Deep Learning for Raiffeisenbank International

Jiří Materna

@mlcollegecom

@mlcollegecom

#mlcollege

About me

- Ph.D. in Natural Language Processing and Artificial Intelligence at Masaryk University
- 10 years at seznam.cz (last 8 years as Head Of Research)
- Founder and co-organizer of ML Prague
- Mentor at StartupYard
- ML Freelancer and consultant

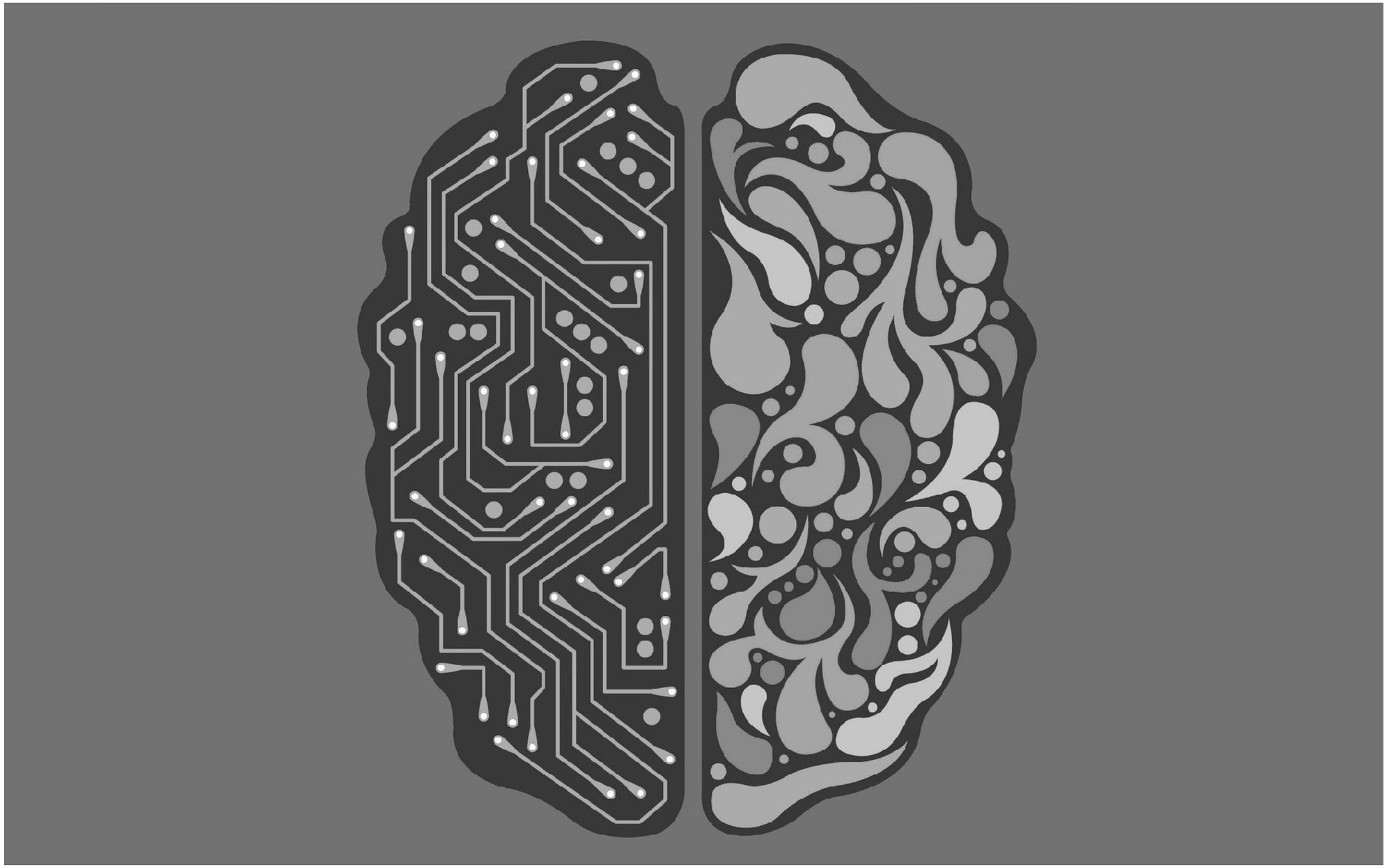
www.mlguru.com **www.mlprague.com** **www.mlcollege.com**

Outline

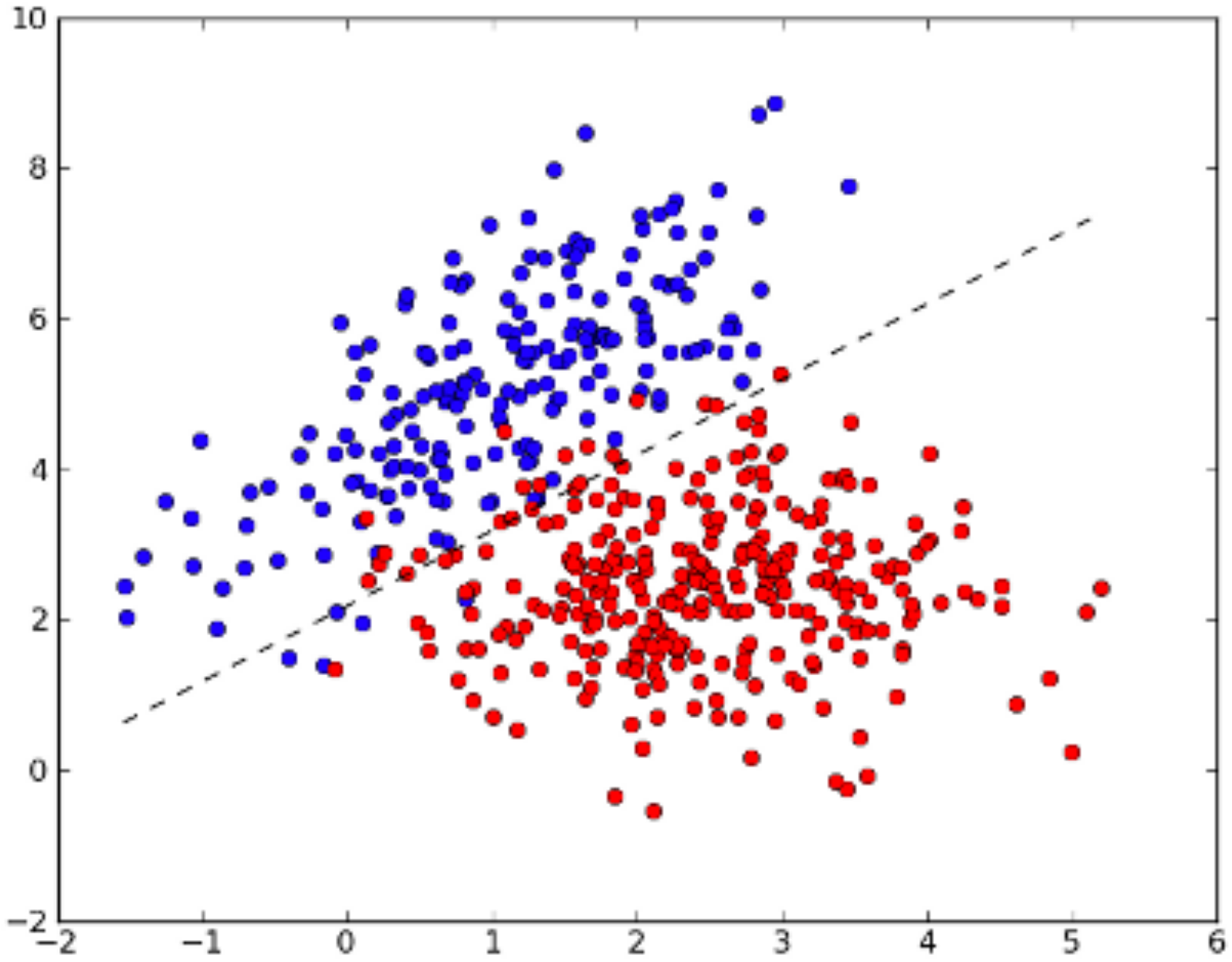
Day 1

- Scikit-learn tutorial
- Practical classification task
- Practical regression task
- Introduction to neural networks
- Methods for training neural networks
- Keras tutorial
- Practical classification and regression tasks solved using neural networks

What is (not) machine learning?

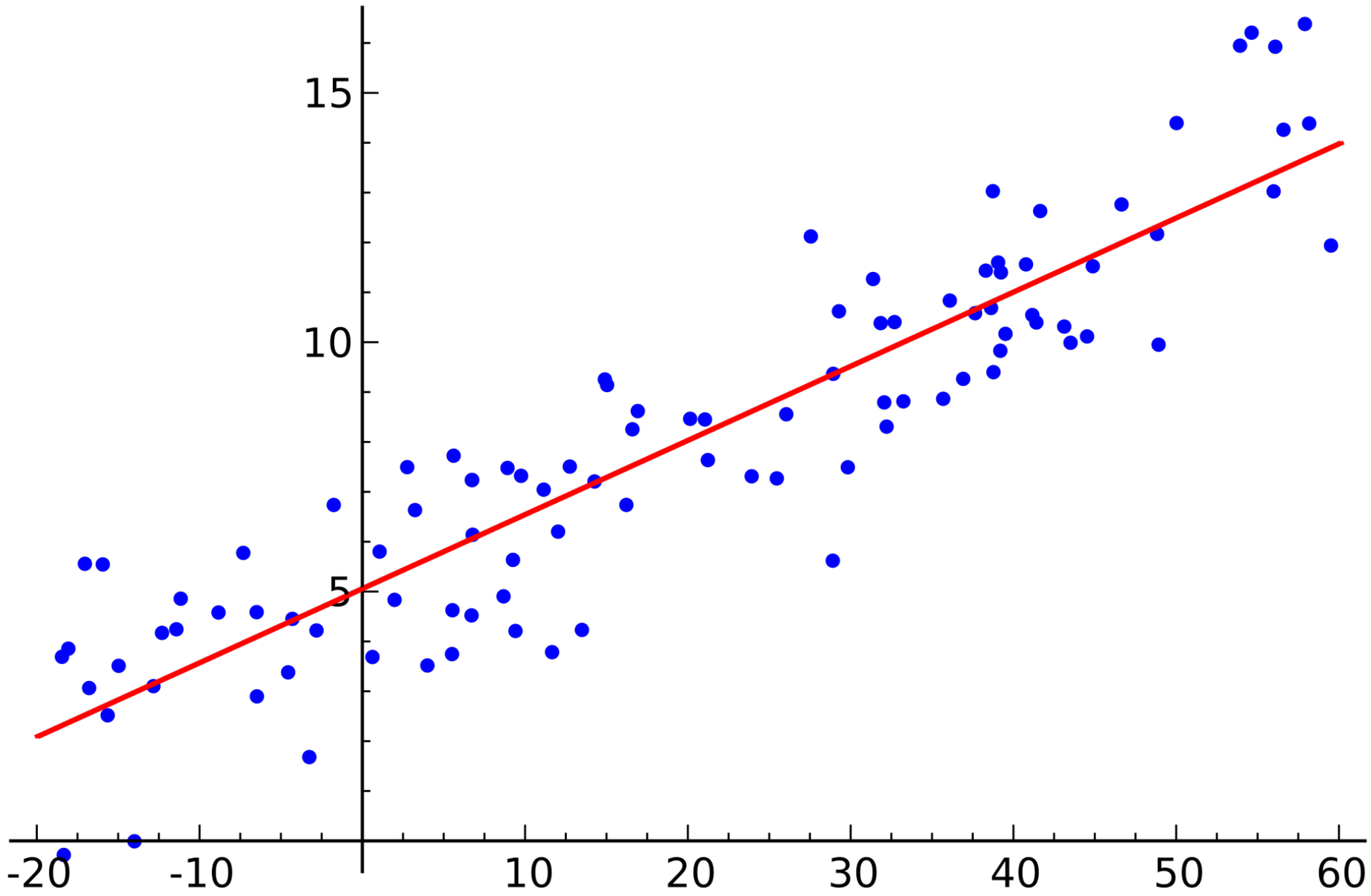


Classification



Source: http://mlpy.sourceforge.net

Regression



Source: www.wikipedia.org

Scikit-learn tutorial

http://scikit-learn.org/stable/

01 - Scikit learn tutorial

Classification task

02-Classification1-assignment.ipynb

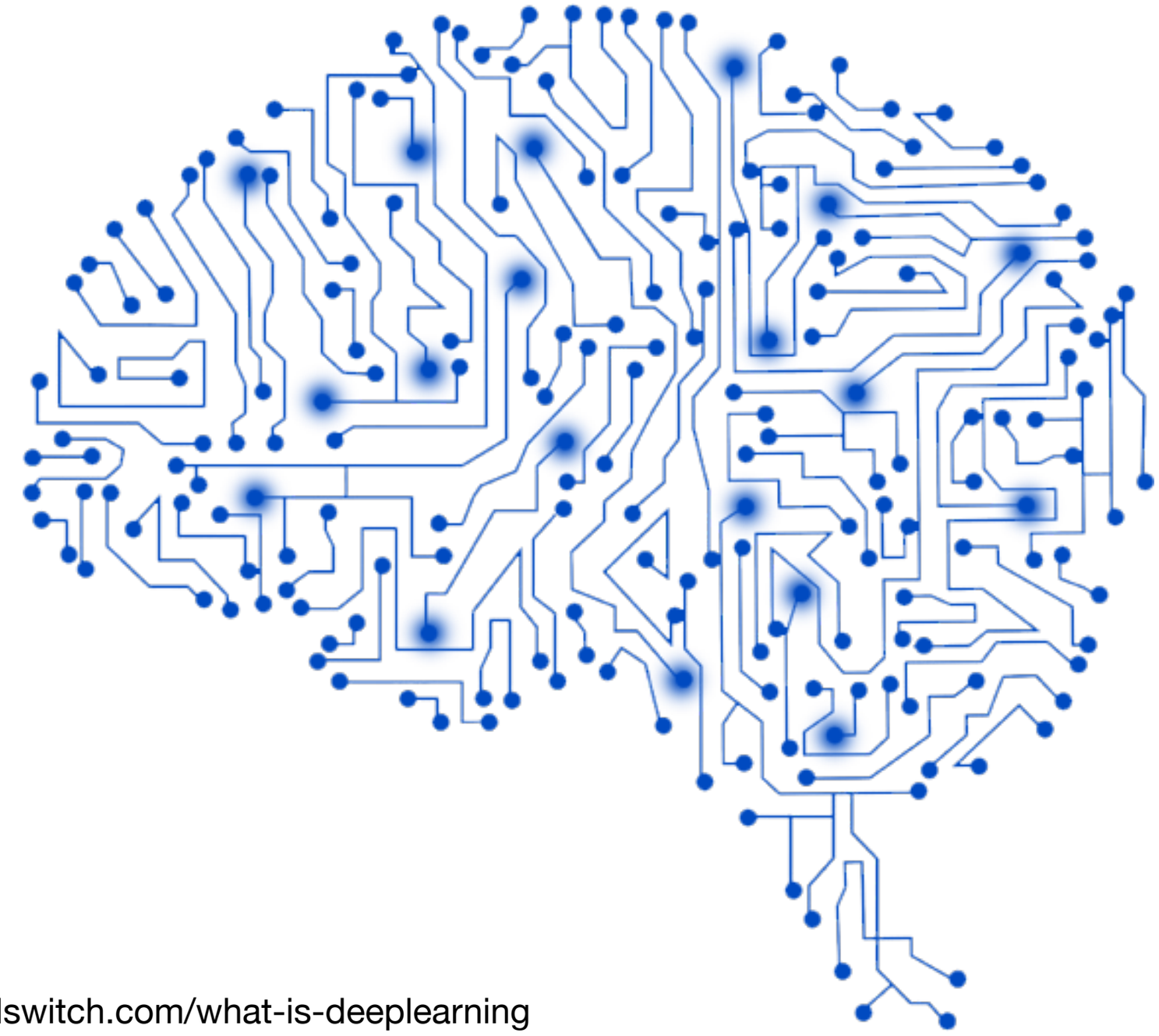
03-Classification2-assignment.ipynb

Regression task

04-Regression1-assignment.ipynb

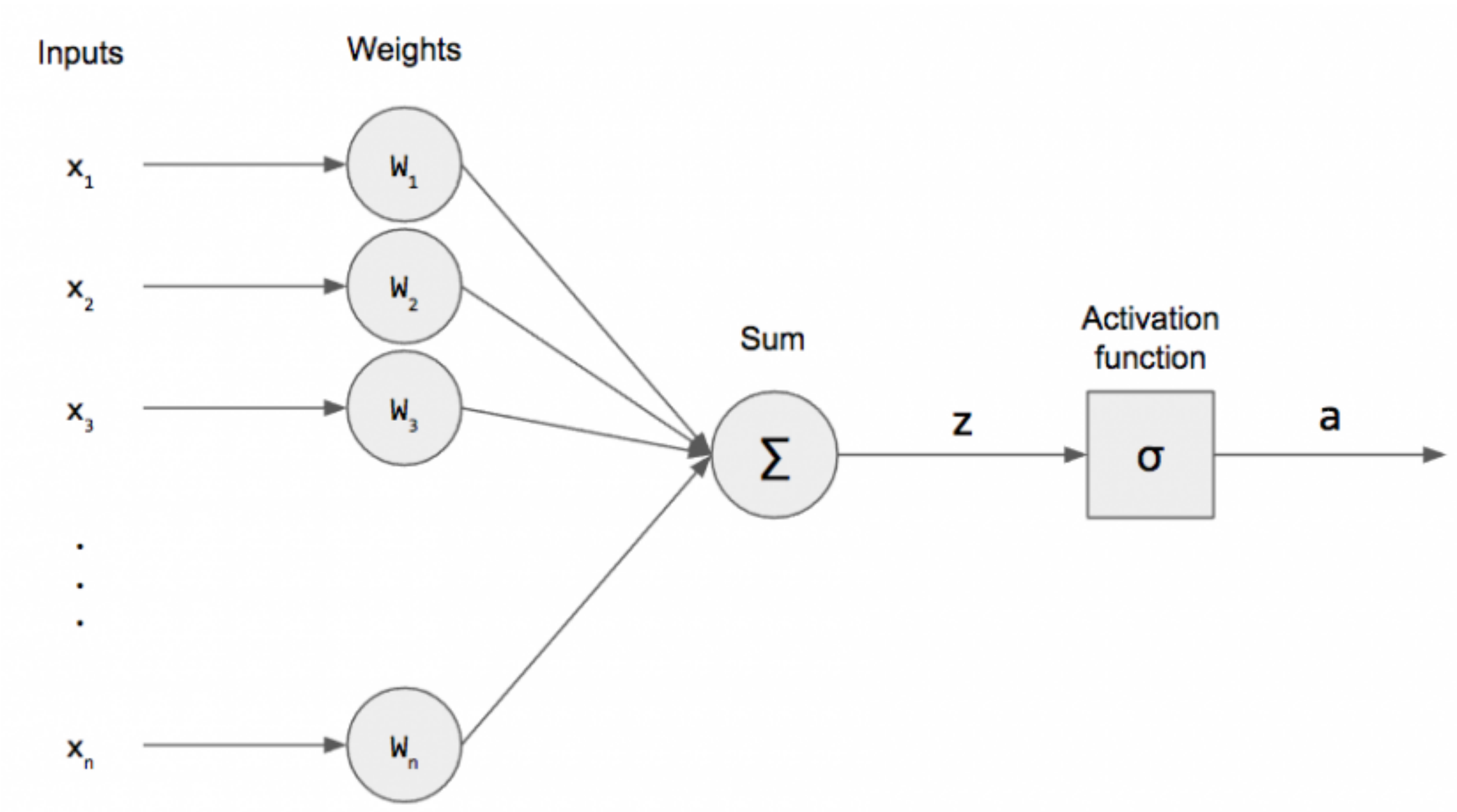
05-Regression2-assignment.ipynb

Neural networks and deep learning



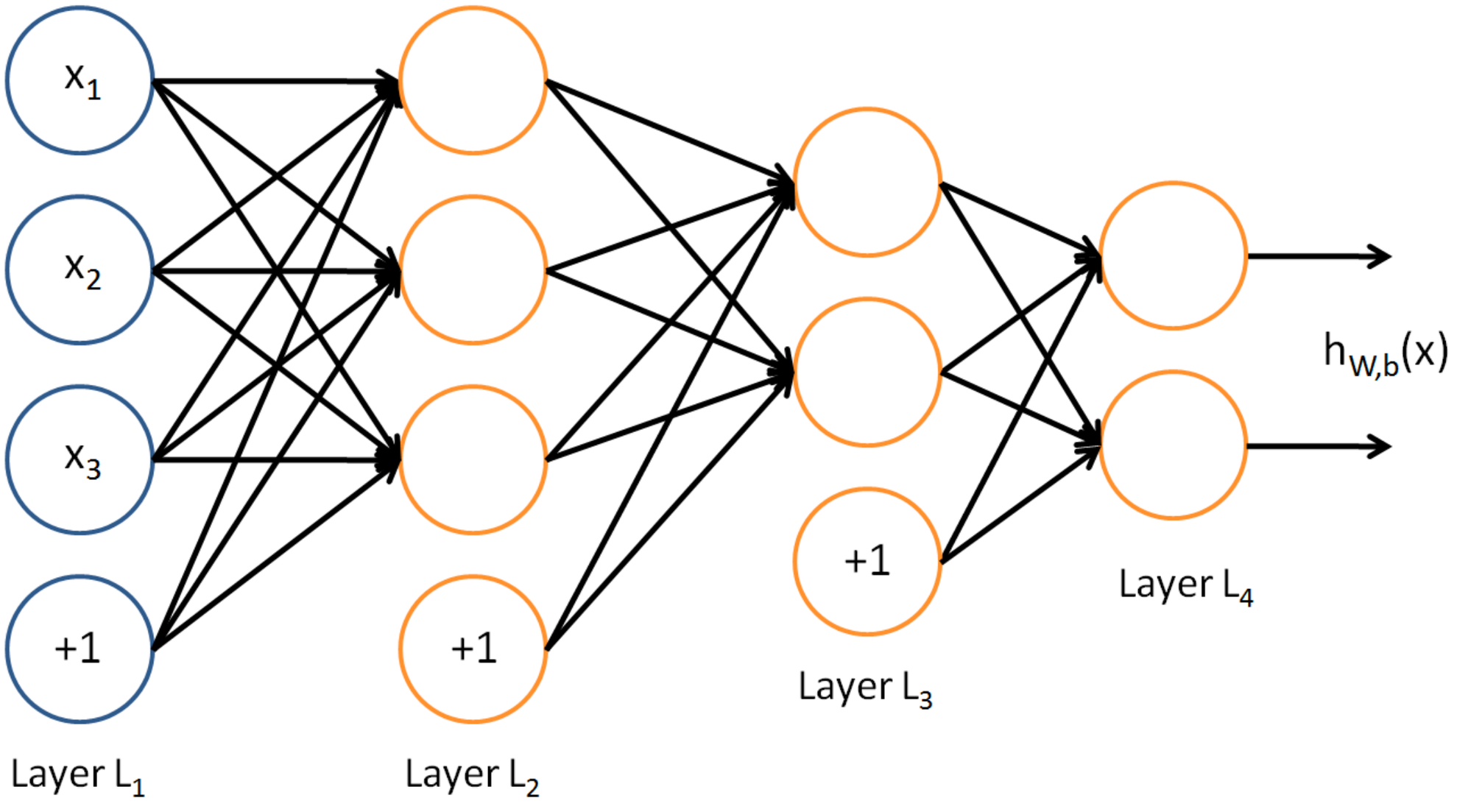
Source: http://www.rapidswitch.com/what-is-deeplearning

Perceptron



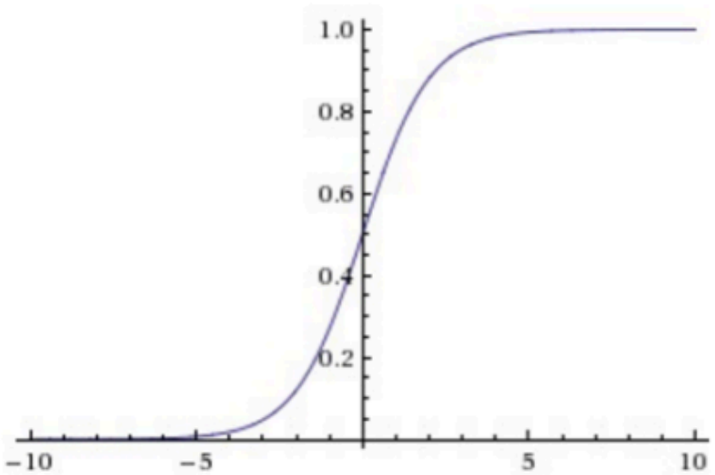
Source: https://pythonmachinelearning.pro

