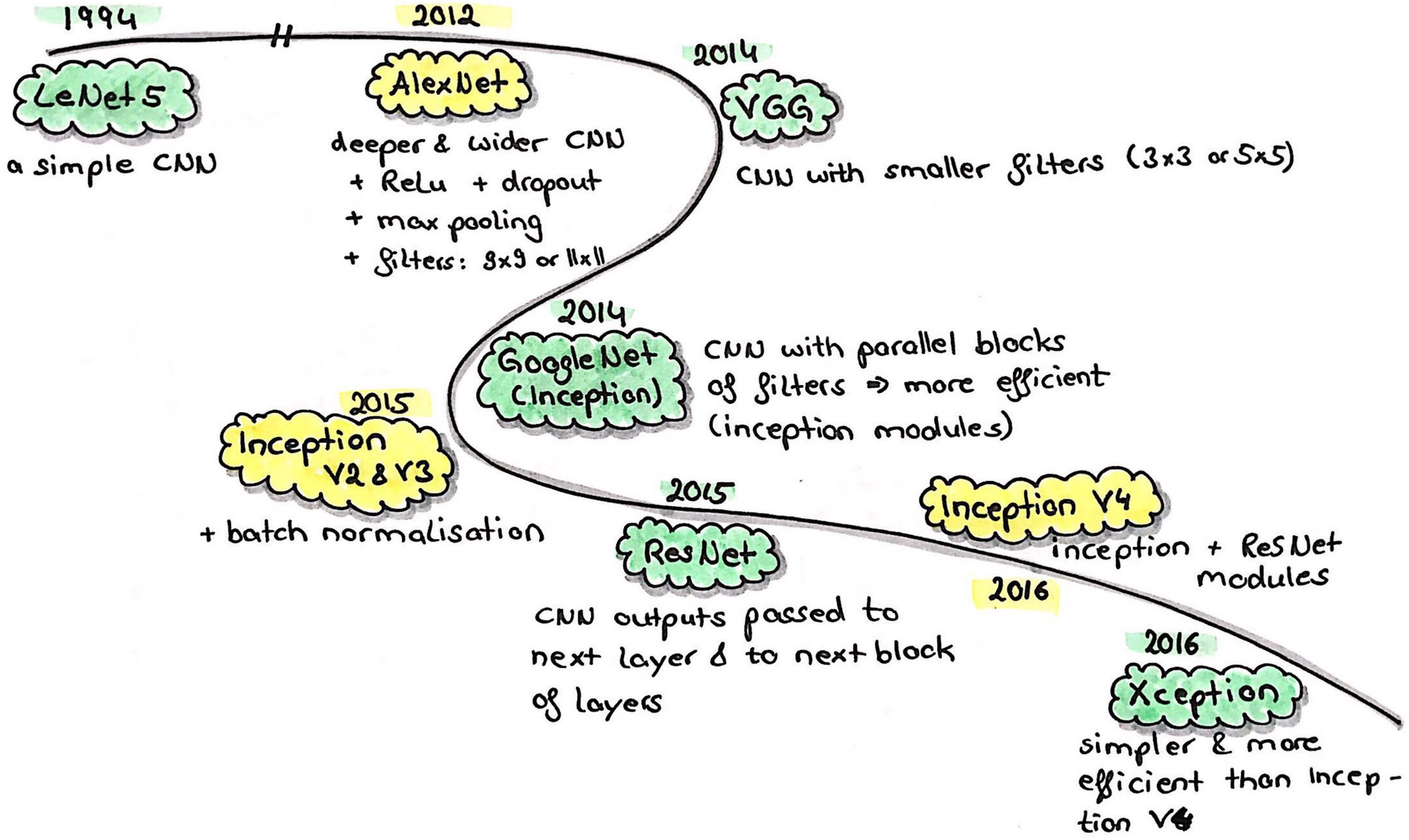


- reduces compute time
- boils information down

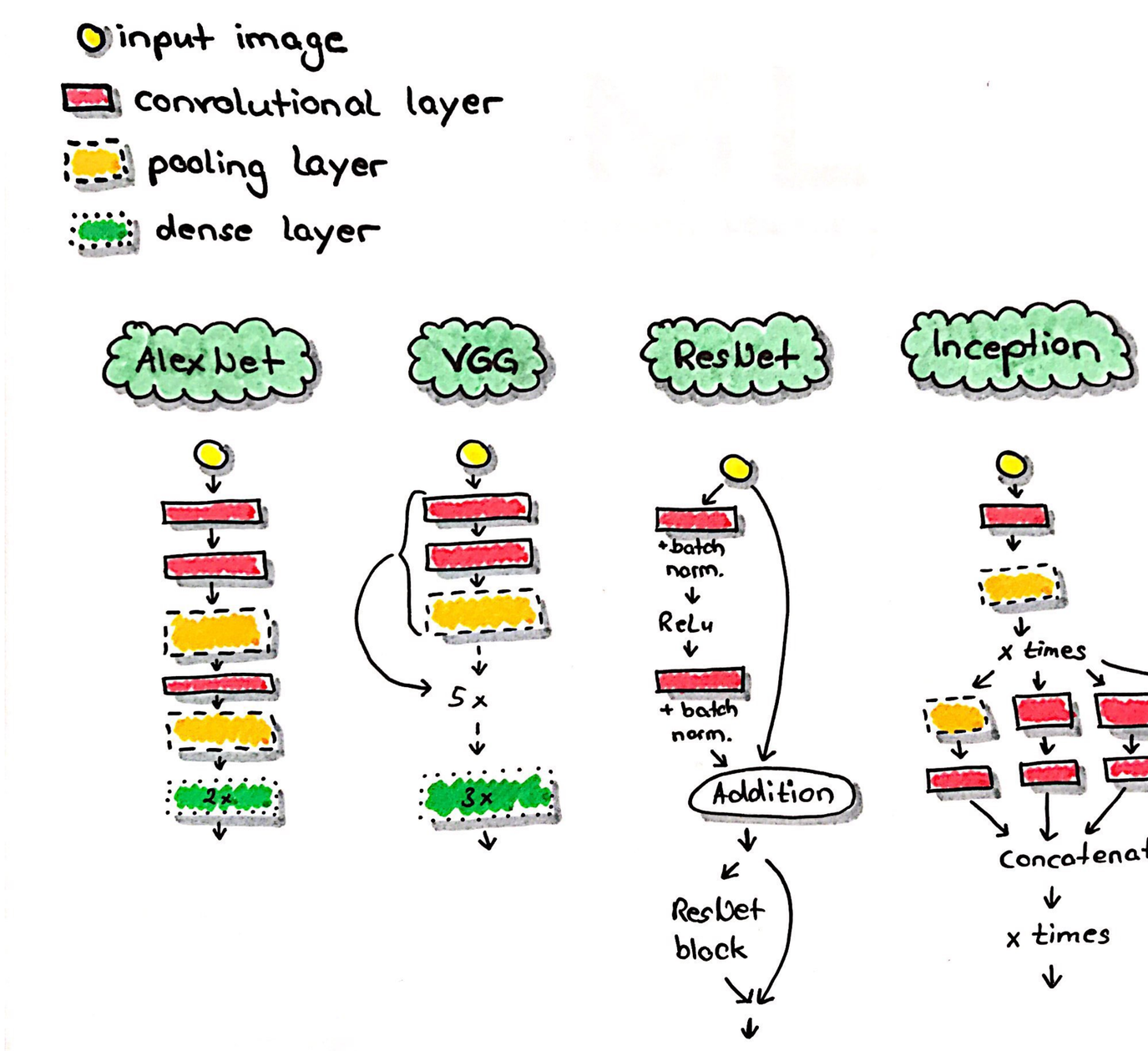
ConvNets


Input layer:
3
3
Convolutional layer (CL)
CL
Pooling layer (PL)
PL
Fully connected (dense) layers
Output layer:
dog 0.05
car 0.9
cat 0.05

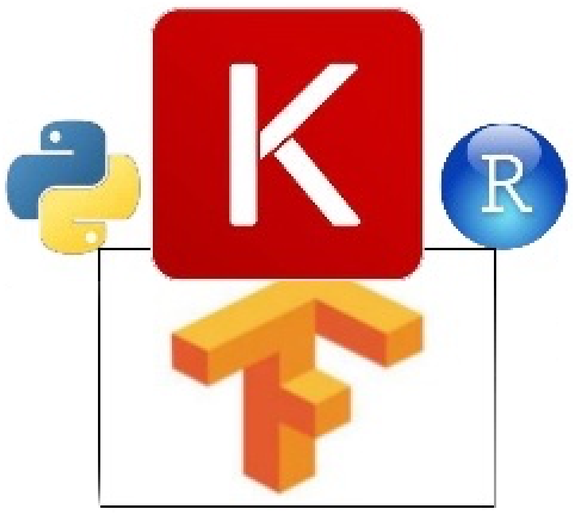
Evolution of neural nets for image recognition