Multilayer Neural Networks

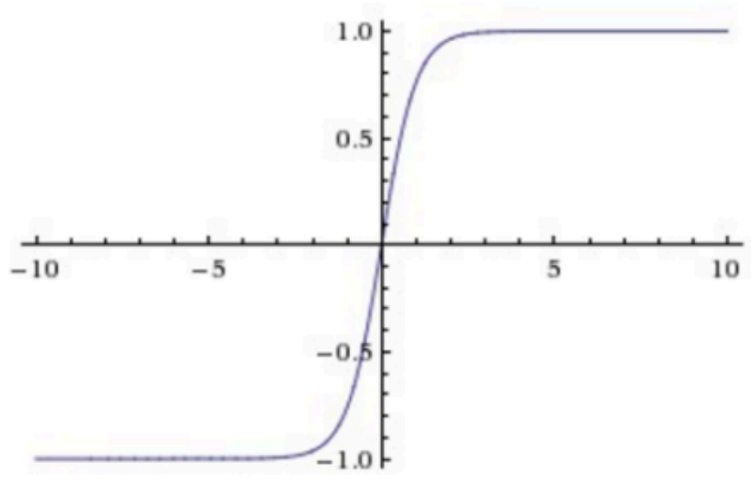


Source: http://ufldl.stanford.edu/

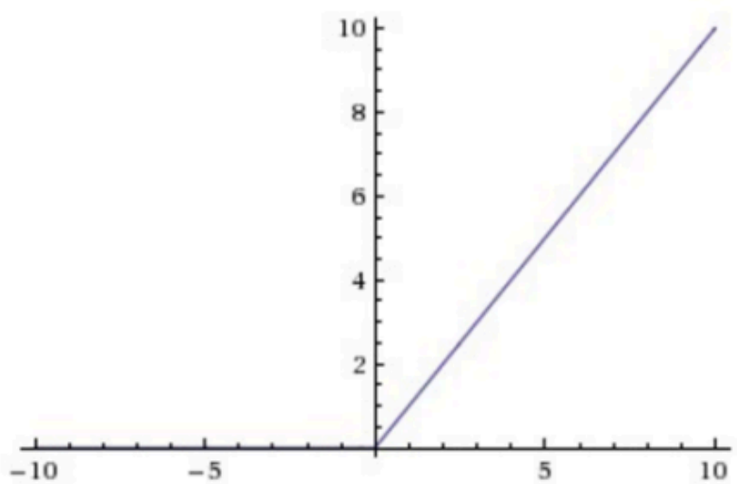
Activation functions



Sigmoid



tanh

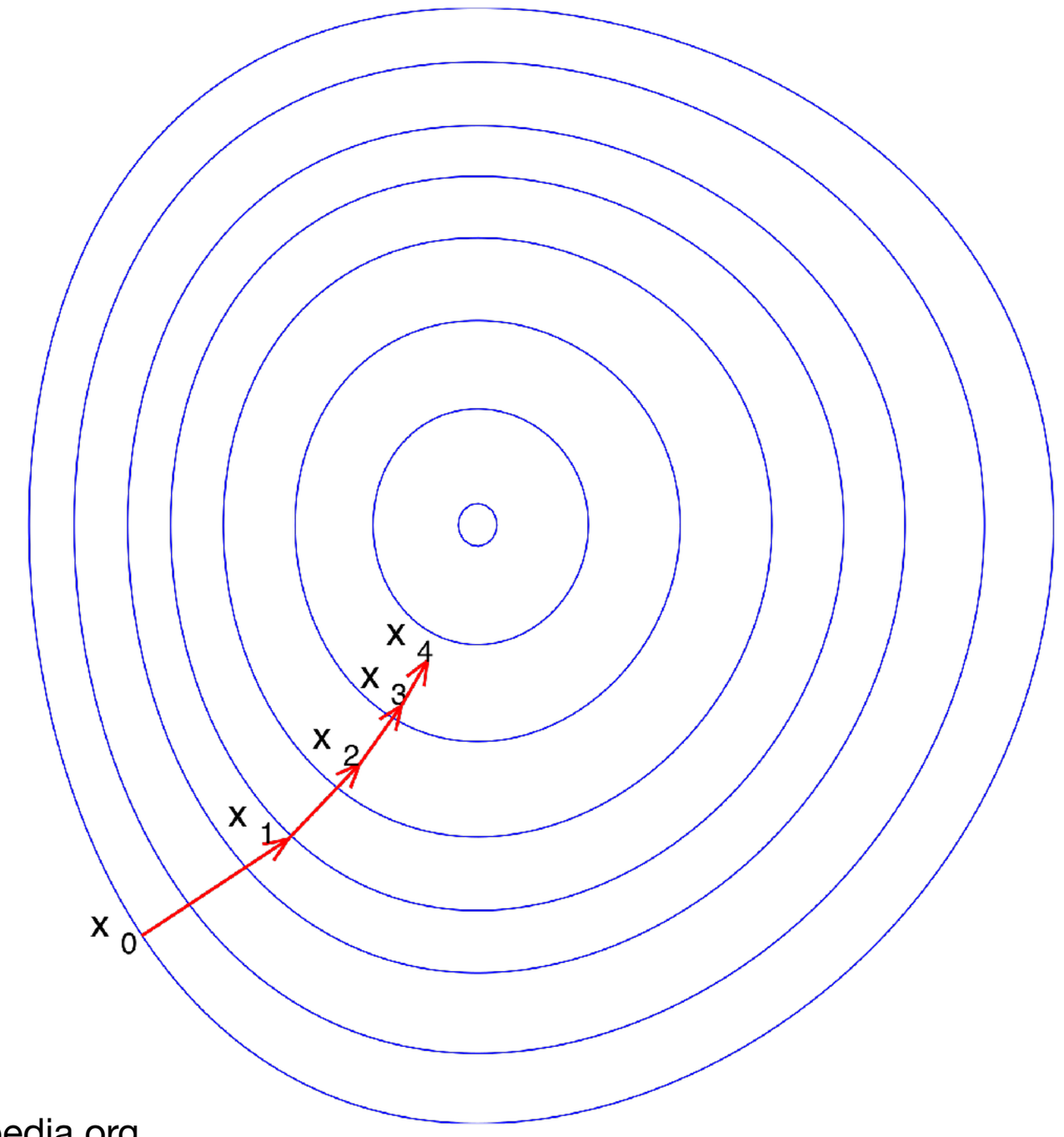


ReLU

Softmax:

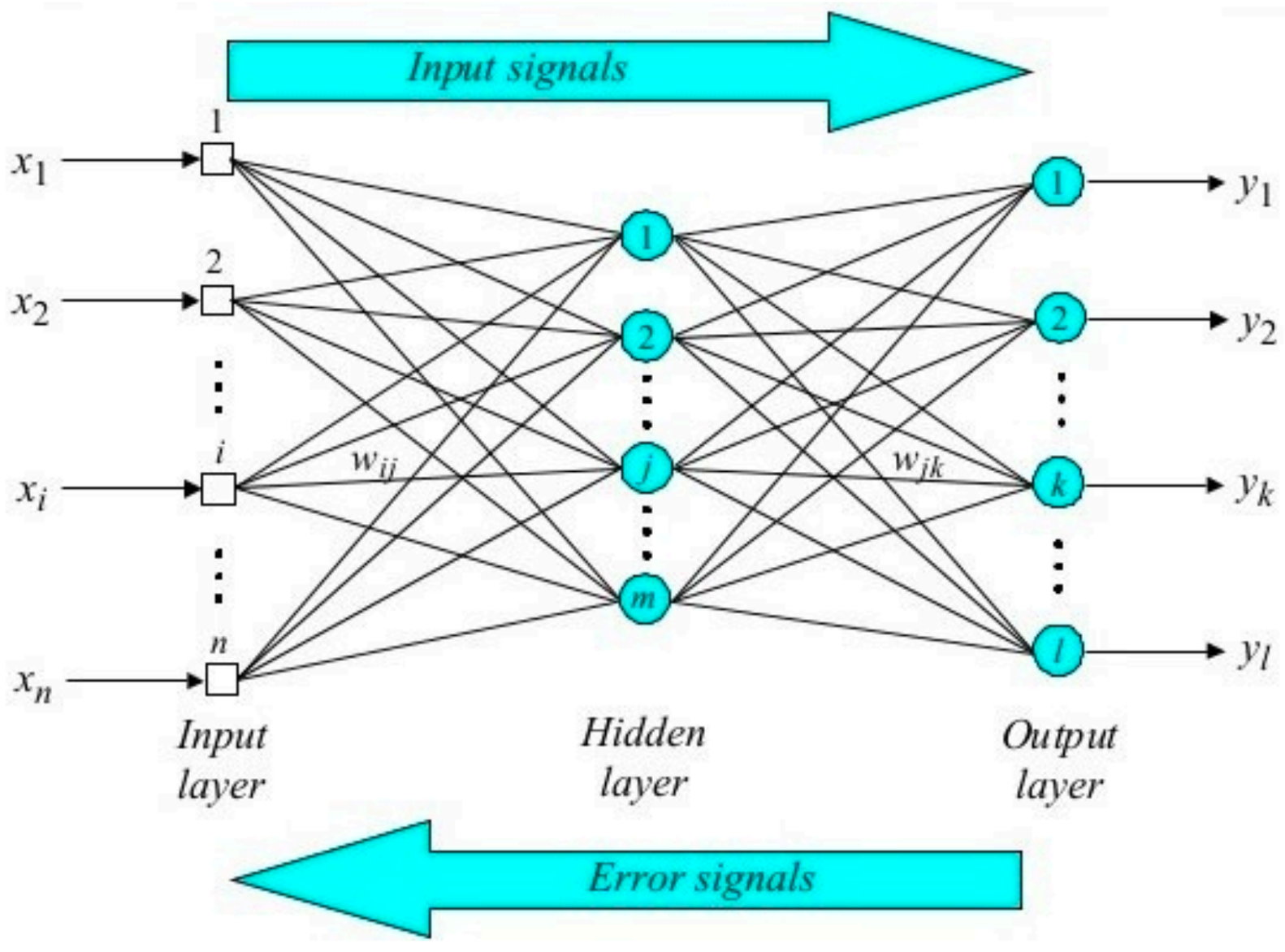
$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Steepest gradient descent



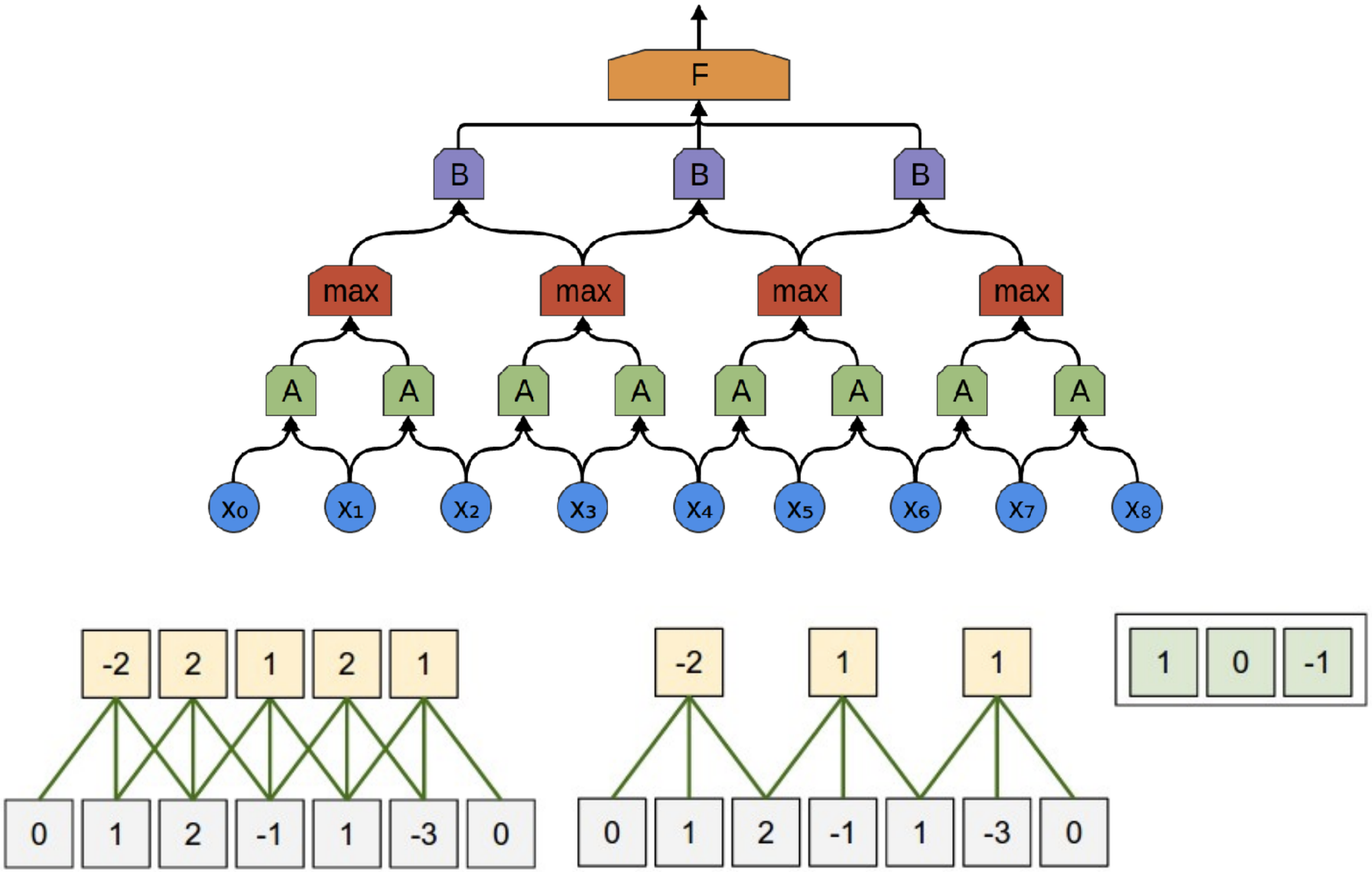
Source: https://www.wikipedia.org

Back propagation



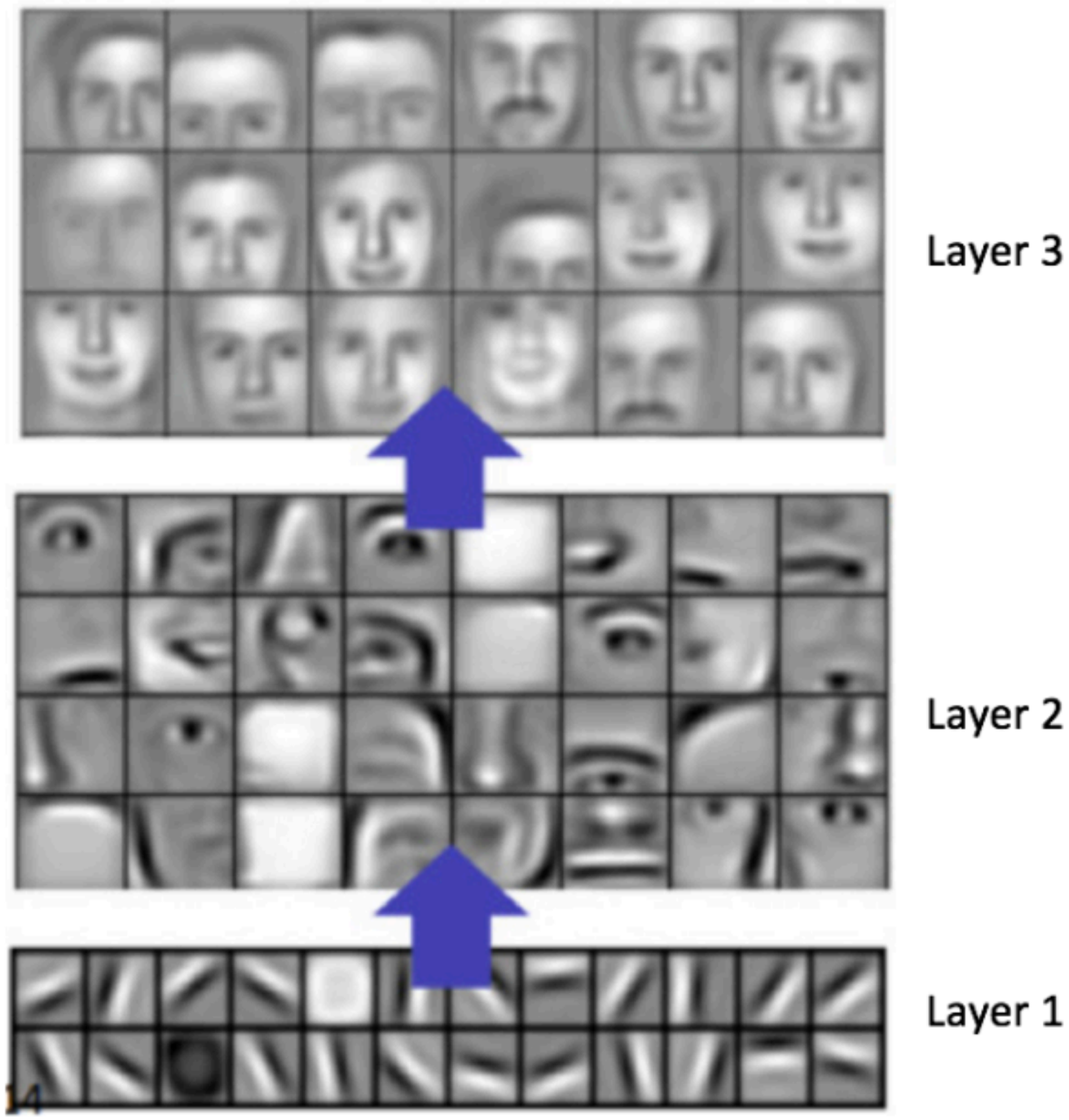
Source: https://www.quora.com/How-do-artificial-neural-networks-work

Convolution



Source: https://www.tensorflow.org

Weights visualization

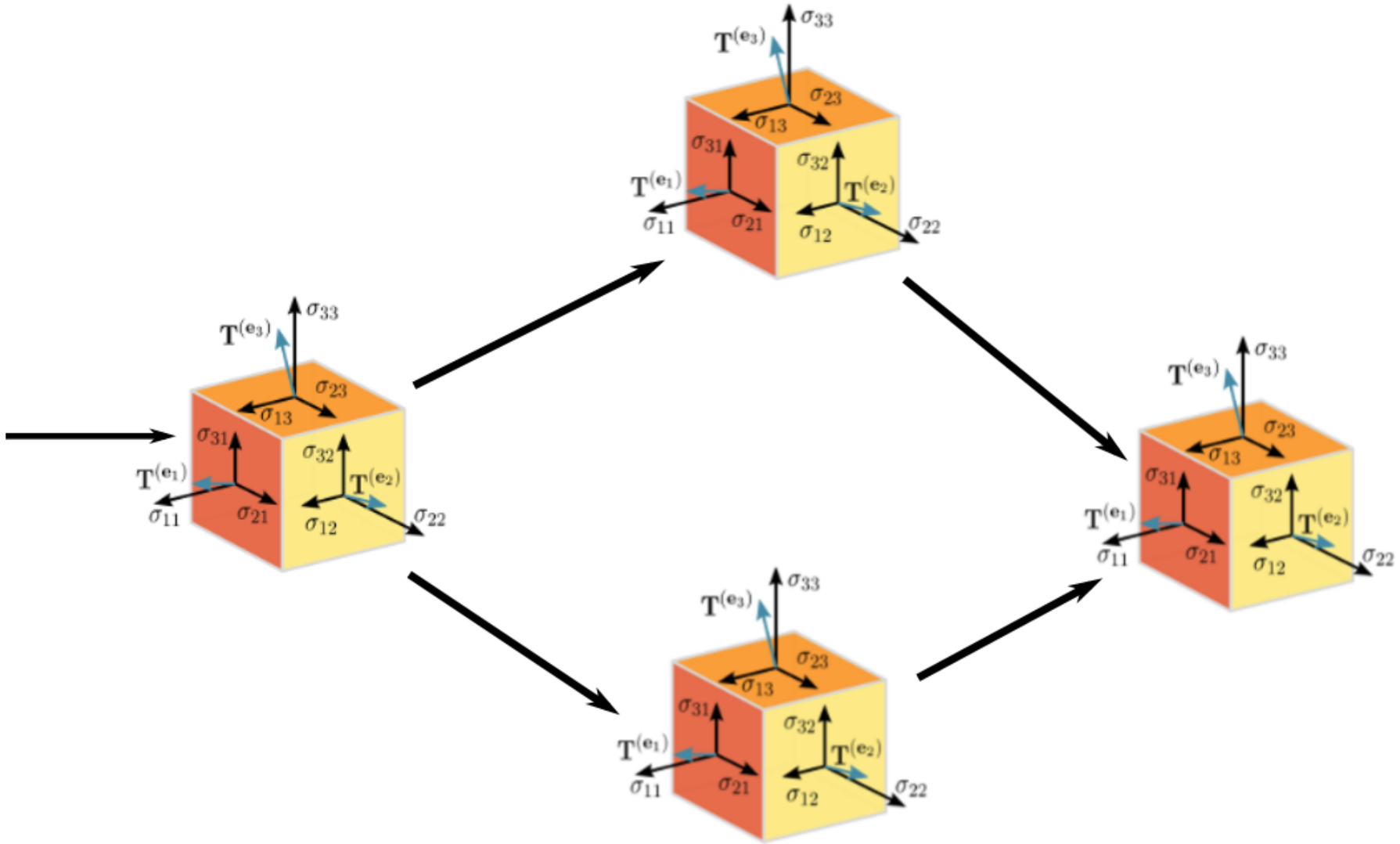


Important terms

- deep learning
- batch and mini-batch learning
- epoch

What is not TensorFlow

What is TensorFlow?



Keras tutorial

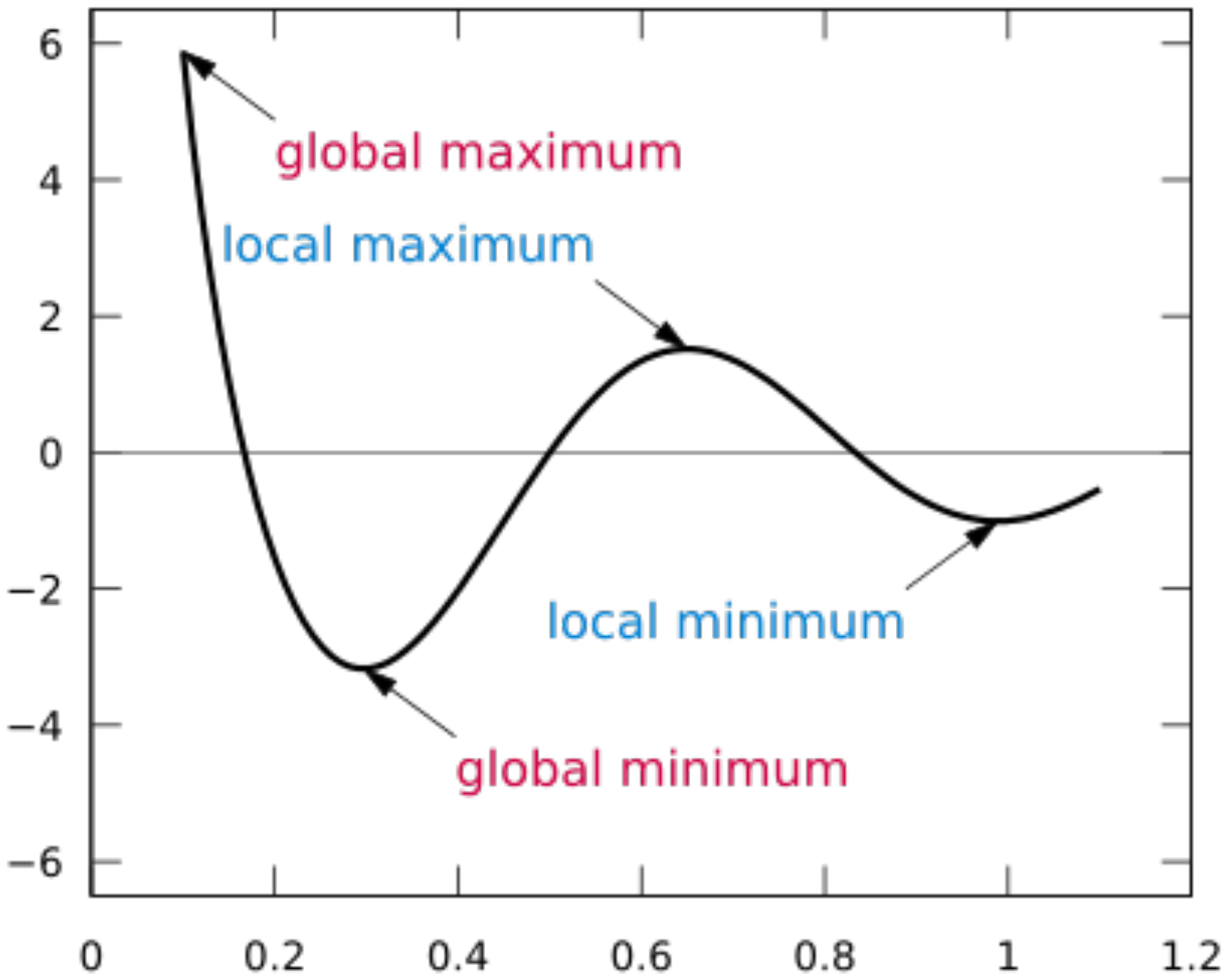
06-Keras-introduction.ipynb

Outline

Day 2

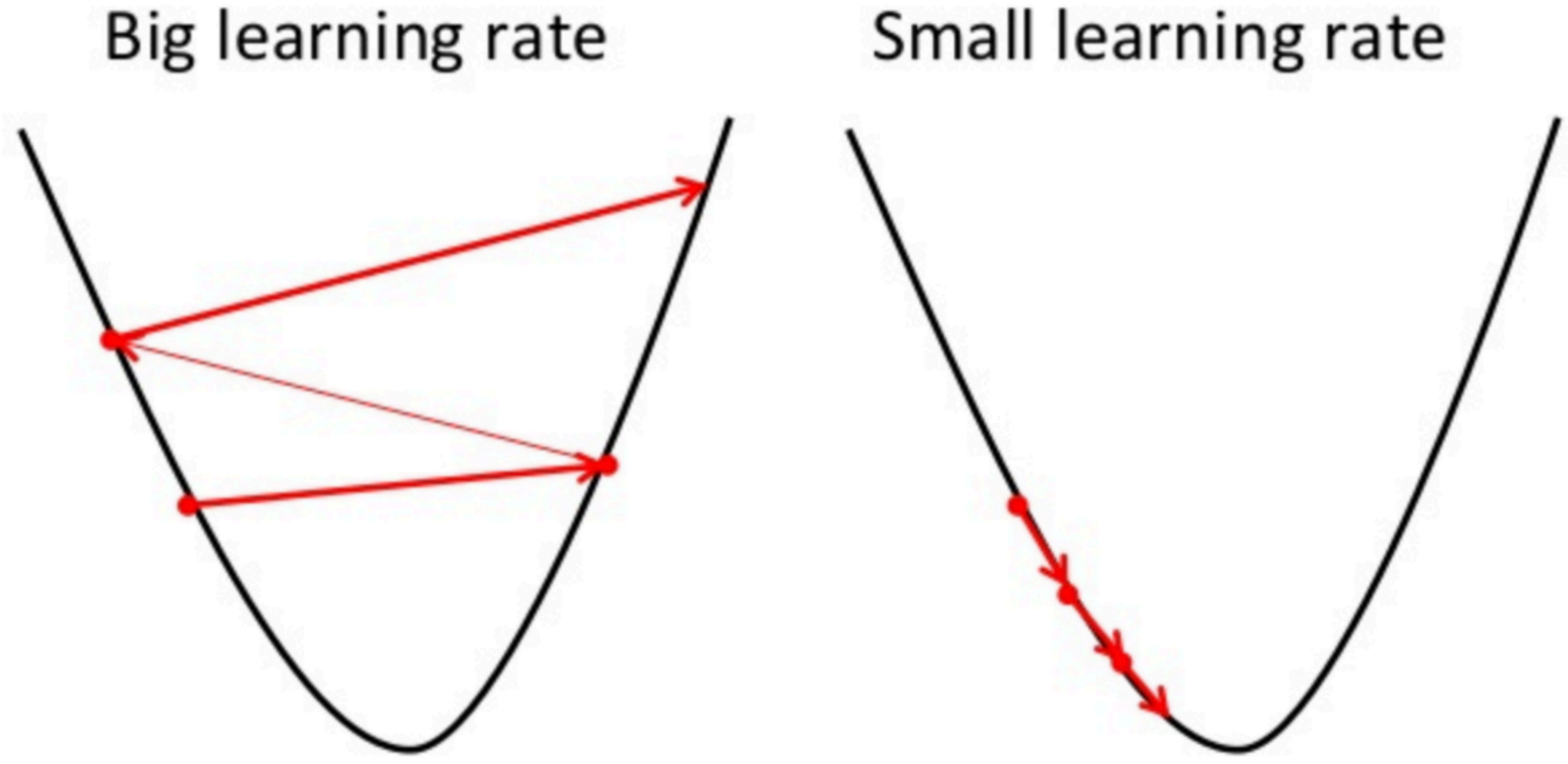
- Optimizers and their evolution
- Loss functions and their properties
- Practical classification task
- Practical regression task
- Initialization of weights in neural networks
- Normalization and Regularization in neural networks
- Functional model definition in Keras
- Practical example of a complex neural network

Parameter optimization strategies



https://heartbeat.fritz.ai/an-empirical-comparison-of-optimizers-for-machine-learning-models-b86f29957050

Learning rate tuning



Gradient Descent Variants

(Batch) Gradient Descent

$$w_{t+1} = w_t - \lambda \frac{\partial e(X, y)}{\partial w_t}$$

Stochastic Gradient Descent

$$w_{t+1} = w_t - \lambda \frac{\partial e(X^i, y^i)}{\partial w_t}$$

Mini-Batch Gradient Descent

$$w_{t+1} = w_t - \lambda \frac{\partial e(X^{(i,i+n)}, y^{(i,i+n)})}{\partial w_t}$$

Momentum and Nesterov Accelerated Gradient

Naive momentum

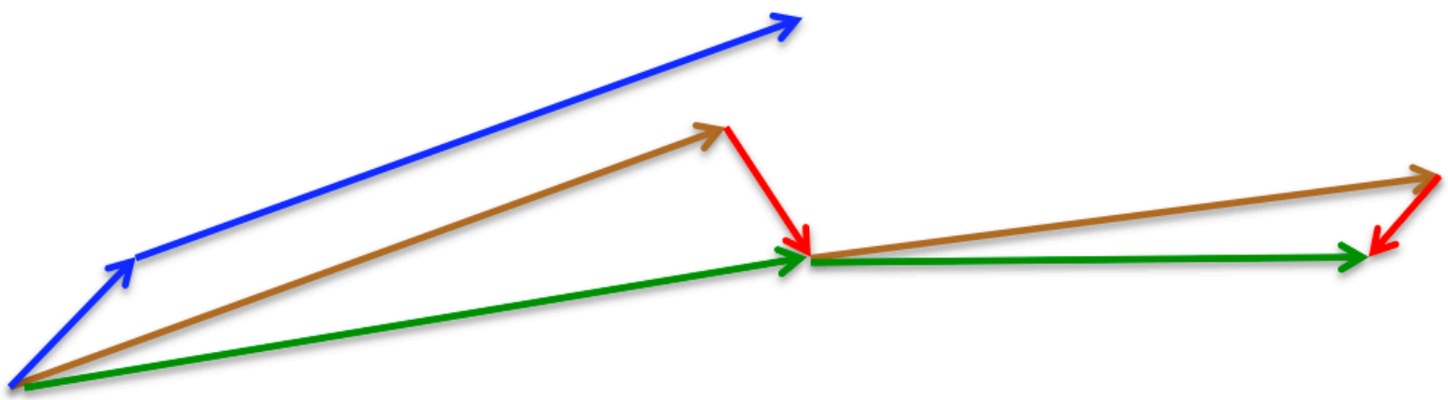
$$v_t = \gamma v_{t-1} + \lambda \frac{\partial e(w_t)}{\partial w_t}$$

$$w_{t+1} = w_t - v_t$$

Nesterov Accelerated Gradient

$$v_t = \gamma v_{t-1} + \lambda \frac{\partial e(w_t - \gamma v_{t-1})}{\partial w_t}$$

$$w_{t+1} = w_t - v_t$$



https://ruder.io/optimizing-gradient-descent/index.html#nesterovacceleratedgradient

Adaptive Gradient Algorithms

Adagrad

$$w_{t+1} = w_t - \frac{\lambda}{\sqrt{\sum_{i=1}^{t} g_i^2 + \epsilon}} g_t$$

$$g_i = \frac{\partial e(w_i)}{\partial w_i}$$

RMSProp

$$w_{t+1} = w_t - \frac{\lambda}{\sqrt{\mathbb{E}[g^2]_t - \epsilon}} g_t^2$$

$$\mathbb{E}[g^2]_t = \gamma \mathbb{E}[g^2]_{t-1} + (1-\gamma) g_t^2$$

Adam and Nadam

Adam Combination of RMSProp with momentum

Nadam Combination of RMSProp with Nesterov momentum

Loss functions for deep learning

Mean Squared Error

$$MSE = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}$$

Mean Absolute Error

$$MSE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}$$

Cross Entropy (Negative Log Likelihood)

Categorical Cross Entropy $$CCE = -\frac{\sum_{i=1}^{n}\sum_{j=1}^{c} y_{i,j} log(\hat{y}_{i,j})}{n}$$

Binary Cross Entropy

$$BCE = -\frac{\sum_{i=1}^{n}[y_{i,j} log(\hat{y}_{i,j}) + (1 - y_{i,j}) log(\hat{y}_{i,j})]}{n}$$

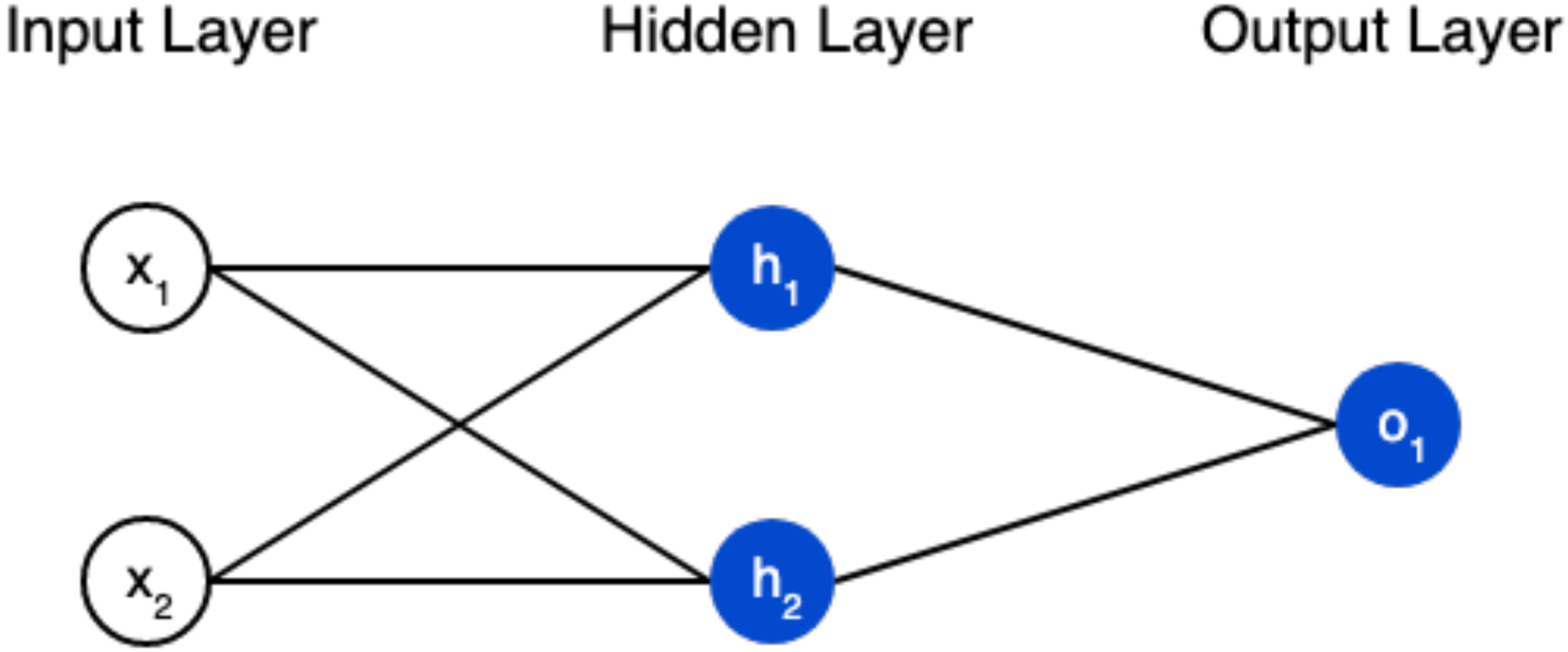
Reimplementation of the classification and regression task using NN

07-Classification-nn-assignment.ipynb

08-Regression-nn-assignment.ipynb

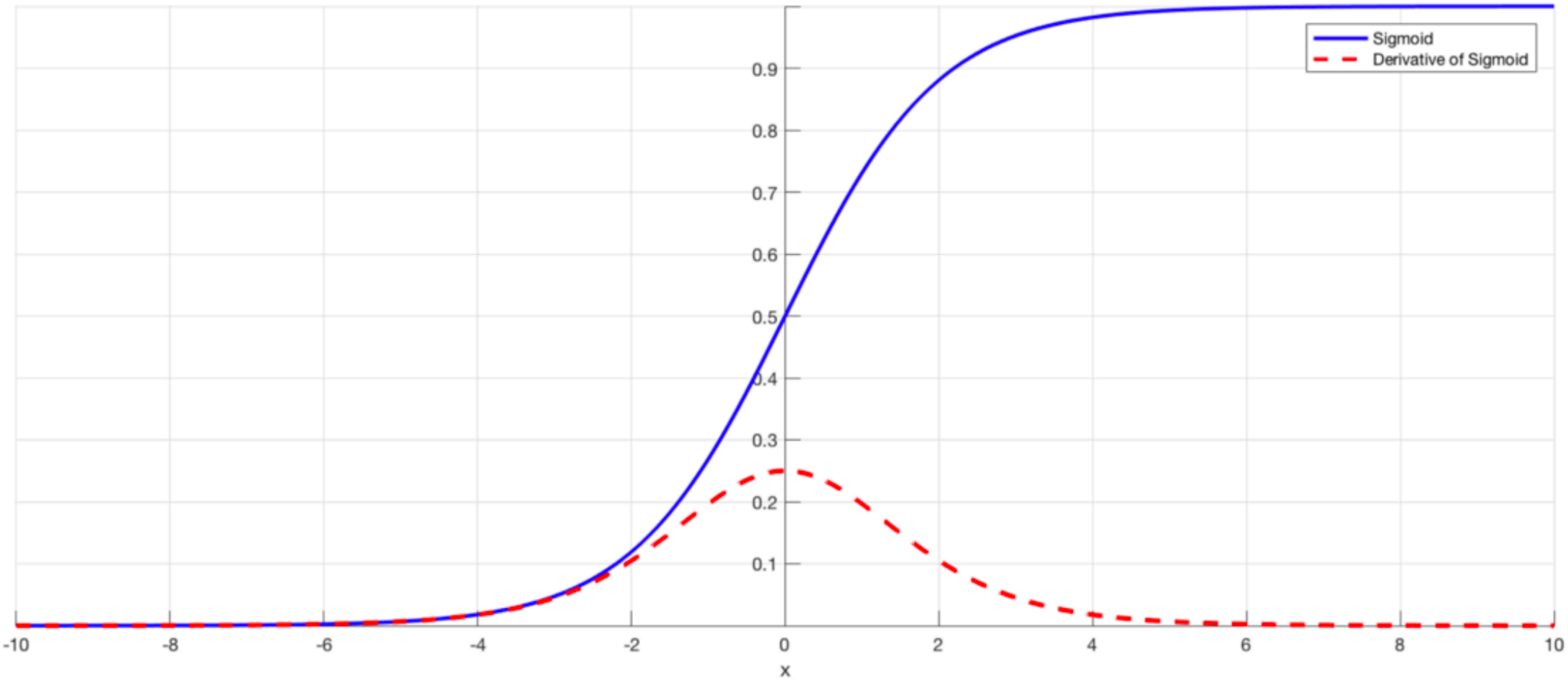
Weight initialization

Zero or constant initialization



https://victorzhou.com/blog/intro-to-neural-networks/

Too low or too high initialization



https://towardsdatascience.com/derivative-of-the-sigmoid-function-536880cf918e

Xavier and He initializers

1. The mean of the activations should be zero
2. The variance of the activations should stay the same across every layer

Xavier (Glorot) initialization for tanh

$$\mathbf{W}^l \sim \mathcal{N}(\mu = 0, \sigma^2 = \frac{1}{n^{l-1}})$$

$$b^l = 0$$

He (Kaiming) initialization for relu

$$\mathbf{W}^l \sim \mathcal{N}(\mu = 0, \sigma^2 = \frac{2}{n^{l-1}})$$

$$b^l = 0$$

Experiment with various initializations for a deep network

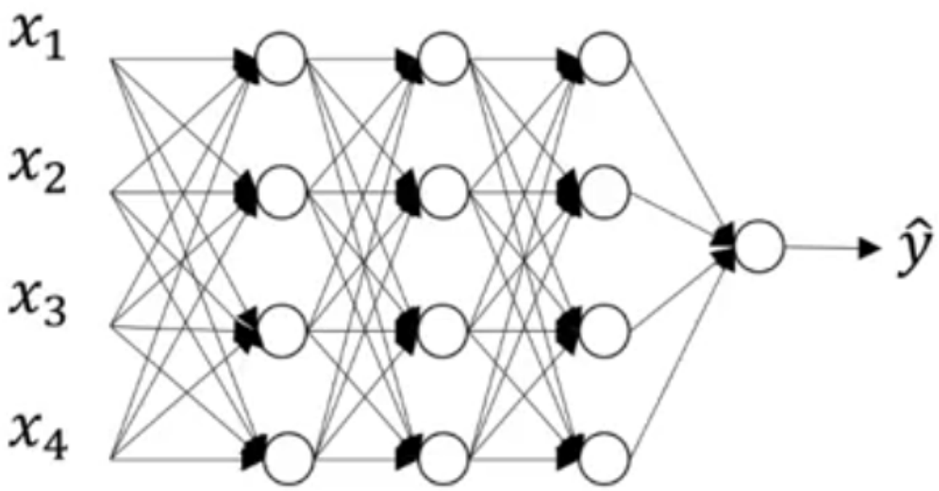
08-Regression-nn-assignment.ipynb

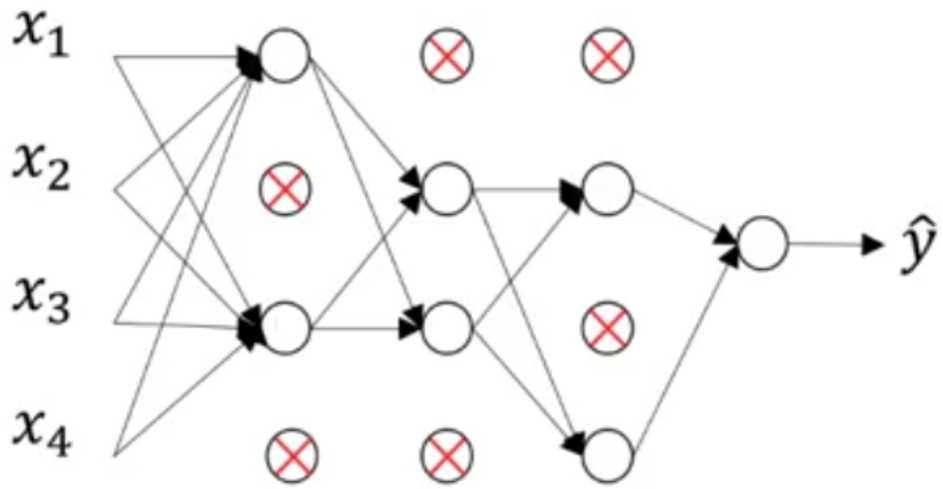
L2 Regularization in deep learning

$$cost(w^1, b^1, \ldots, w^L, b^L) = \frac{1}{n}\sum_{i=1}^{n} Loss(y_i, \hat{y}_i) + \frac{\lambda}{2n}\sum_{l=1}^{L} \|w^l\|_F^2$$

Frobenius norm $$||w||_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} |w_{i,j}|^2}$$