CNN architectures

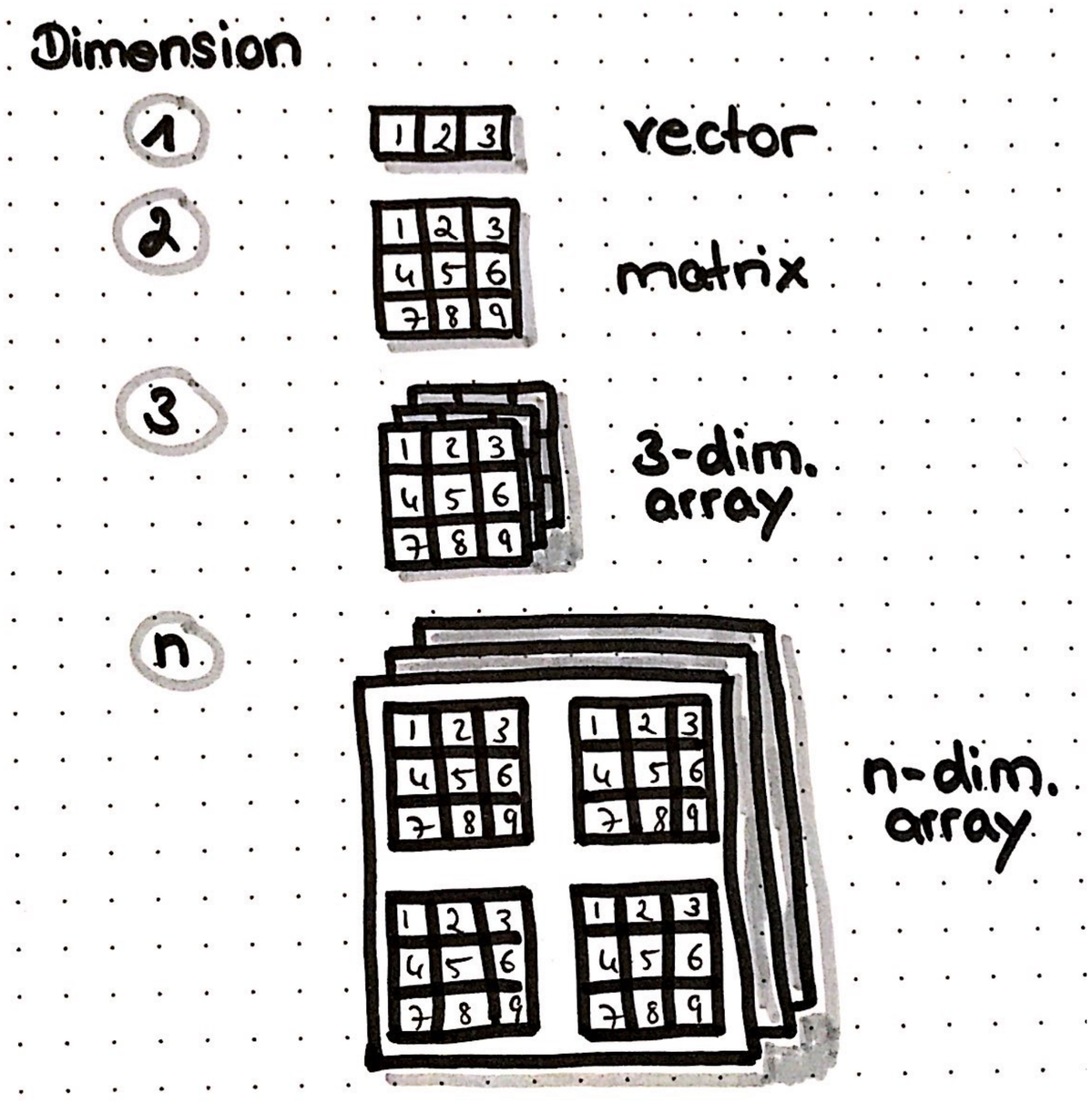


Introduction to TensorFlow



What are tensors?

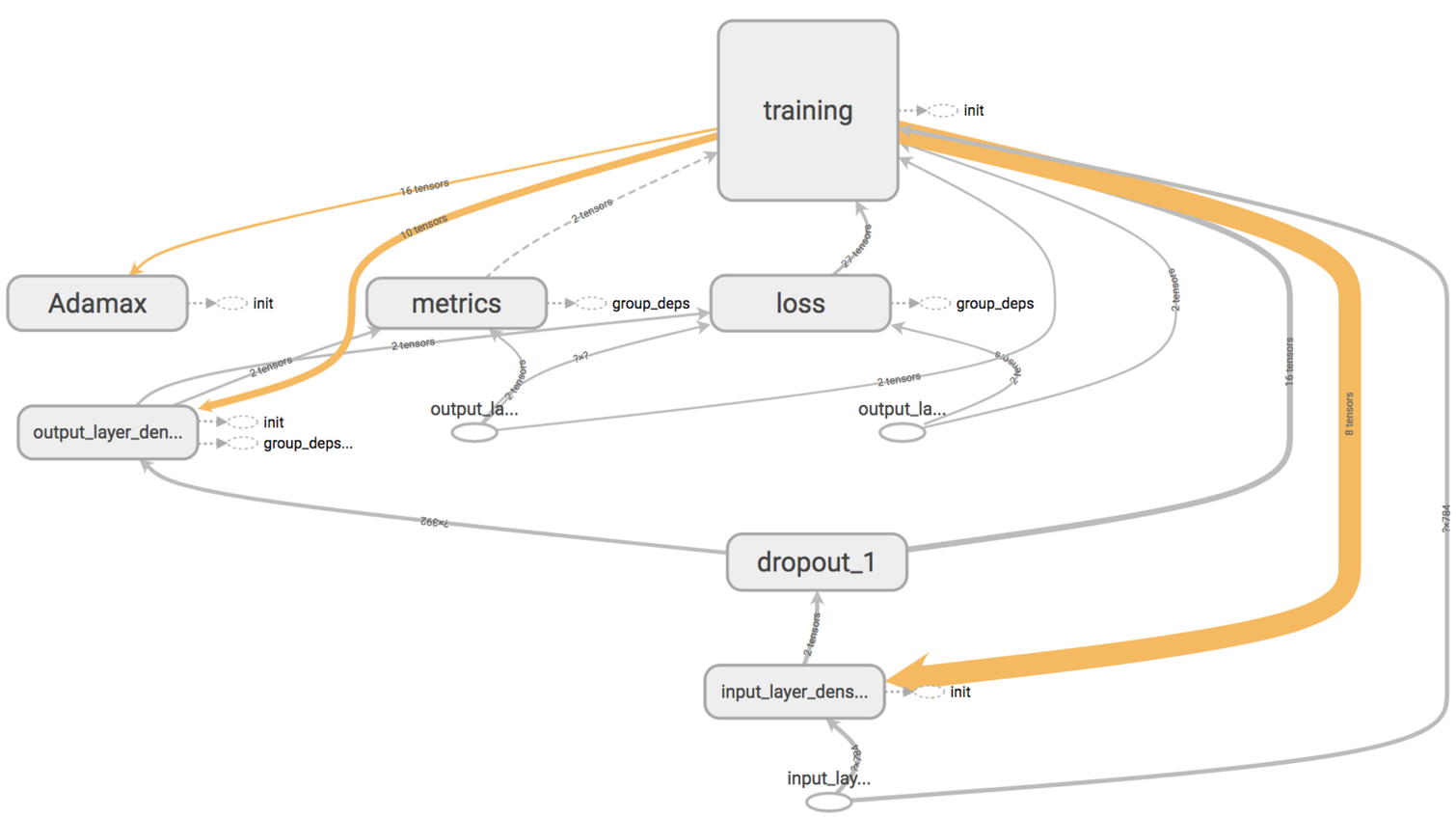
Tensors = multidimensional arrays



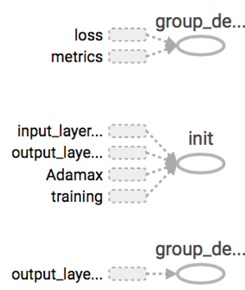
Tensor "Flow"