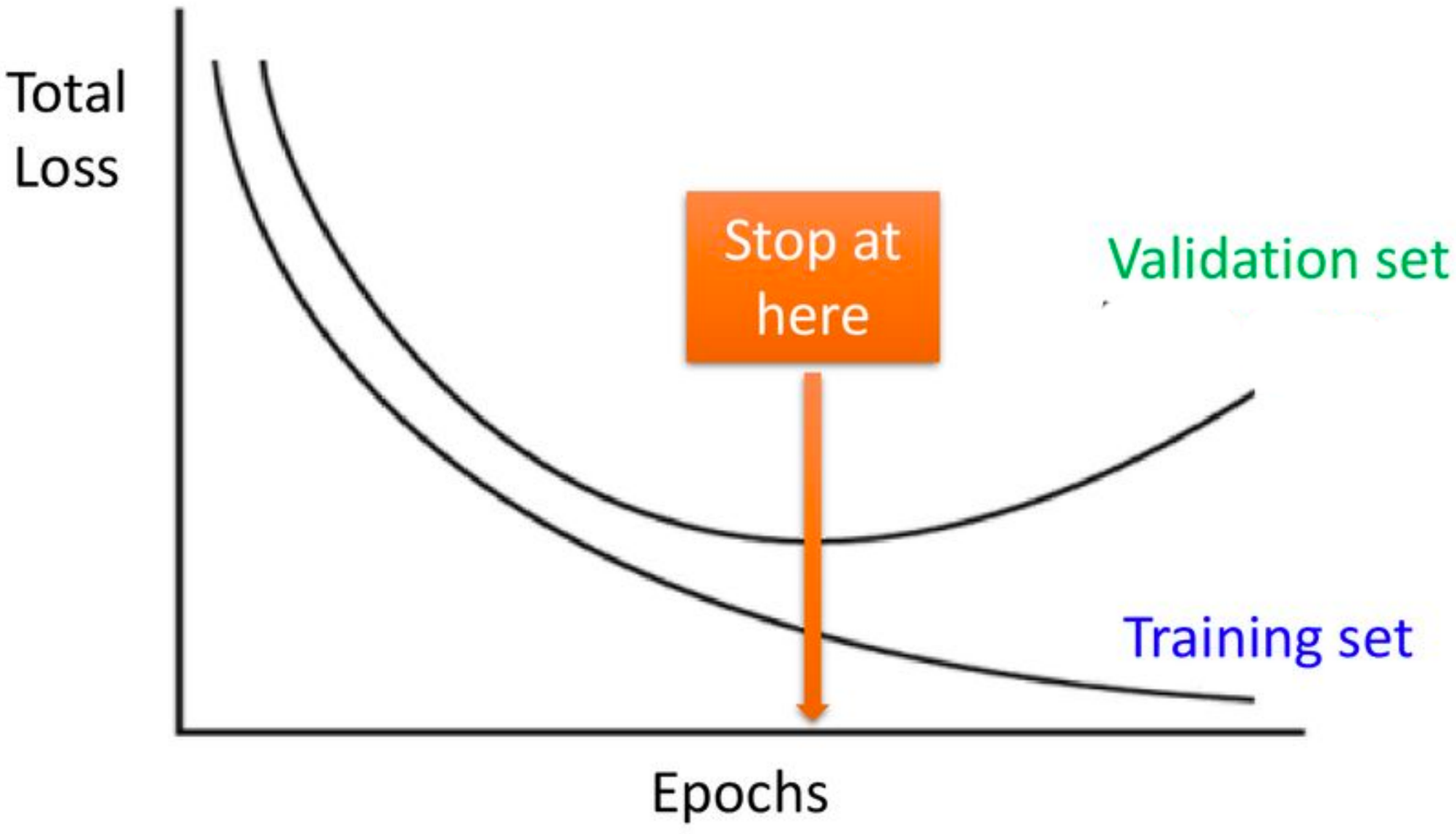
Dropout





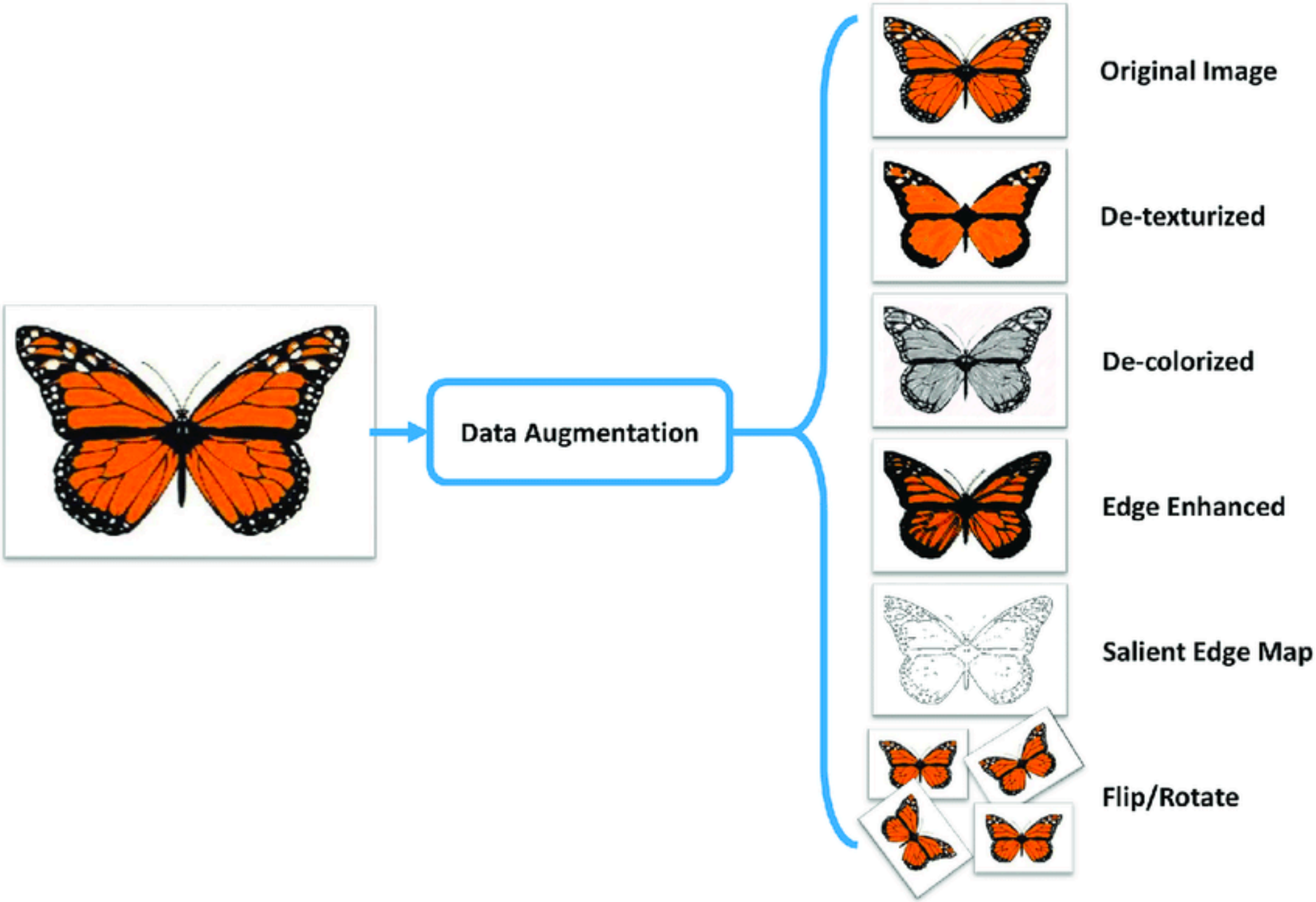
https://towardsdatascience.com/how-to-improve-a-neural-network-with-regularization-8a18ecda9fe3

Early stopping



https://keras.io/getting_started/intro_to_keras_for_engineers/

Data augmentation



https://medium.com/secure-and-private-ai-writing-challenge/data-augmentation-increases-accuracy-of-your-model-but-how-aa1913468722

Batch normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

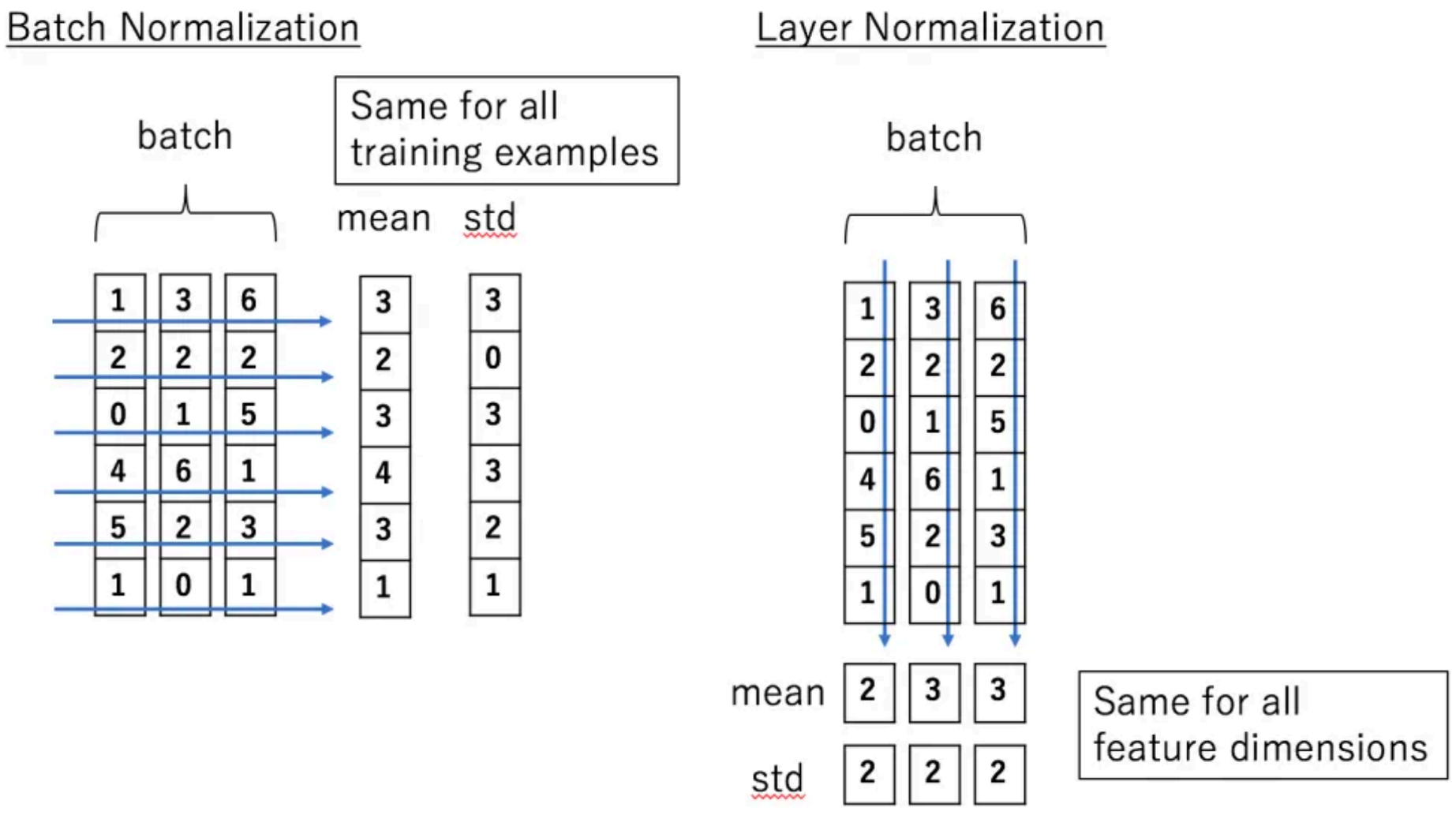
$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

https://arxiv.org/abs/1502.03167

Layer normalization



https://mlexplained.com/2018/01/13/weight-normalization-and-layer-normalization-explained-normalization-in-deep-learning-part-2/

Functional API in Keras

```
# Sequential model
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(10, input_shape=(10,),activation='relu')
model.add(Dense(20, activation='relu')
model.add(Dense(10, activation='relu')
model.add(Dense(1, activation='sigmoid')
```

```
# Functional model
from keras.models import Model
from keras.layers import Input, Dense

visible = Input(shape=(10,))
hidden1 = Dense(10, activation='relu')(visible)
hidden2 = Dense(20, activation='relu')(hidden1)
hidden3 = Dense(10, activation='relu')(hidden2)
output = Dense(1, activation='sigmoid')(hidden3)
model = Model(inputs=visible, outputs=output)
```

Shared Input

```
# Shared Input Layer
from keras.utils import plot_model
from keras.models import Model
from keras.layers import Input, Dense, Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.pooling import MaxPooling2D
from keras.layers.merge import concatenate
# input layer
visible = Input(shape=(64,64,1))
# first feature extractor
conv1 = Conv2D(32, kernel_size=4, activation='relu')(visible)
pool1 = MaxPooling2D(pool_size=(2, 2))(conv1)
flat1 = Flatten()(pool1)
# second feature extractor
conv2 = Conv2D(16, kernel_size=8, activation='relu')(visible)
pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)
flat2 = Flatten()(pool2)
# merge feature extractors
merge = concatenate([flat1, flat2])
# interpretation layer
hidden1 = Dense(10, activation='relu')(merge)
# prediction output
output = Dense(1, activation='sigmoid')(hidden1)
model = Model(inputs=visible, outputs=output)
```

Multiple inputs (outputs)

```python
# Multiple Inputs
from keras.utils import plot_model
from keras.models import Model
from keras.layers import Input
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.pooling import MaxPooling2D
from keras.layers.merge import concatenate
# first input model
visible1 = Input(shape=(64,64,1))
conv11 = Conv2D(32, kernel_size=4, activation='relu')(visible1)
pool11 = MaxPooling2D(pool_size=(2, 2))(conv11)
conv12 = Conv2D(16, kernel_size=4, activation='relu')(pool11)
pool12 = MaxPooling2D(pool_size=(2, 2))(conv12)
flat1 = Flatten()(pool12)
# second input model
visible2 = Input(shape=(32,32,3))
conv21 = Conv2D(32, kernel_size=4, activation='relu')(visible2)
pool21 = MaxPooling2D(pool_size=(2, 2))(conv21)
conv22 = Conv2D(16, kernel_size=4, activation='relu')(pool21)
pool22 = MaxPooling2D(pool_size=(2, 2))(conv22)
flat2 = Flatten()(pool22)
# merge input models
merge = concatenate([flat1, flat2])
# interpretation model
hidden1 = Dense(10, activation='relu')(merge)
hidden2 = Dense(10, activation='relu')(hidden1)
output = Dense(1, activation='sigmoid')(hidden2)
model = Model(inputs=[visible1, visible2], outputs=output)
```

Practical example on regularization and normalization

09-Normalization-and-regularization-assignment.ipynb

Outline

Day 3

- Neural Network architectures
- Convolutional NN and Transfer learning
- Recurrent neural networks (simple RNN, LSTM, GRU) and their examples
- Autoencoders and Restricted Boltzmann Machines and their applications
- Transformers
- U-Net
- Generative Adversarial Networks and their applications

Neural Network architectures design

Neural Network design best practices

- ⭐ **Start from simple architectures**
- ⭐ **Get inspiration from architectures for similar problems**
- ⭐ **Change one parameter only and then validate**

Most common architectures

Feed forward network

Convolutional network

Recurrent network

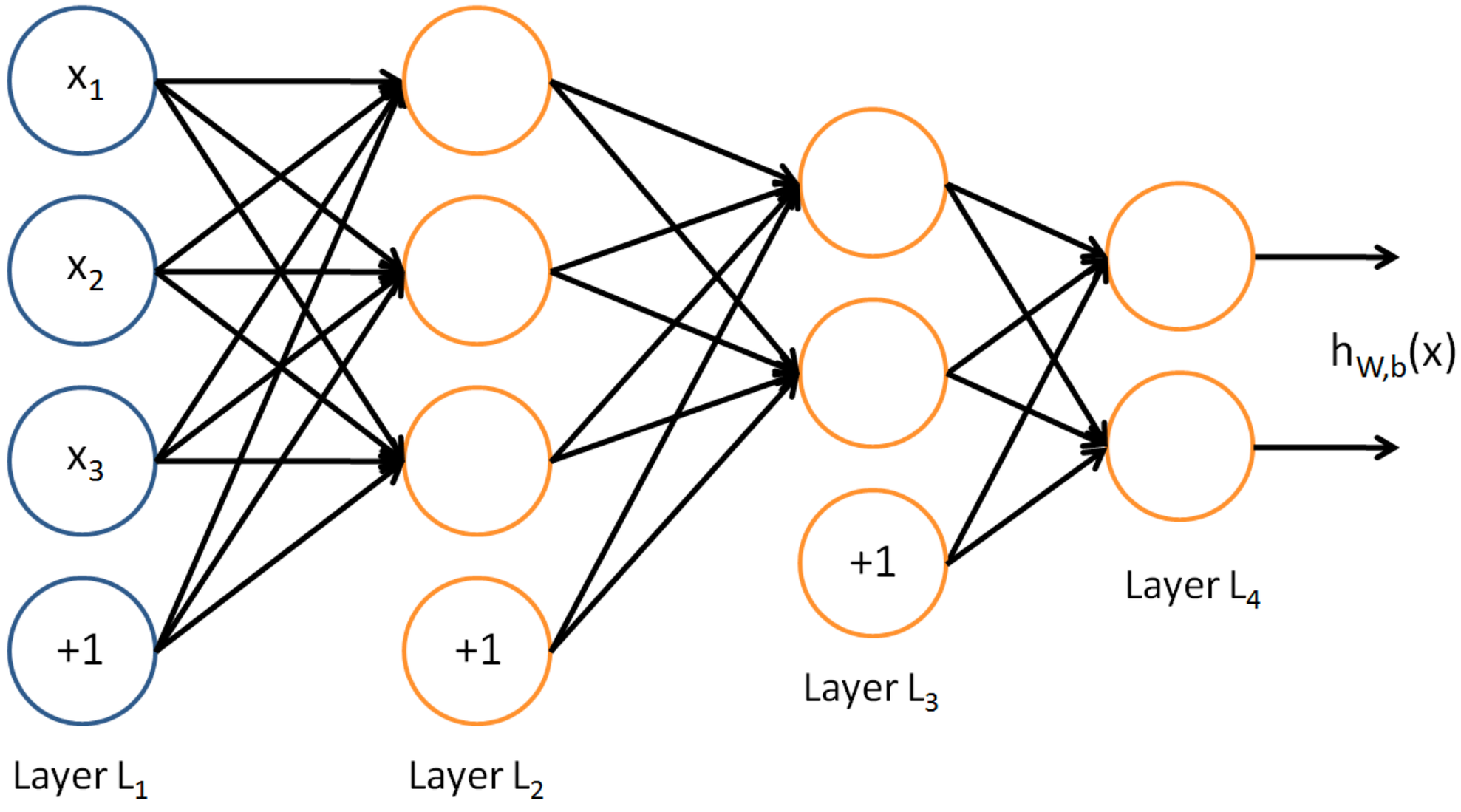
Autoencoder network and Restricted Blotzmann Machines

Transformer network

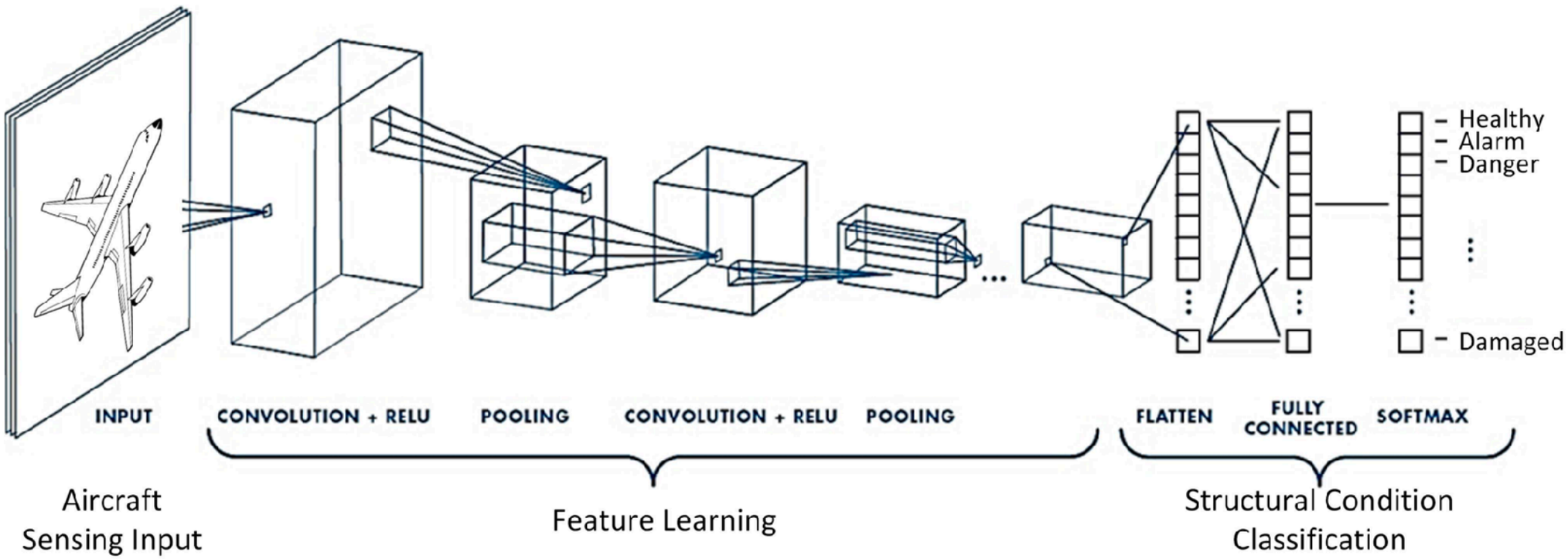
U-Net

Generative adversarial network

Feed Forward Network

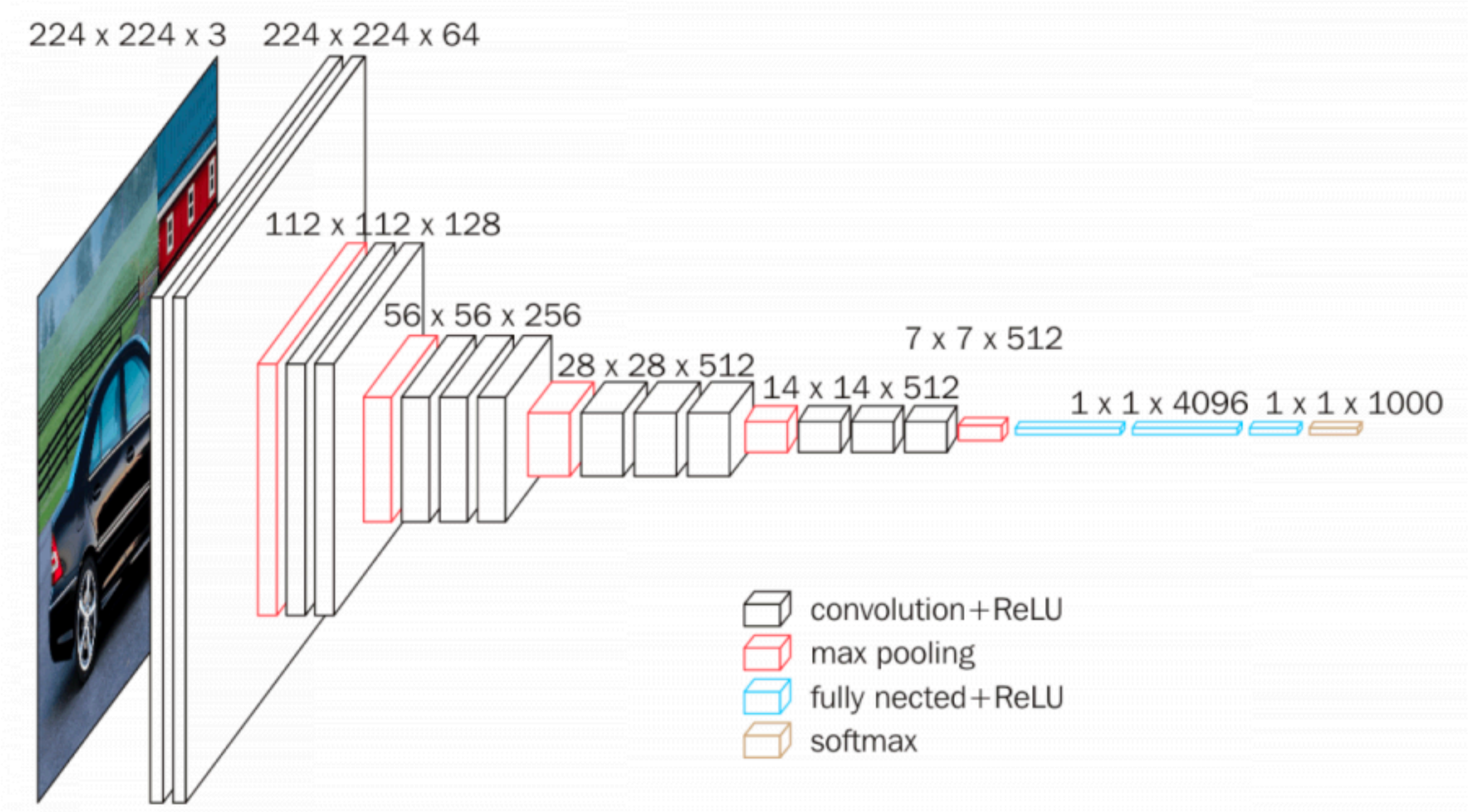


Convolutional Neural Network

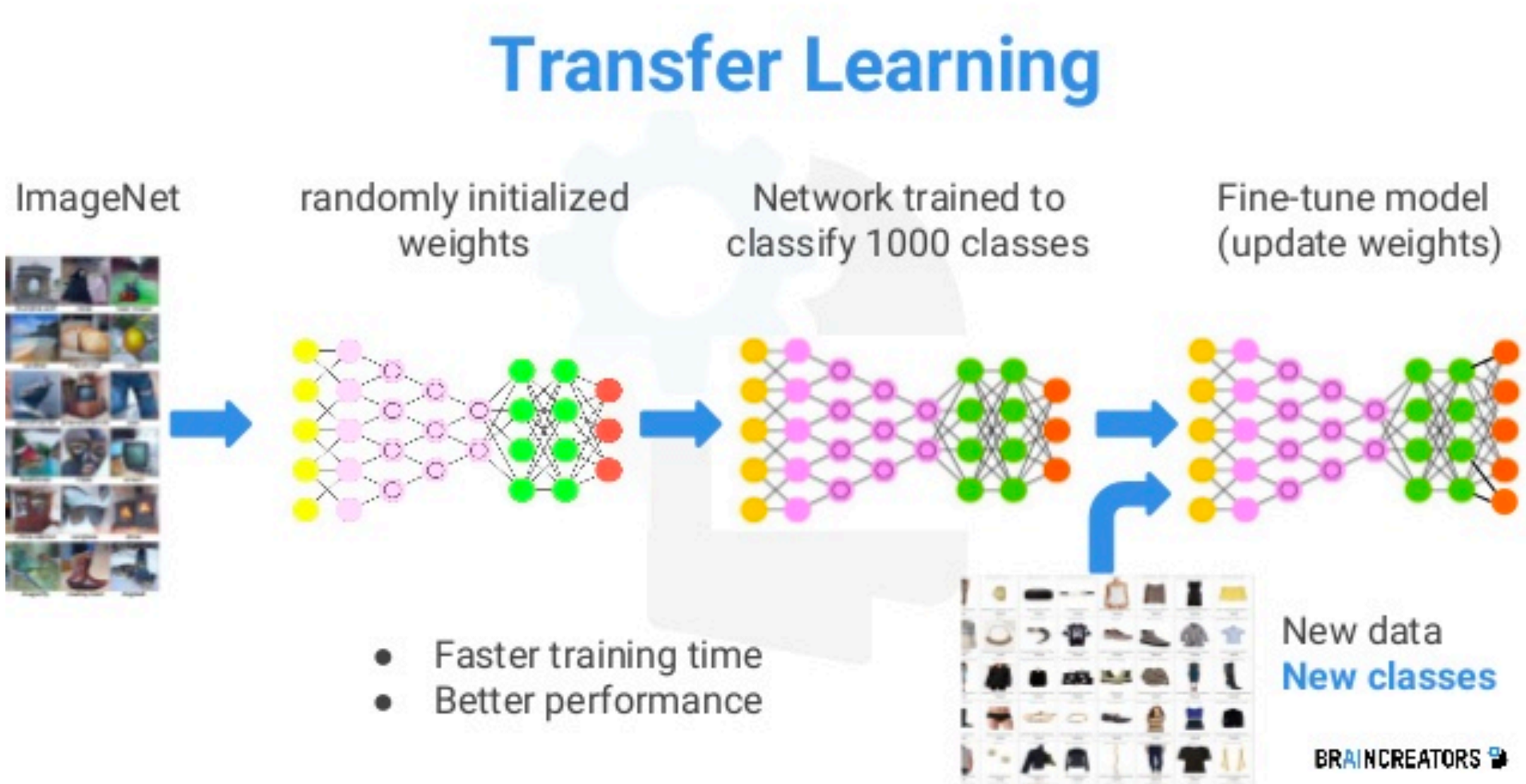


https://www.mdpi.com/sensors/sensors-19-04933/article_deploy/html/images/sensors-19-04933-g001.png