Graphs in TensorBoard

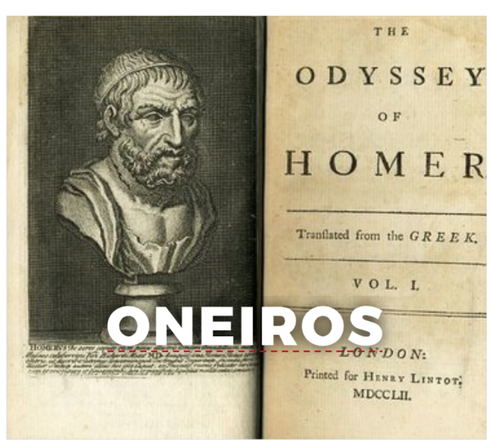
Main Graph

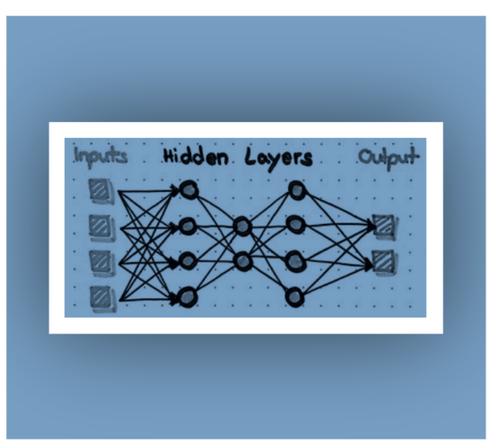


Auxiliary Nodes



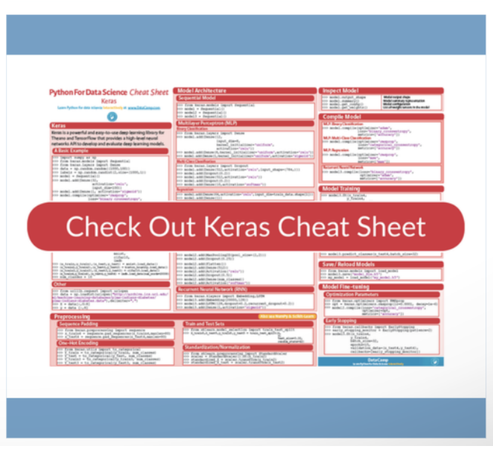
Keras High-Level API for TensorFlow

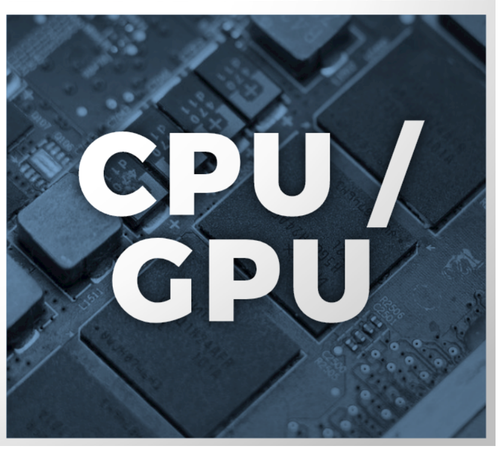




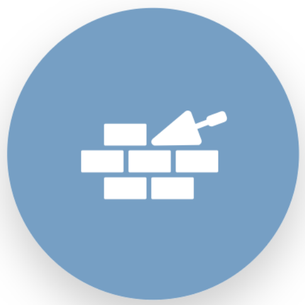
"Oneiroi are beyond our unravelling -
who can be sure what tale they tell?
Not all that men look for comes to pass.
Two gates there are that give passage to fleeting Oneiroi;
one is made of horn, one of ivory.
The Oneiroi that pass through sawn ivory are deceitful,
bearing a message that will not be fulfilled;
those that come out through polished horn have truth behind them,
to be accomplished for men who see them."

Homer, Odyssey 19. 562 ff (Shewring translation).