VGG 16


224 x 224 x 3
224 x 224 x 64
112 x 112 x 128
56 x 56 x 256
28 x 28 x 512
14 x 14 x 512
7 x 7 x 512
1 x 1 x 4096
1 x 1 x 1000
convolution+ReLU
max pooling
fully nected+ReLU
softmax

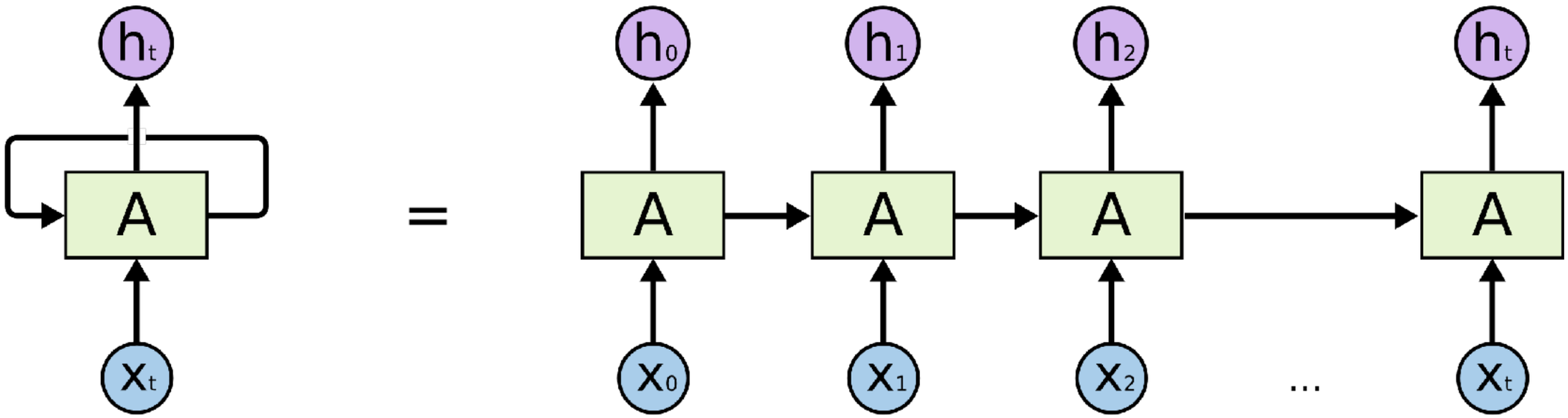
Finetuning


Transfer Learning
ImageNet
randomly initialized weights
Network trained to classify 1000 classes
Fine-tune model (update weights)
Faster training time
Better performance
New data
New classes
BRAINCREATORS

Transfer learning example

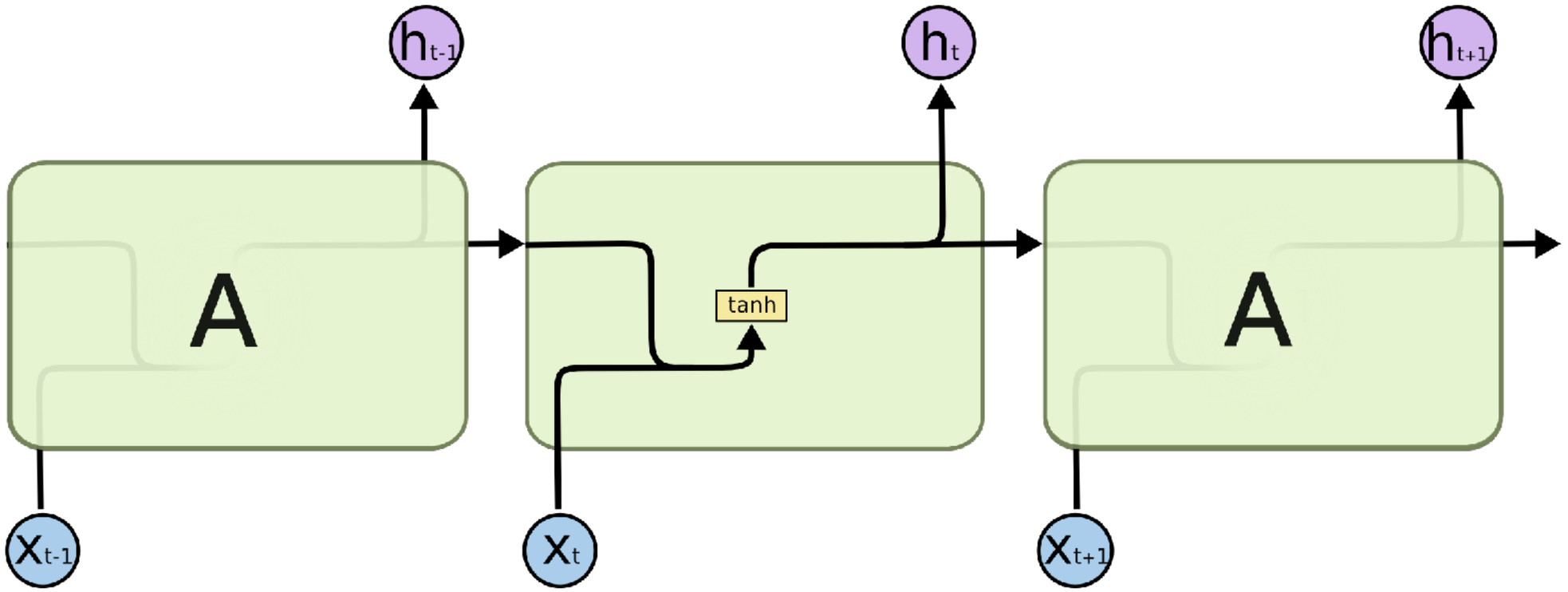
10-Transfer_learning.ipynb

Recurrent Neural Network



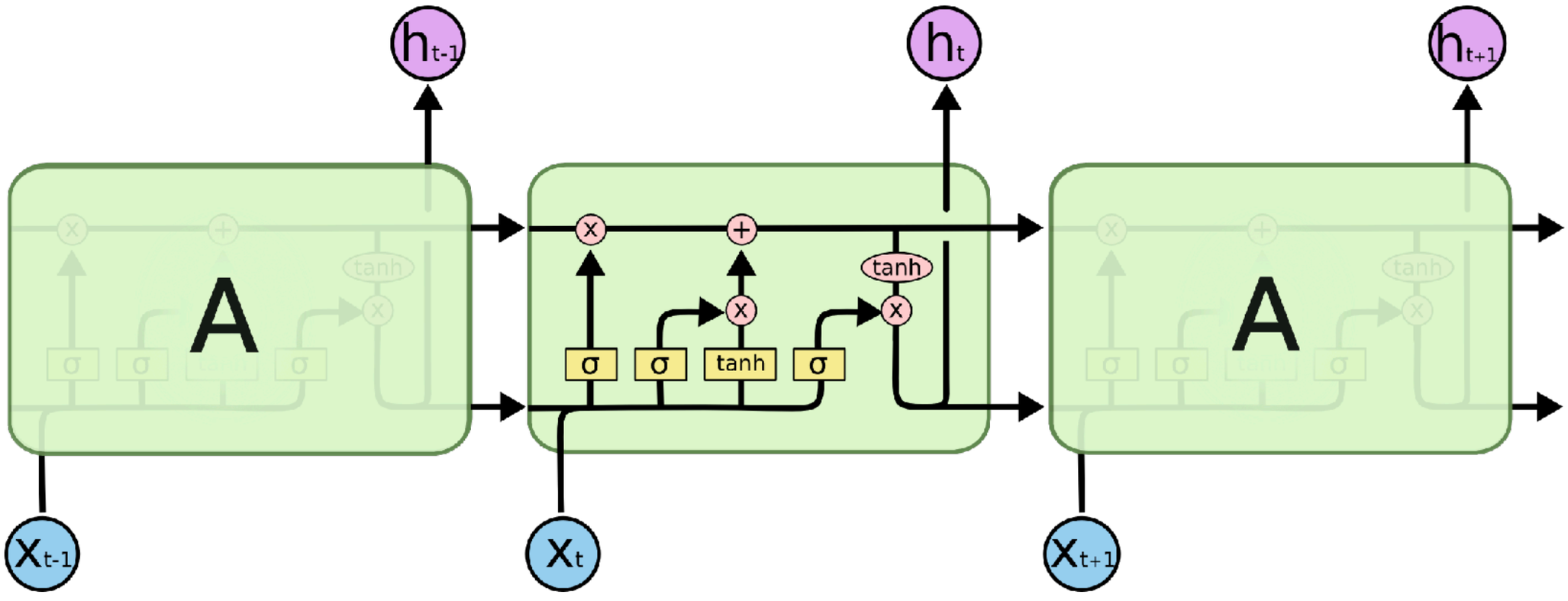
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent Neural Network



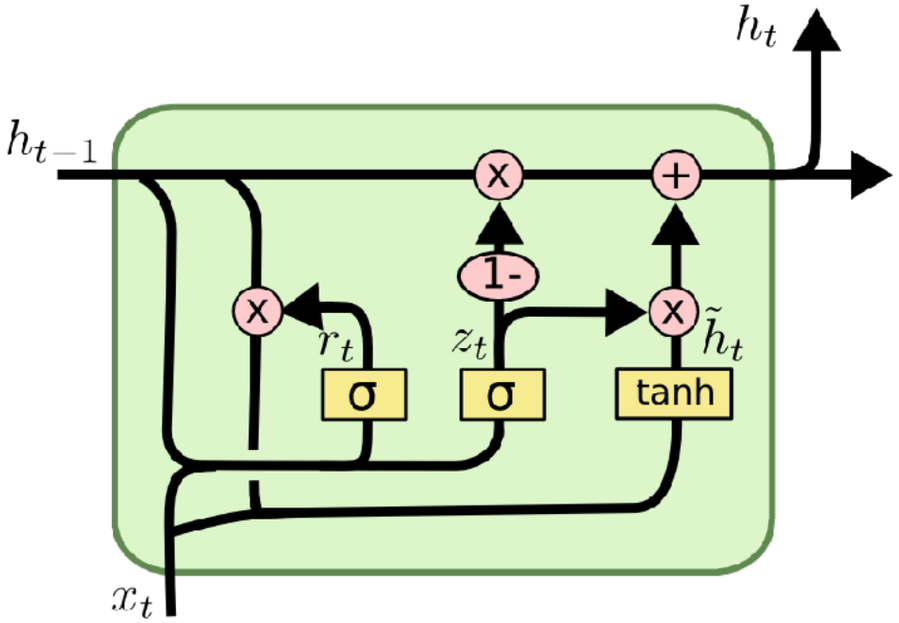
https://www.tensorflow.org

Long Short Term Memory



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Unit



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

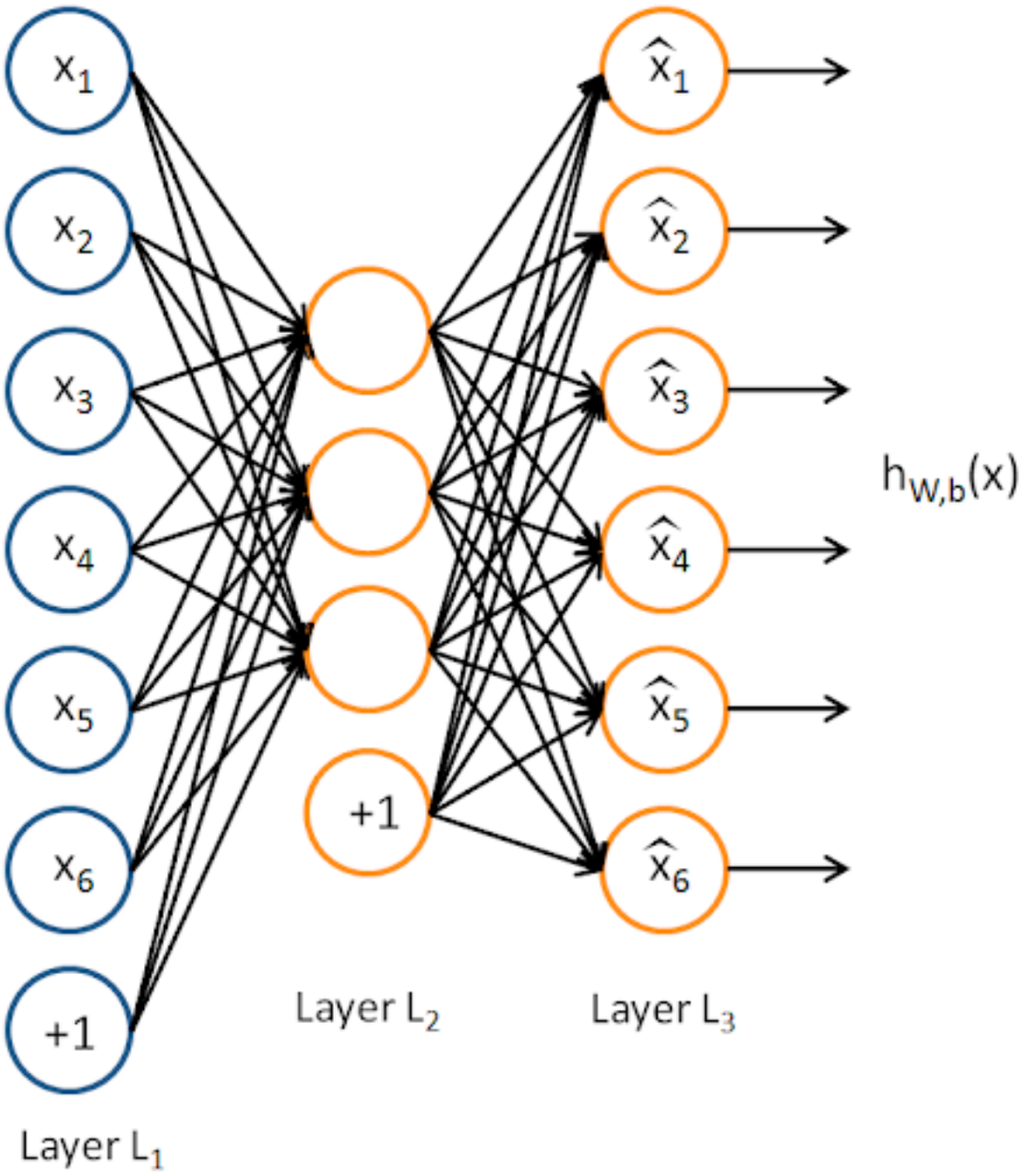
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Regression using RNN

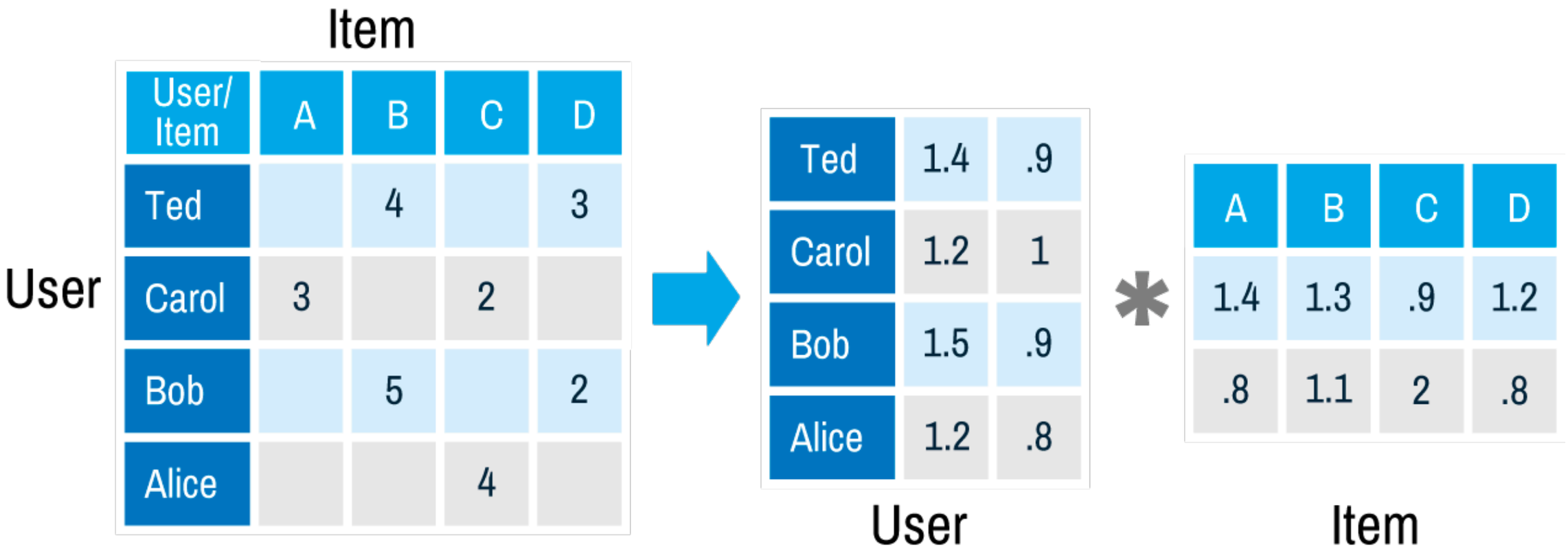
11-Regression-RNN-assignment.ipynb

Autoencoder Network



http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/

Collaborative filtering

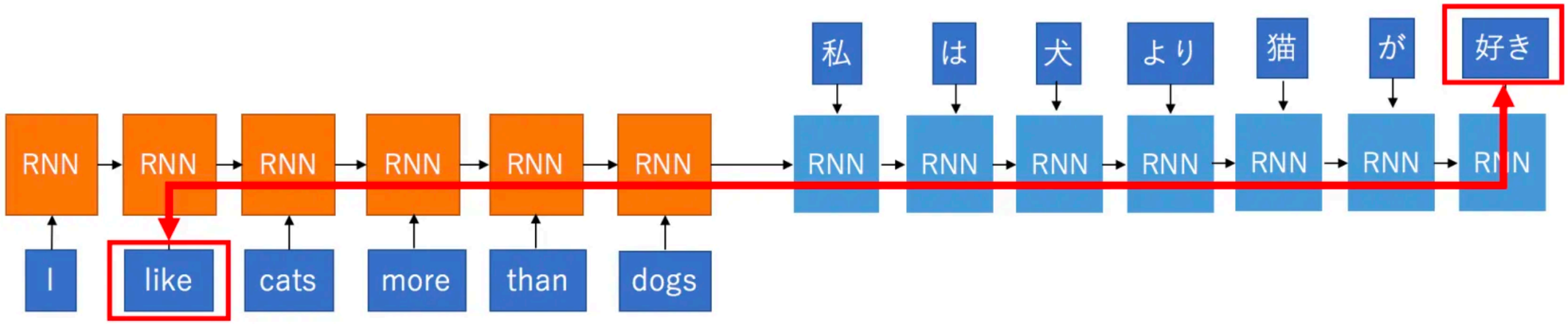


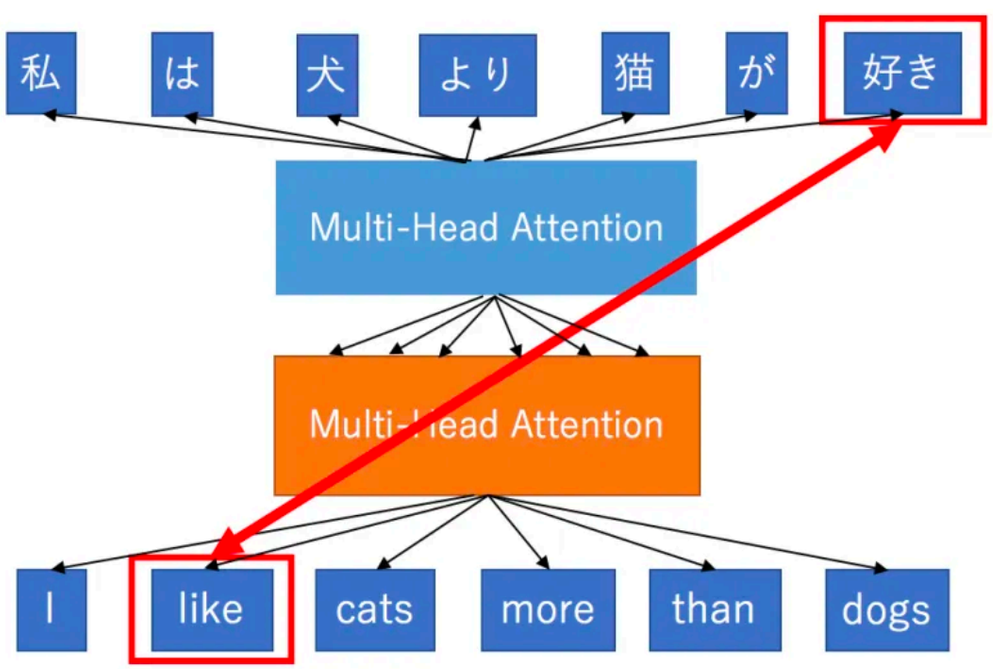
https://mapr.com/ebooks/spark

Embedding-based recommendation

12-Recommendation-assignment.ipynb

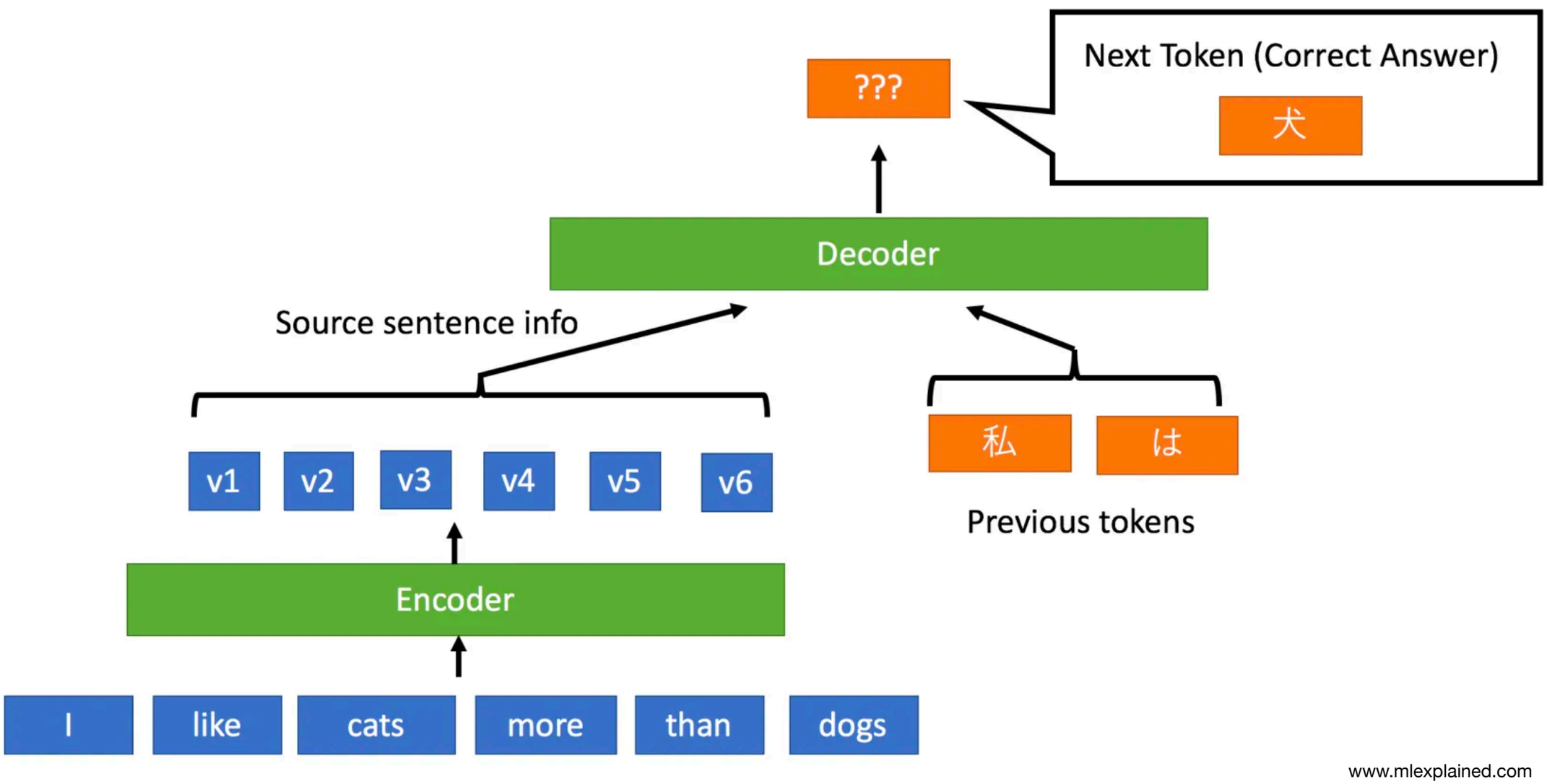
Transformer





source: www.mlexplained.com

Translation with Transformers



GPT-2 Language model

Donald Trump told...

GPT-2 Language model

Donald Trump told the Times he is preparing a "major speech" on his economic plans, but did not provide details on what it will entail.

“I’m getting ready for the speech. And I will have a major speech on Tuesday.” Trump said during an interview in the White House residence.

GPT-3: **https://openai.com/blog/openai-api/**

BERT
(classification)

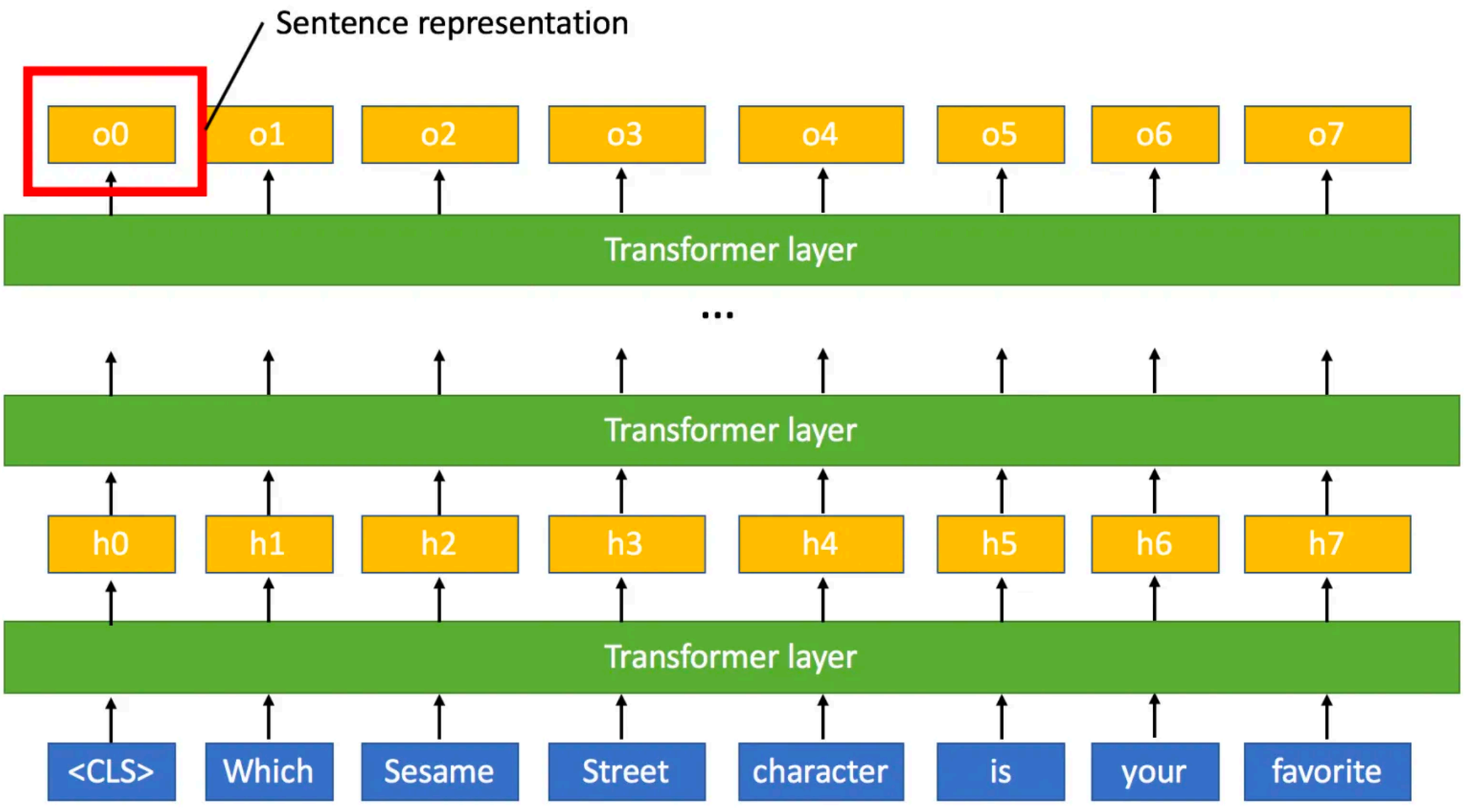
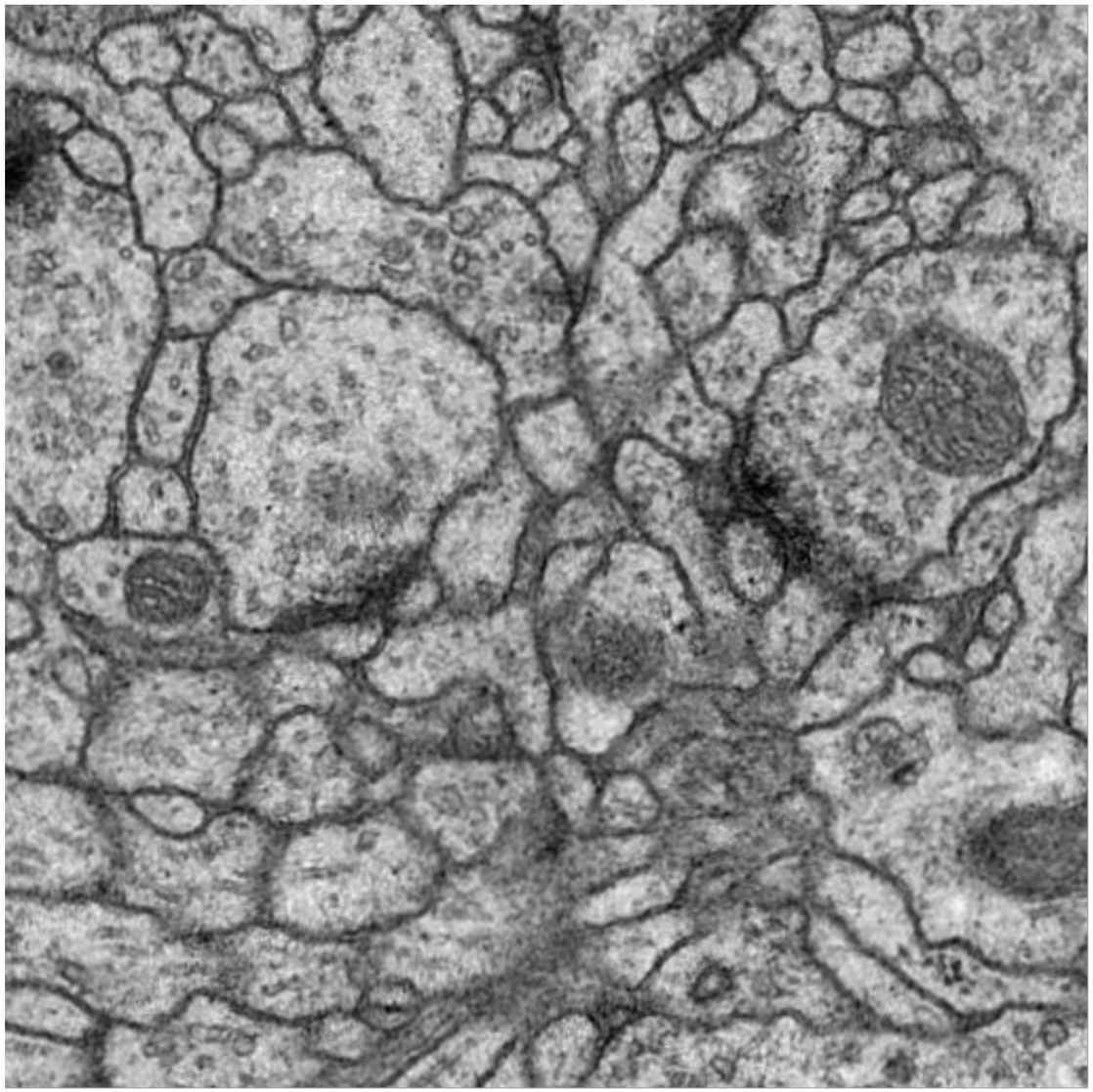
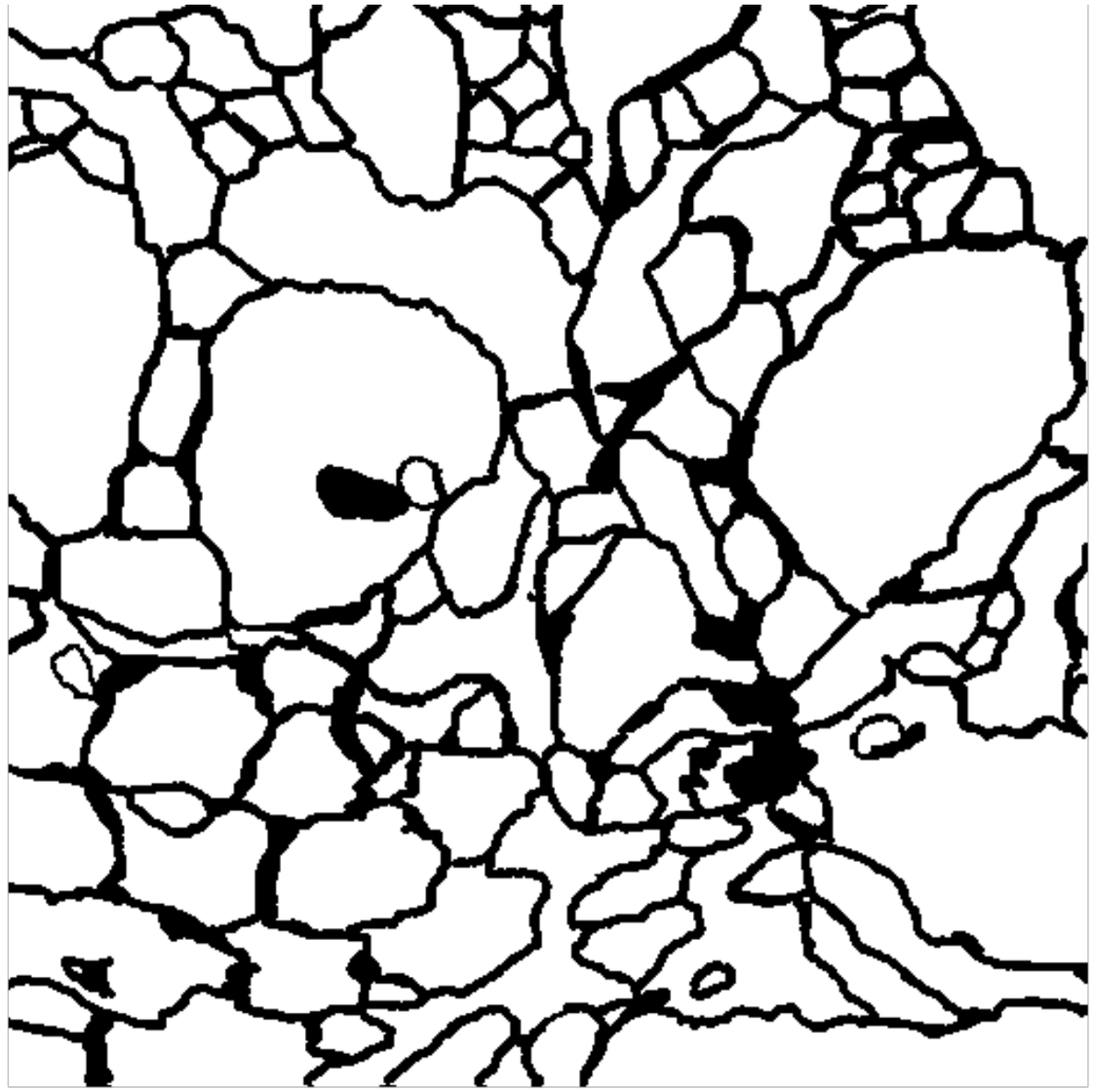