Keras APIs



SEQUENTIAL MODELS

simple

suitable for most cases

linear order of layers

only one direction from input to output

FUNCTIONAL API

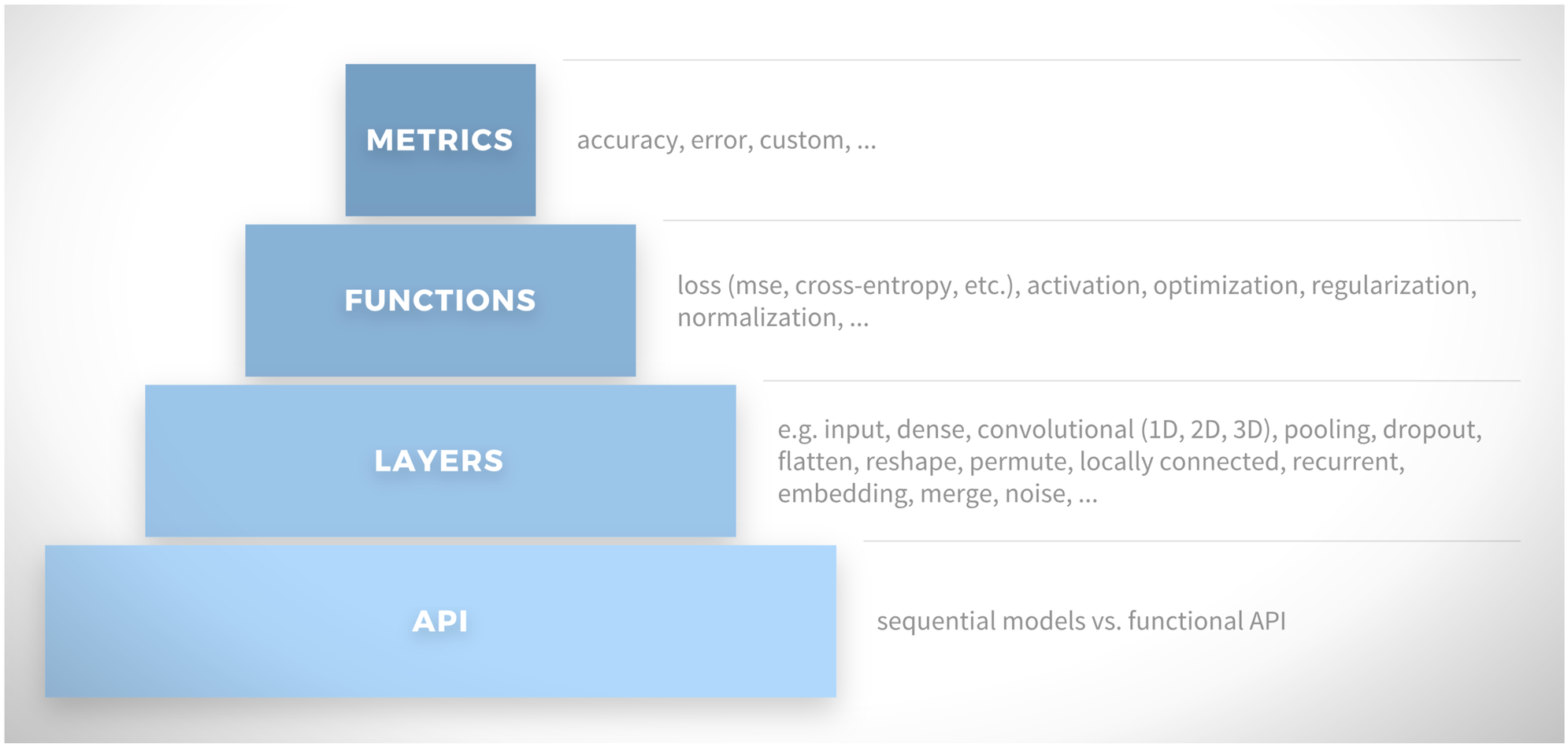
more complex

suitable for complex models

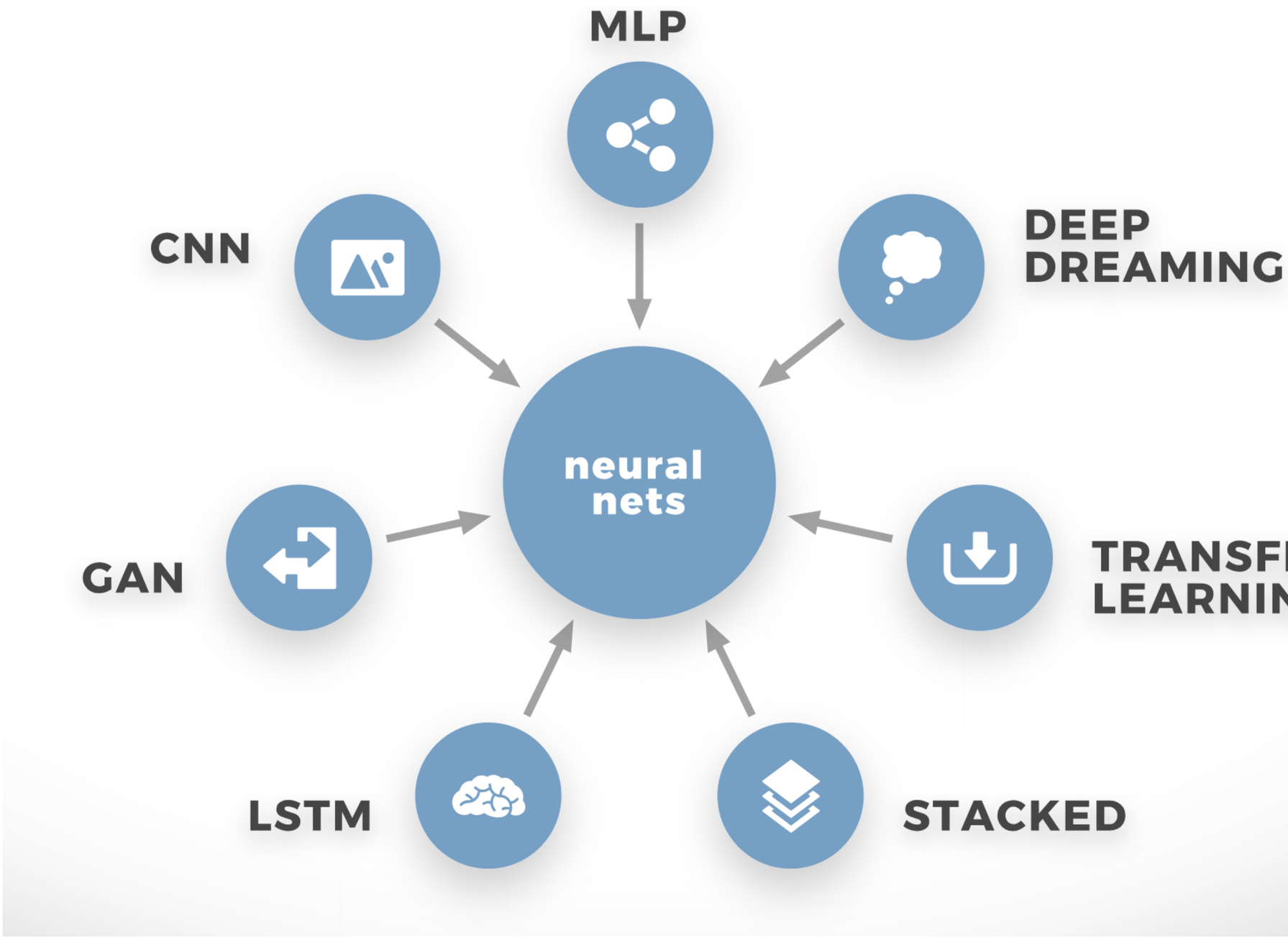
can have multiple in- or outputs

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

Keras layers



Endless possibilities

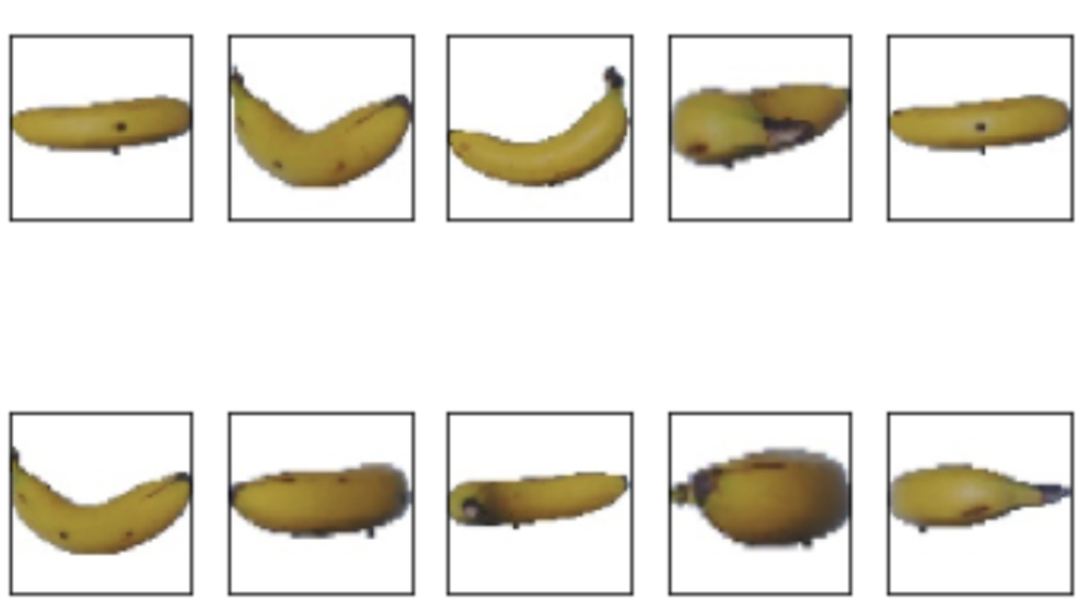


Let's get our hands dirty!