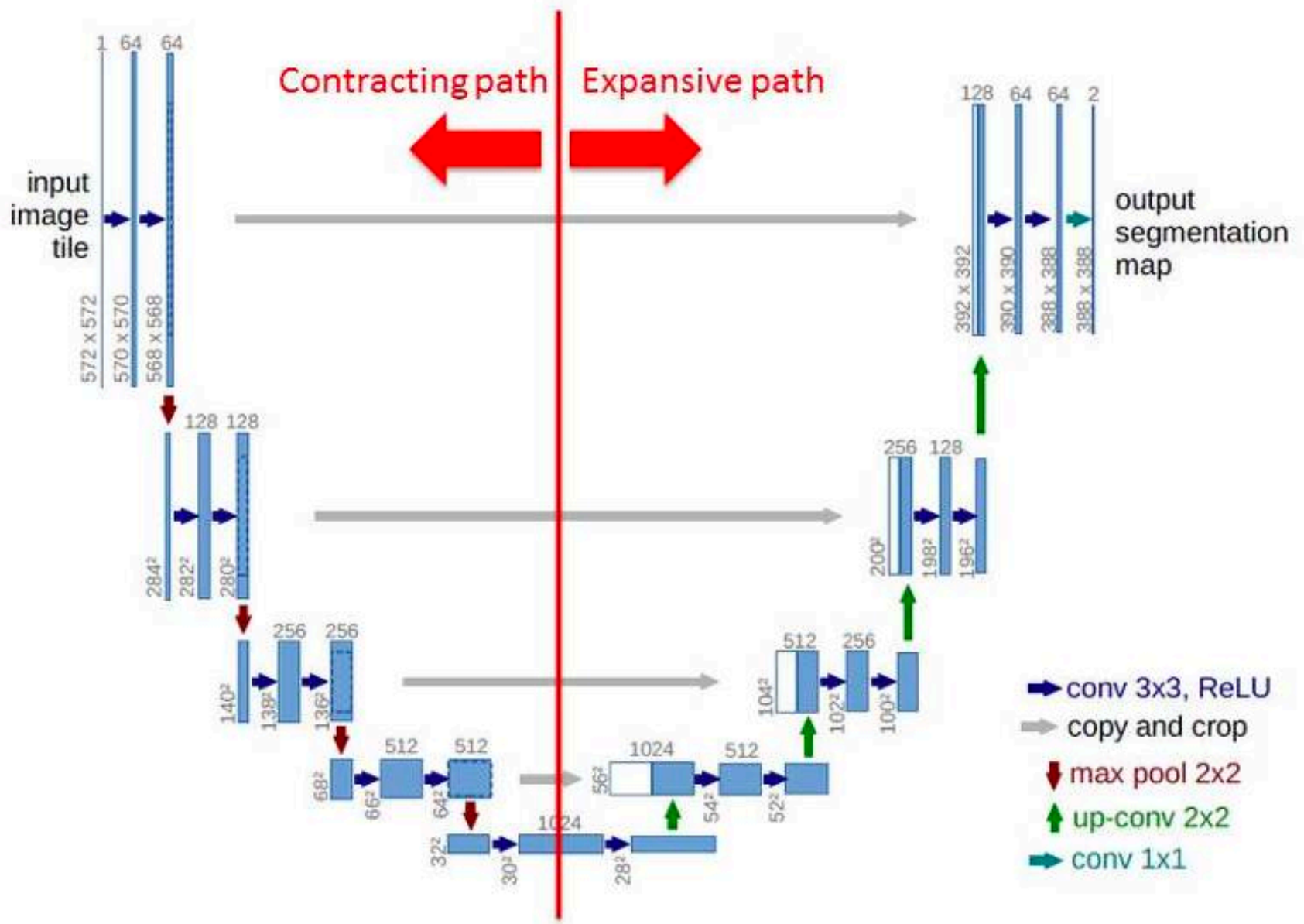
Image segmentation





U-Net

Network Architecture

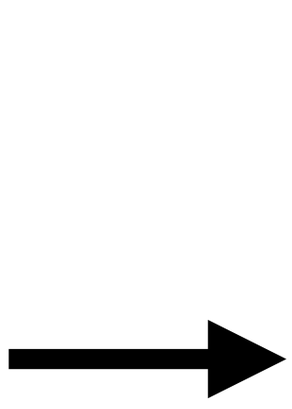


U-Net segmentation example

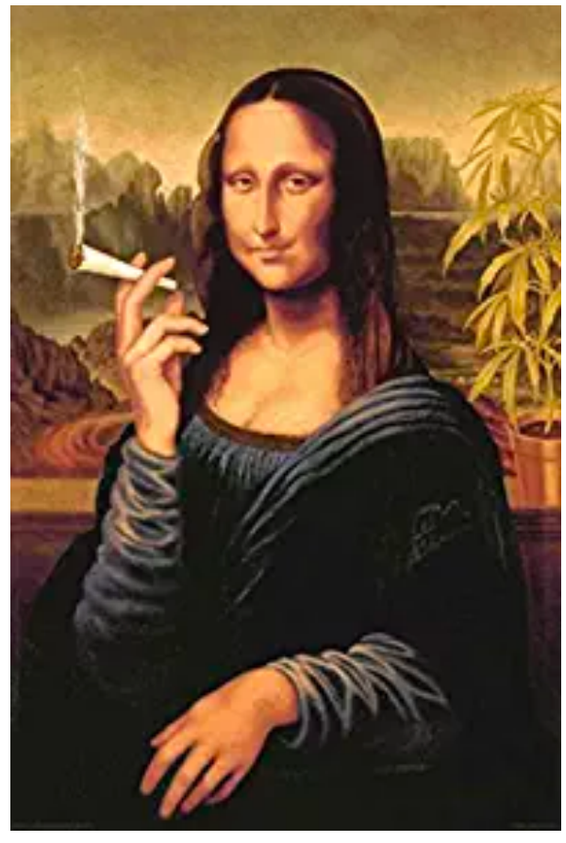
13-Segmentation.ipynb

Generative models with neural networks

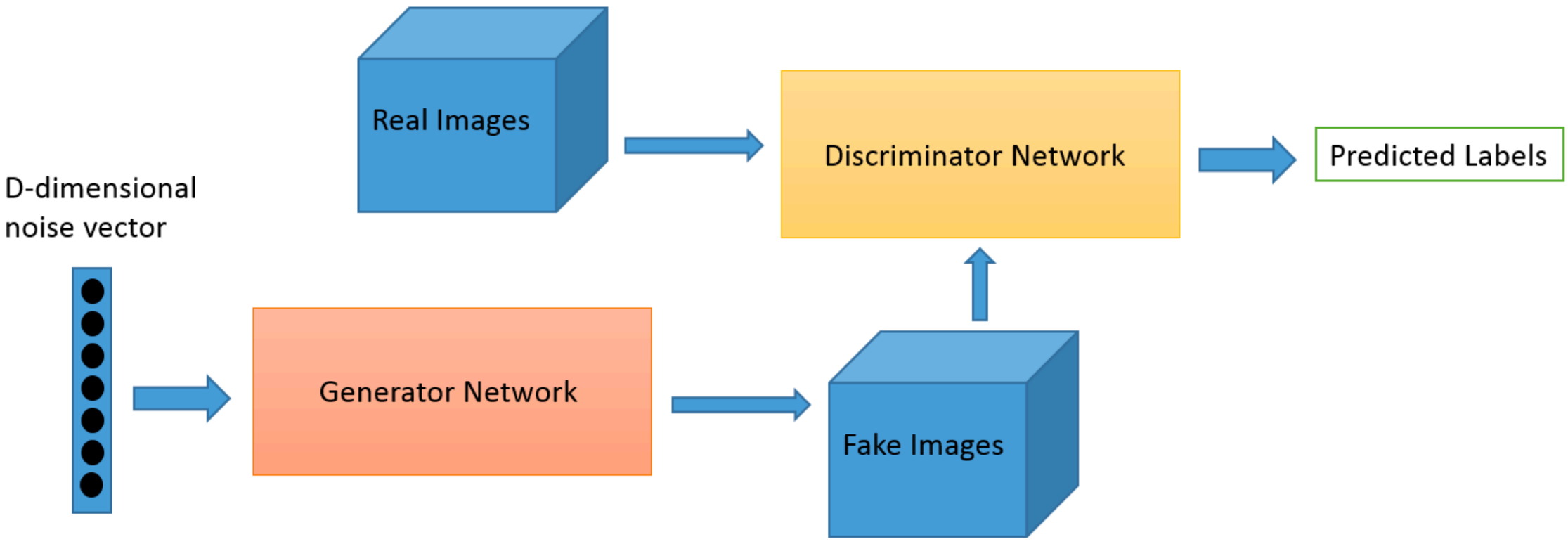
Random noice



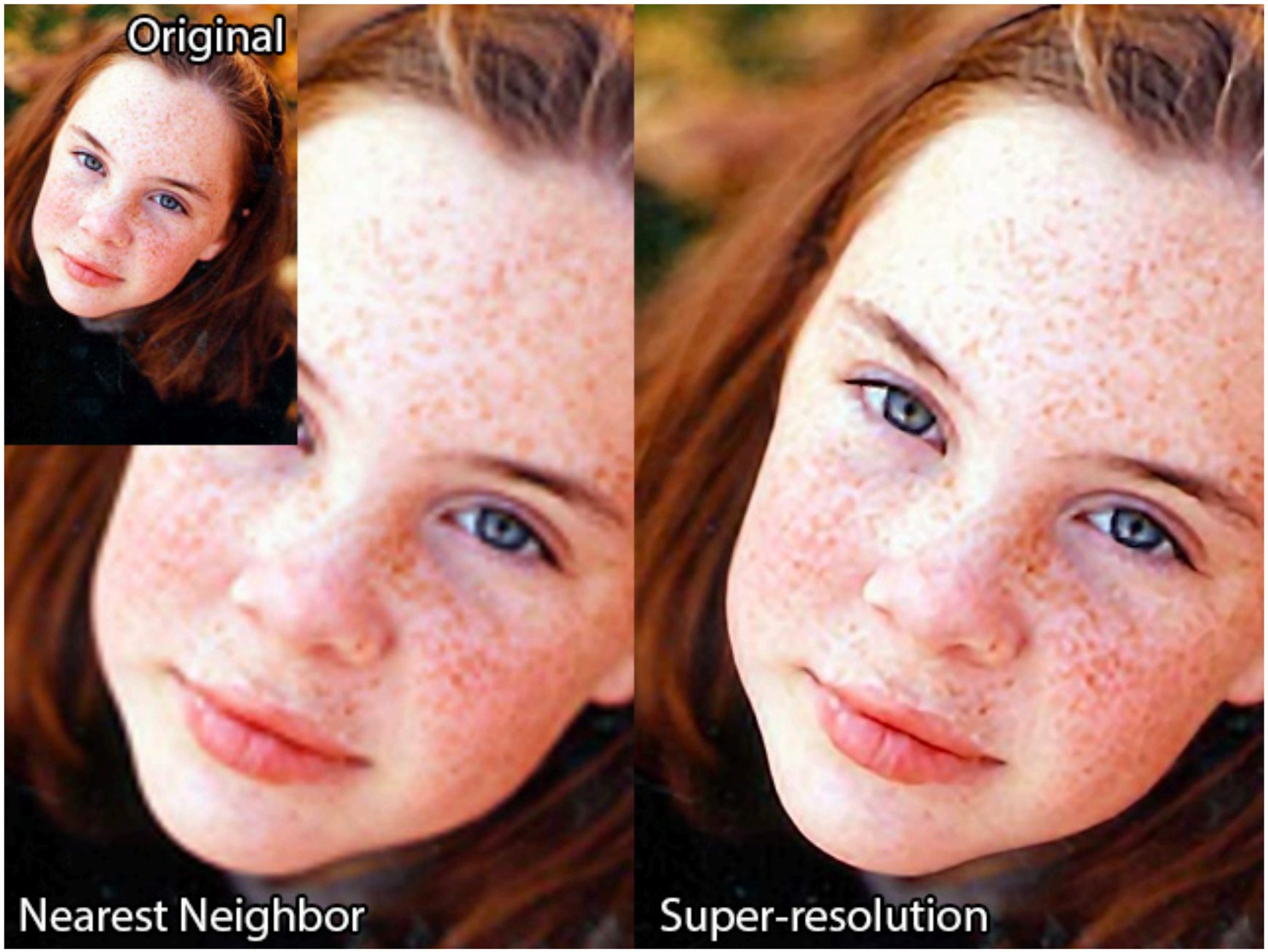
Artificial Neural network



Generative Adversarial Networks



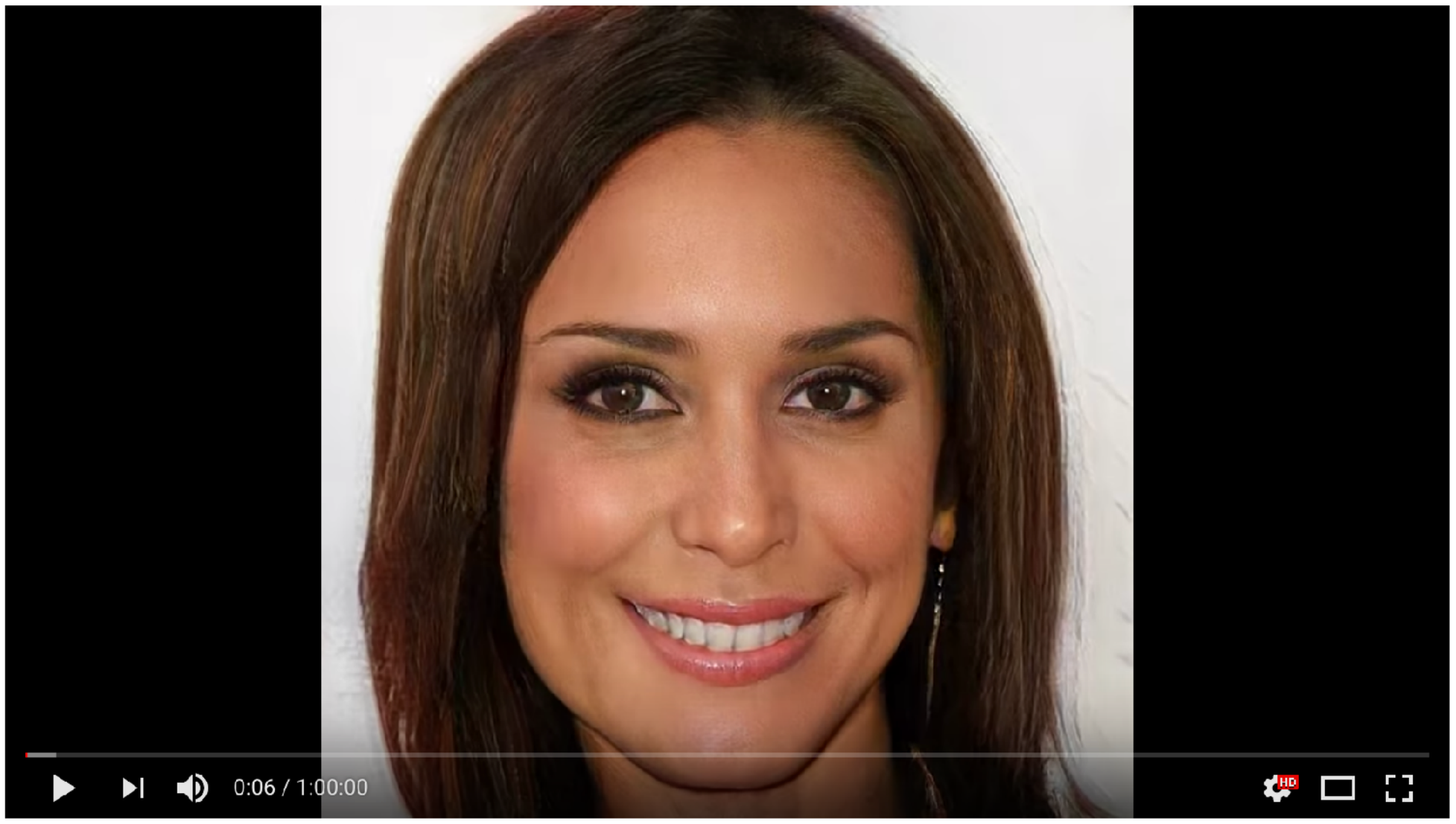
Superresolution



Generative Adversarial Networks

14_GANs.ipynb

Image synthesis



One hour of imaginary celebrities

95 832 zhlédnutí

TO SE MI LÍBÍ NELÍBÍ SE SDÍLET

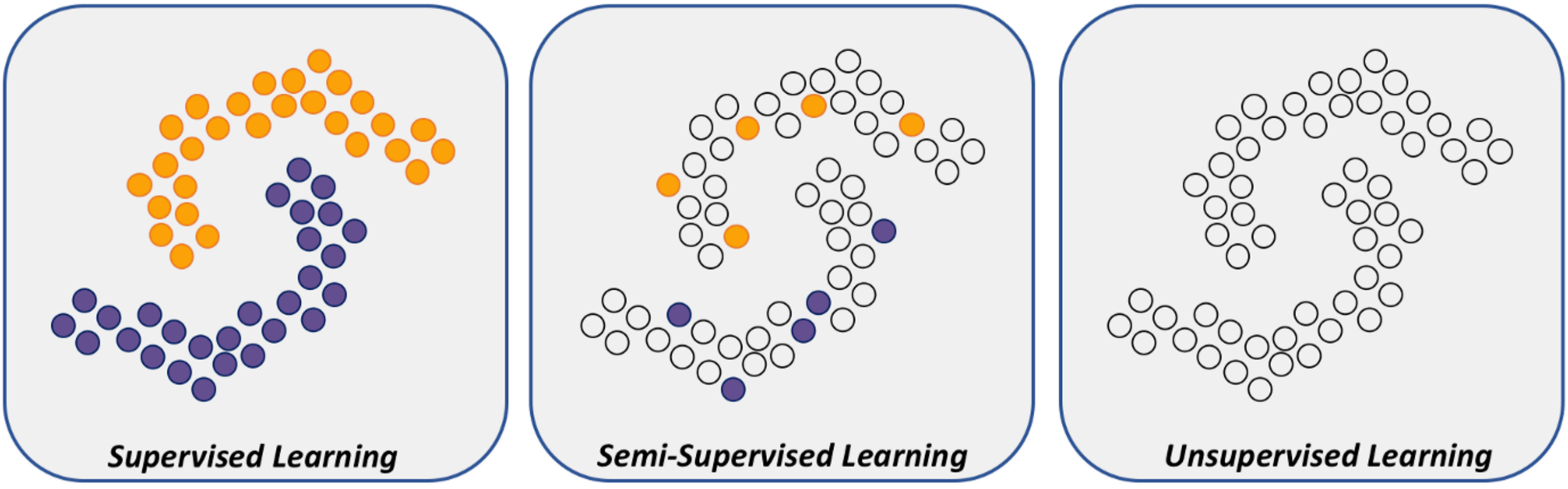
Which one is fake?

Outline

Day 4

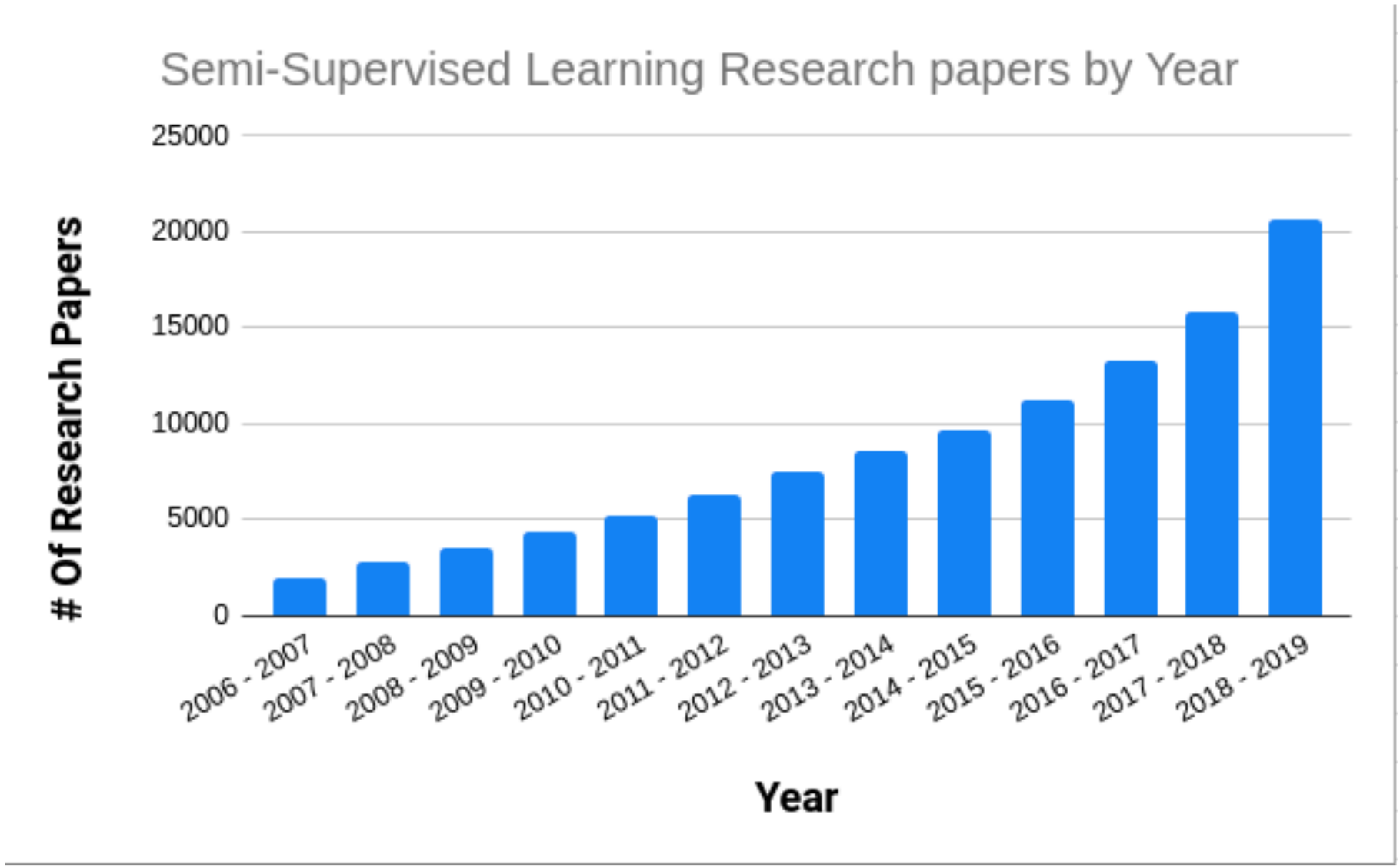
- Semi-supervised learning
- AutoML approaches
- Confidence estimation
- ML Explainability

Semi-supervised learning



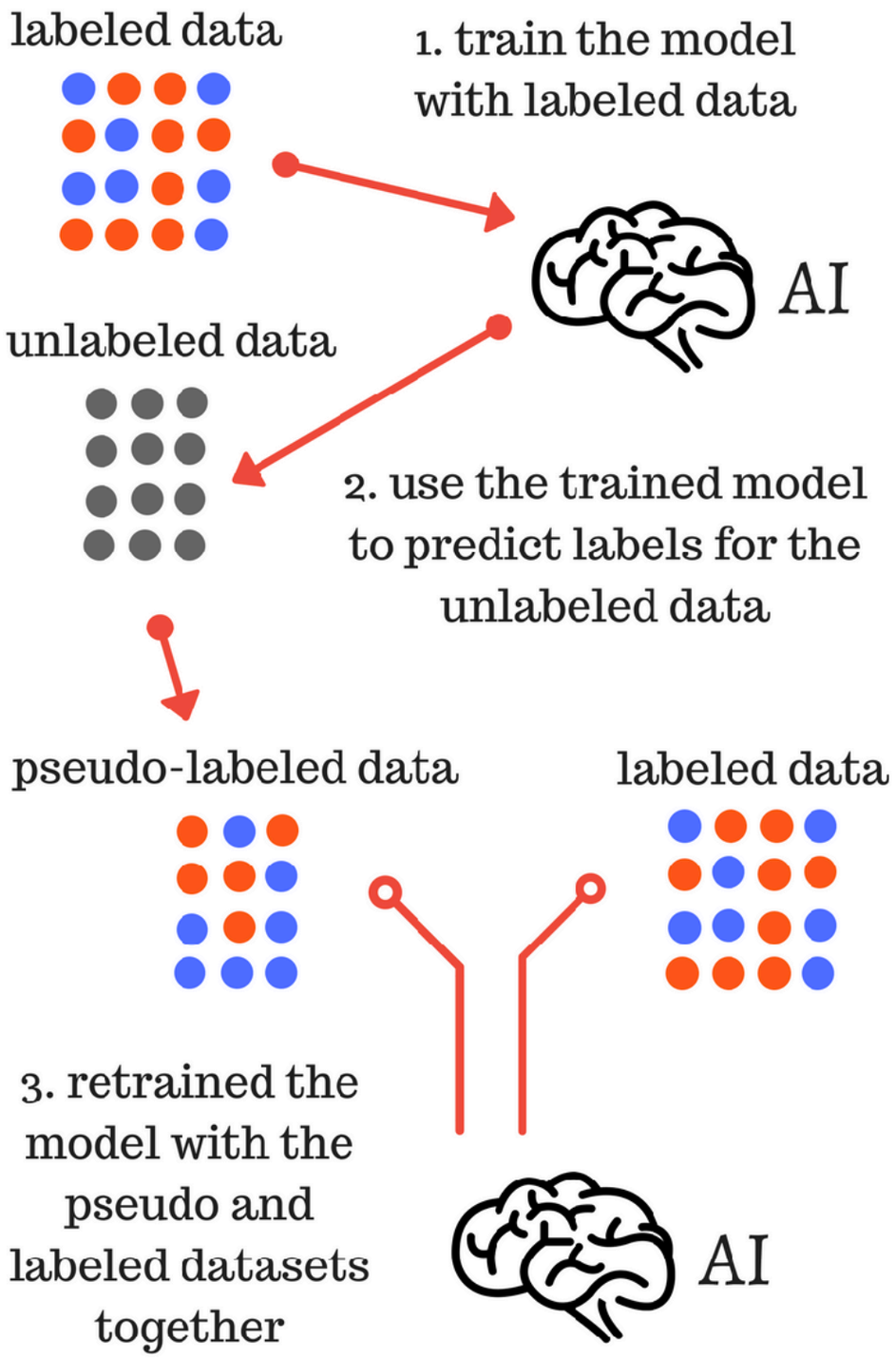
Source: KDnuggets.

Small Data is the new Big Data



https://towardsdatascience.com/pseudo-labeling-to-deal-with-small-datasets-what-why-how-fd6f903213af

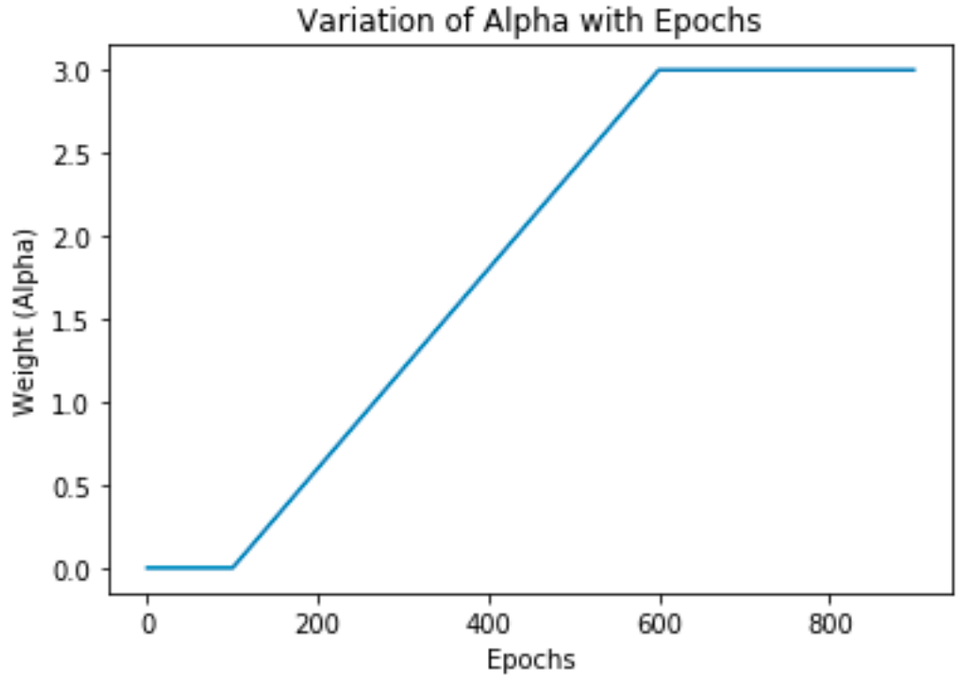
Pseudo-labeling



https://datawhatnow.com/pseudo-labeling-semi-supervised-learning/

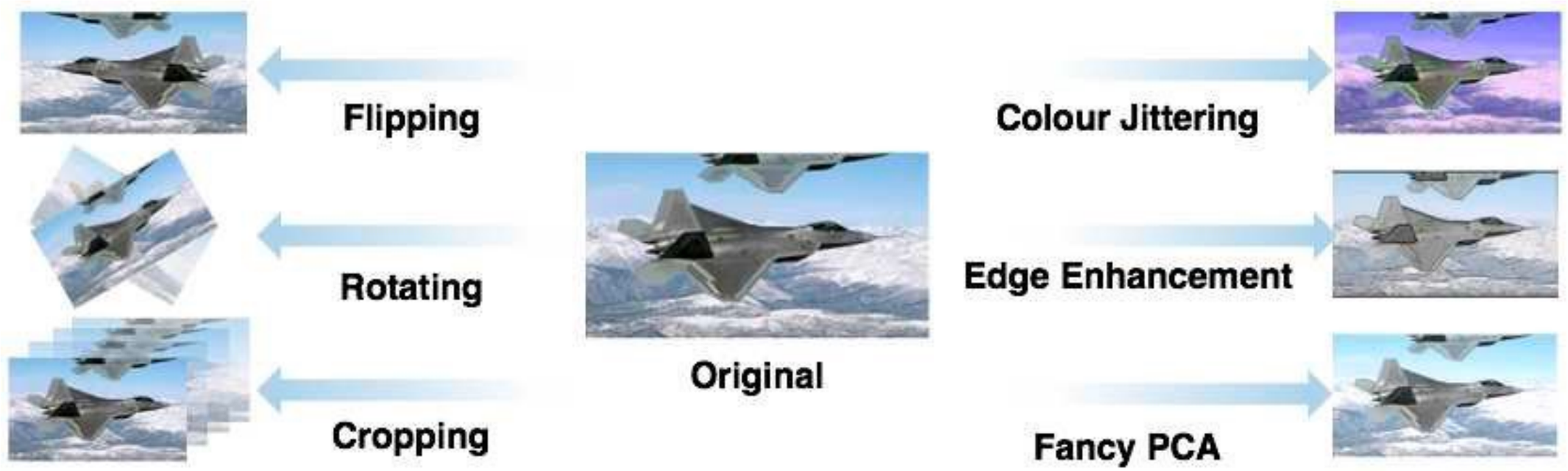
Pseudo-labeling with neural networks

- Train model on a batch of labeled data
- Use the trained model to predict labels on a batch of unlabeled data
- Use the predicted labels to calculate the loss on unlabeled data
- Combine labeled loss with α-weighted unlabeled loss and backpropagate
- Repeat

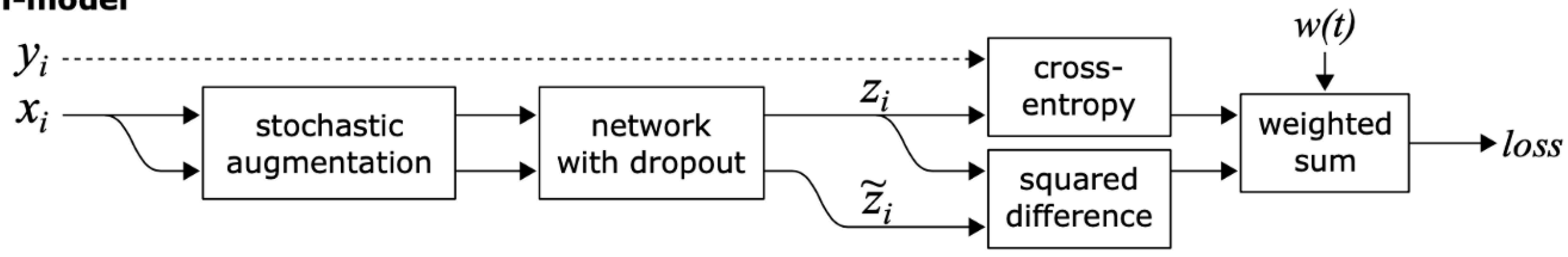


https://towardsdatascience.com/pseudo-labeling-to-deal-with-small-datasets-what-why-how-fd6f903213af

PI-model

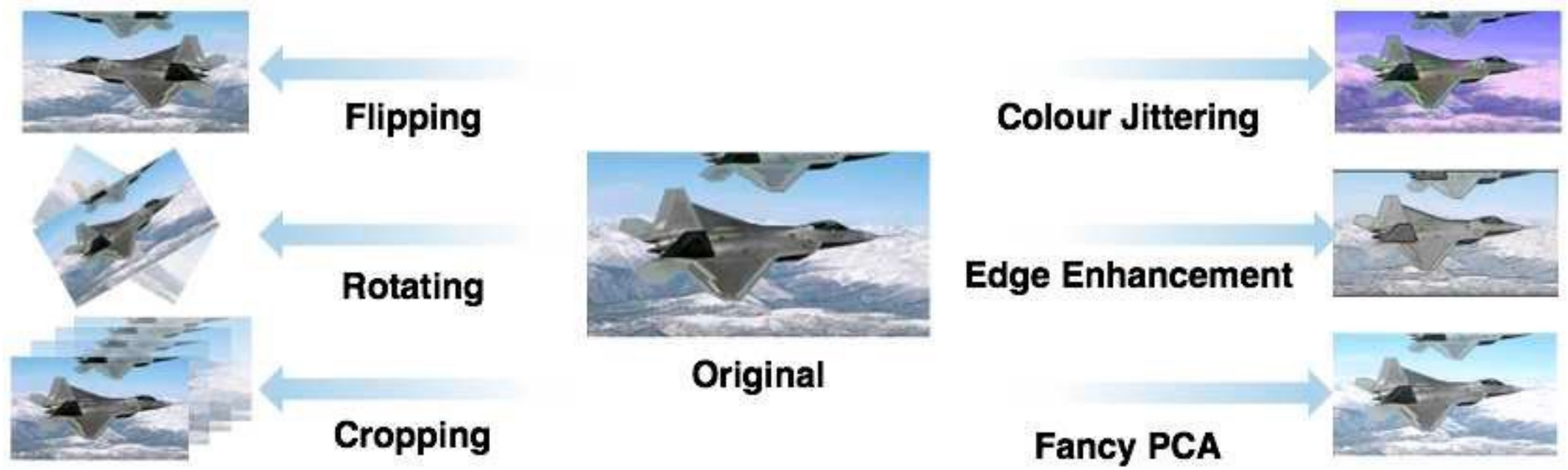


Π-model

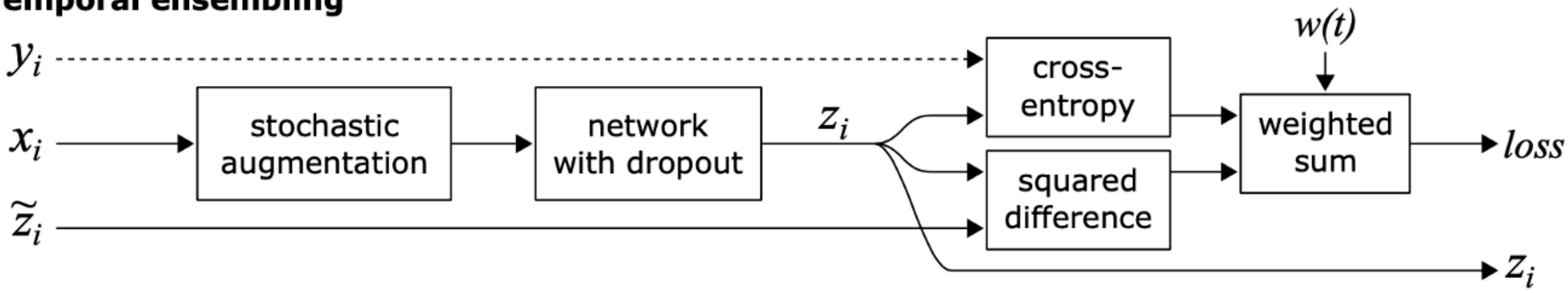


https://medium.com/datadriveninvestor/training-a-machine-learning-model-from-a-few-examples-few-shot-learning-part-1-50402ab8dfa5