EDA

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

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



Apply pretrained nets

- VGG16

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

- Try out: ResNet50

Modify pretrained nets

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

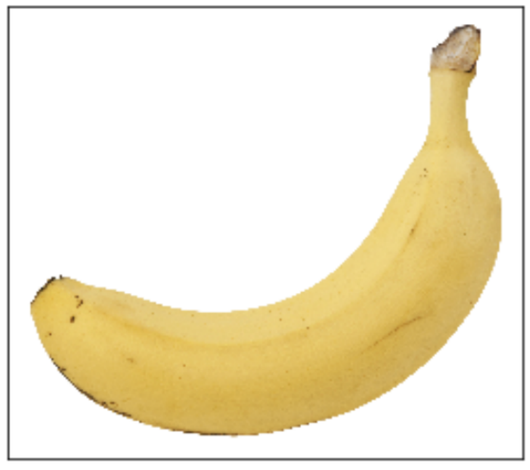
In [6]:
```
# important: exclude top layers
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(75, 75, 3))
#base_model.summary()
```

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

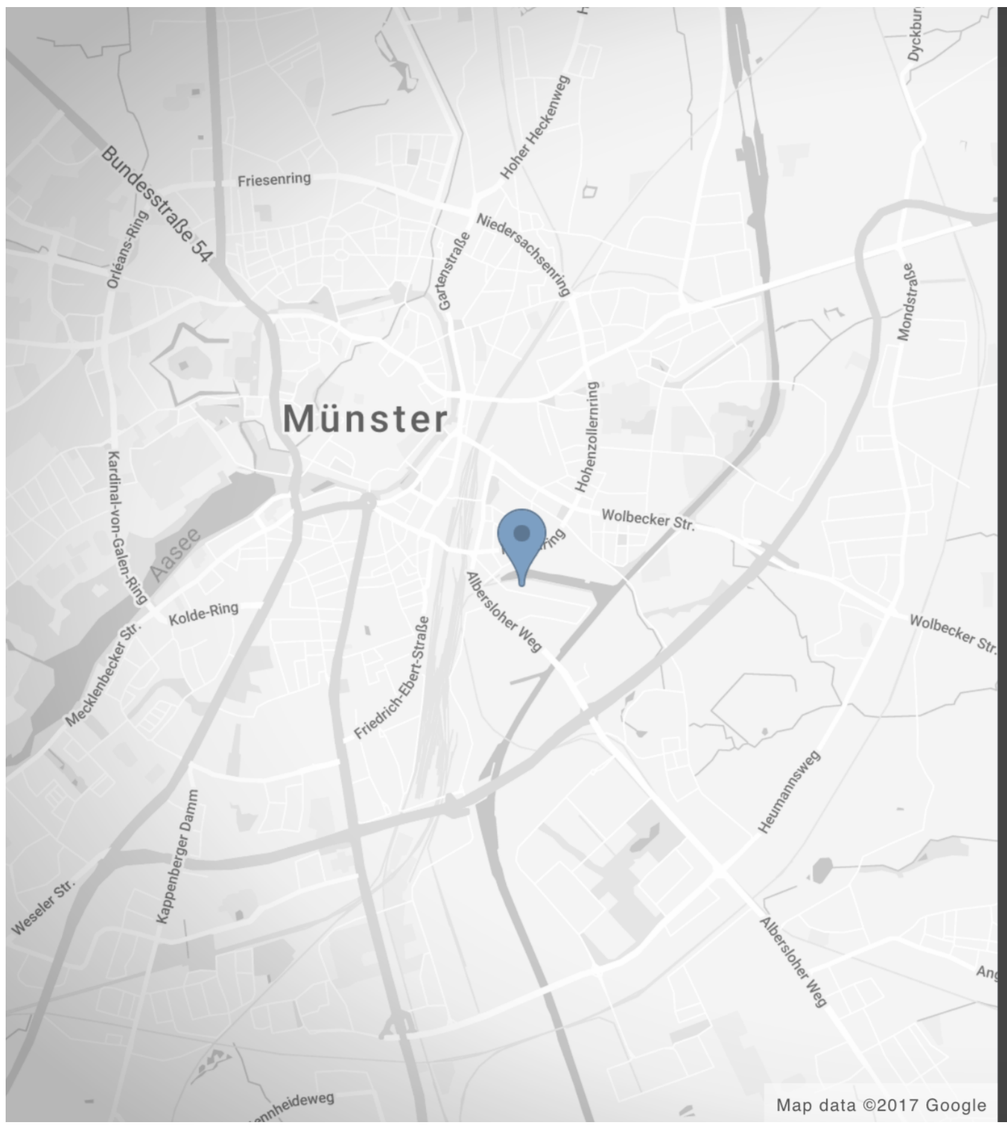
Build your own CNN

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

```
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