Temporal ensembling

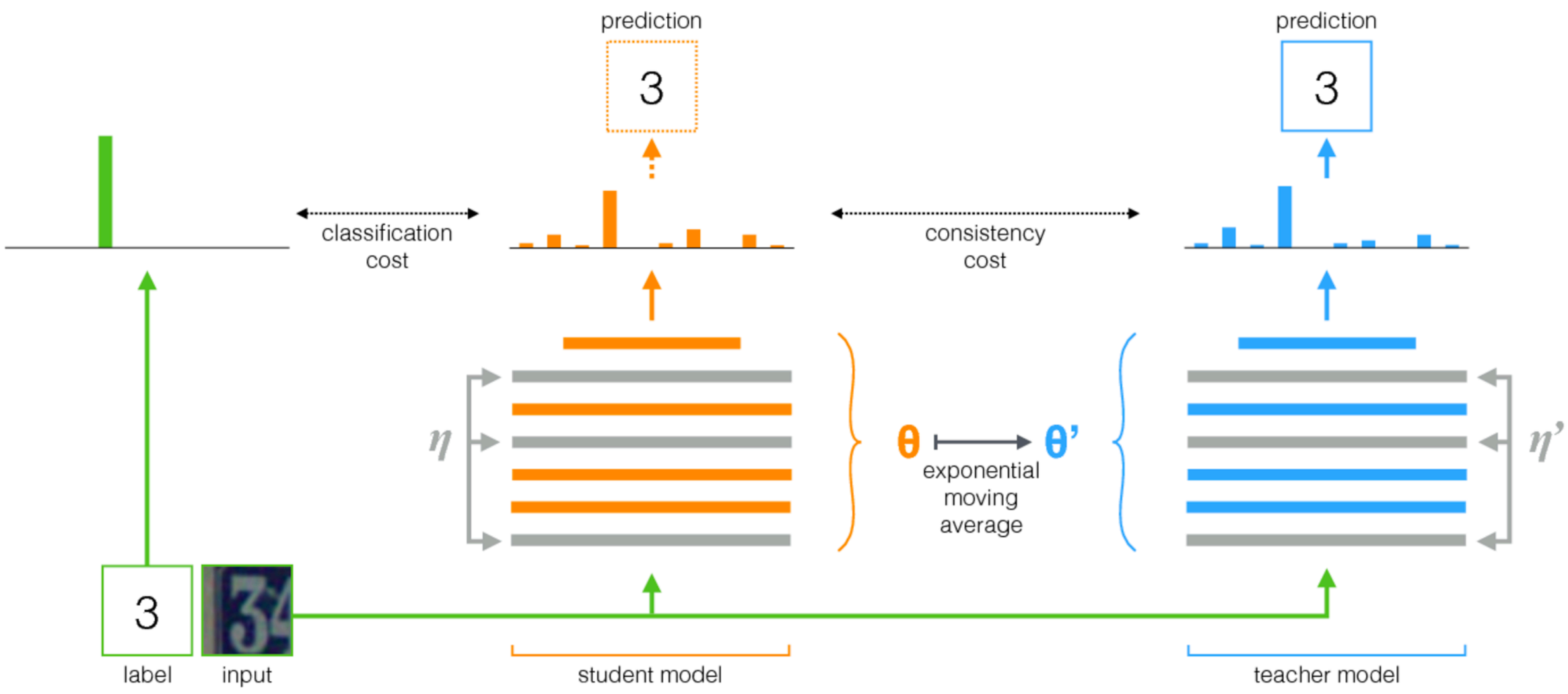


Temporal ensembling



https://medium.com/datadriveninvestor/training-a-machine-learning-model-from-a-few-examples-few-shot-learning-part-1-50402ab8dfa5

Mean-Teacher

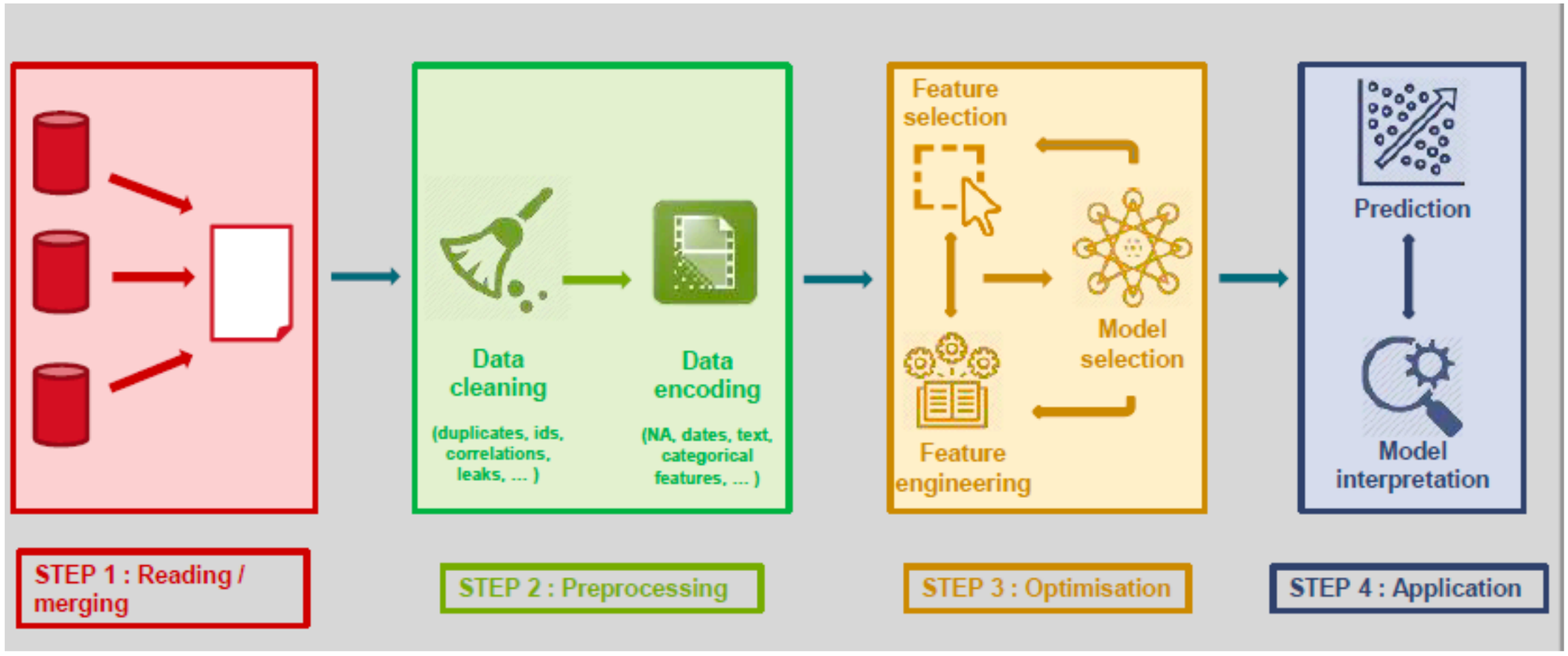


https://github.com/CuriousAI/mean-teacher

Clustering as semi-supervised learning

15-clustering-assignment.ipynb

AutoML approaches



https://www.slideshare.net/AxeldeRomblay?utm_campaign=profiletracking&utm_medium=sssite&utm_source=ssslideview

AutoML frameworks

H2O AutoML

Google's AutoML

AutoGluon

bigml

Microsoft Azure

HYPEROPT

MLBox,
Machine Learning Box

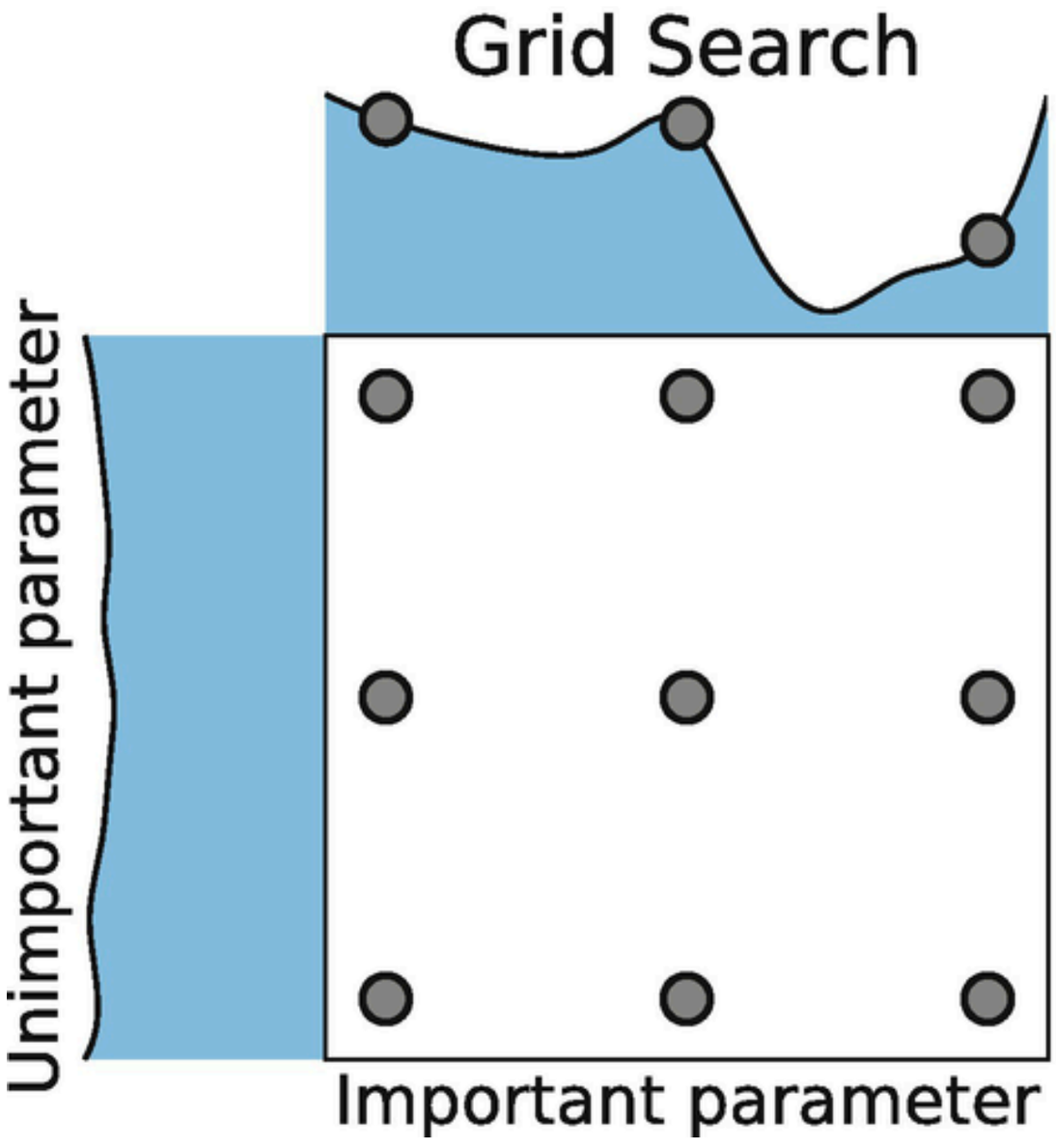
AutoKeras

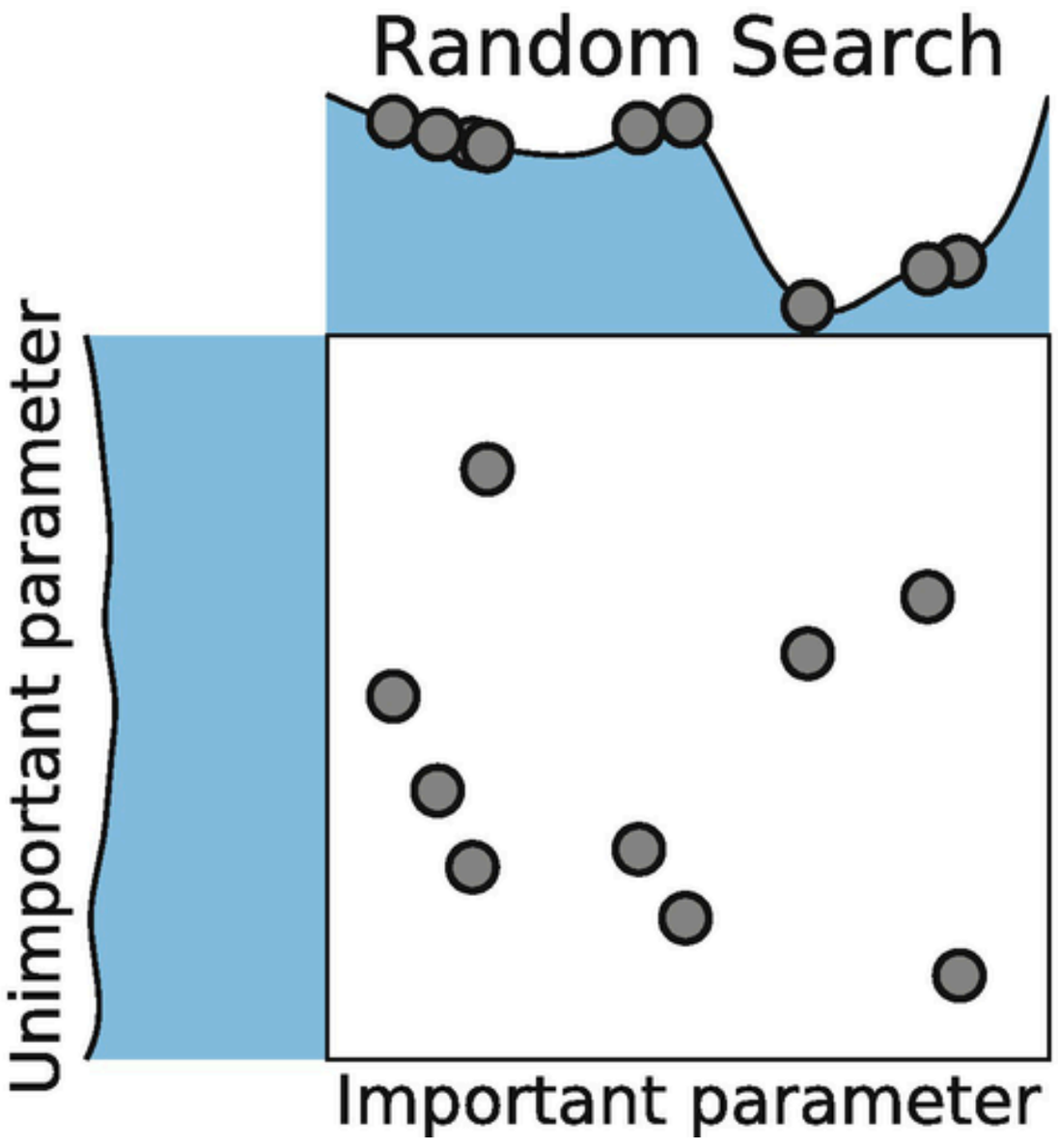
Hyper-parameters tuning

Parameters - learnable weights of the neural network

Hyper-parameters - parameters set before training (learning rate, momentum, dropout rate, etc.)

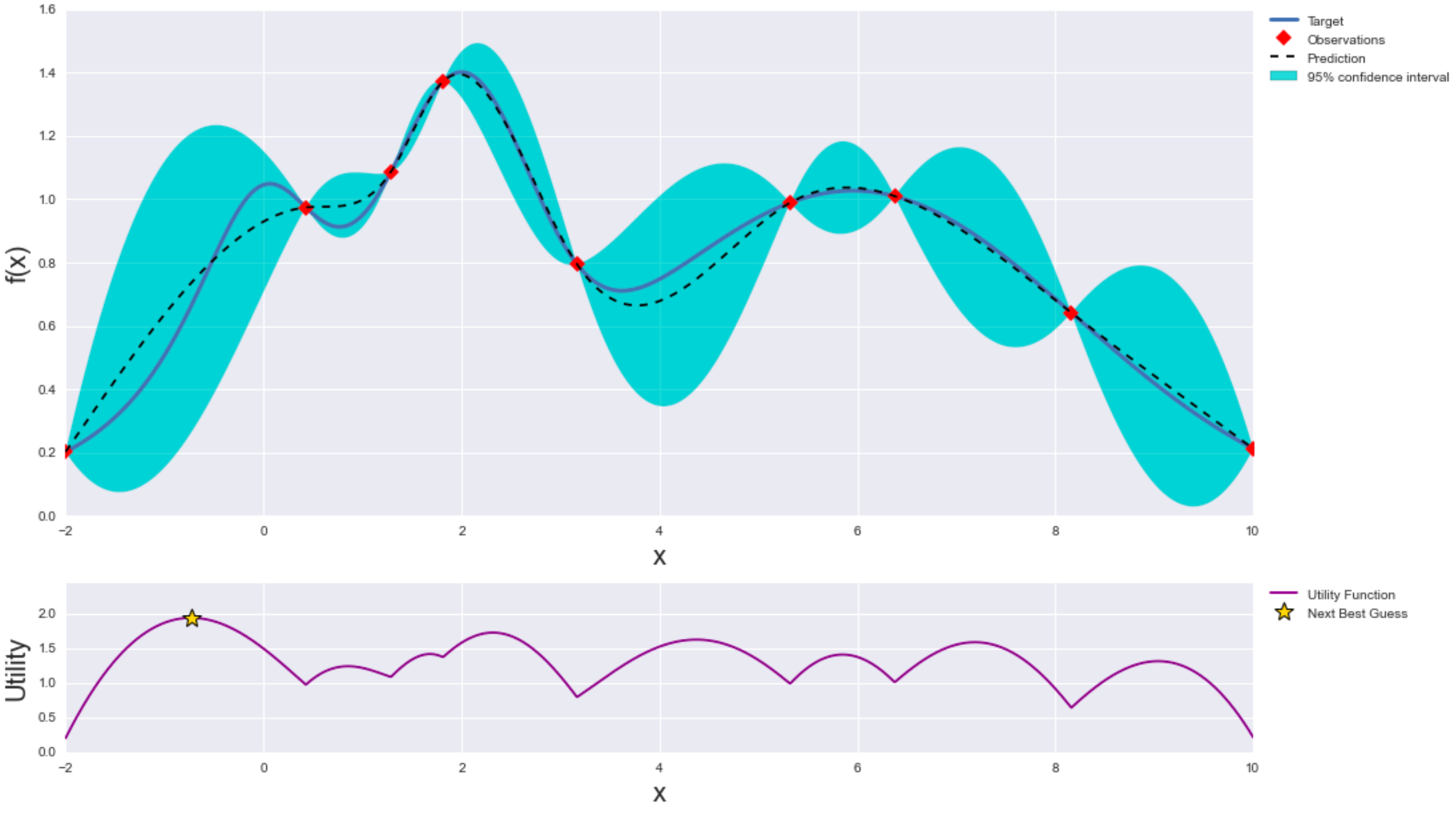
Grid search vs random search





https://medium.com/@cjl2fv/an-intro-to-hyper-parameter-optimization-using-grid-search-and-random-search-d73b9834ca0a

Bayesian optimization using Gaussian processes



https://medium.com/@hiepnguyen034/improving-neural-networks-performance-with-bayesian-optimization-efbaa801ad26

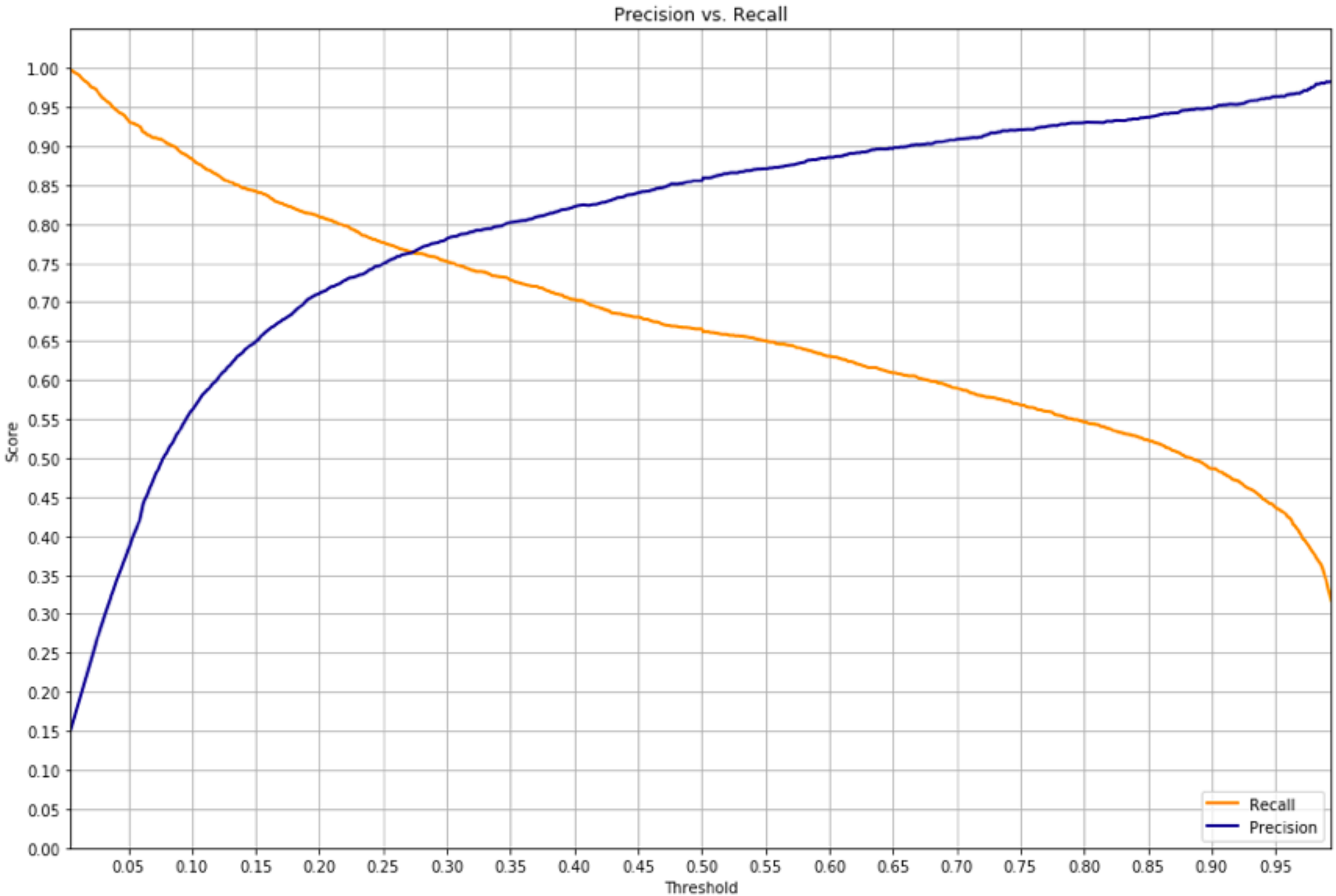
Neural Architecture Search (NAS)

https://towardsdatascience.com/neural-architecture-search-nas-the-future-of-deep-learning-c99356351136

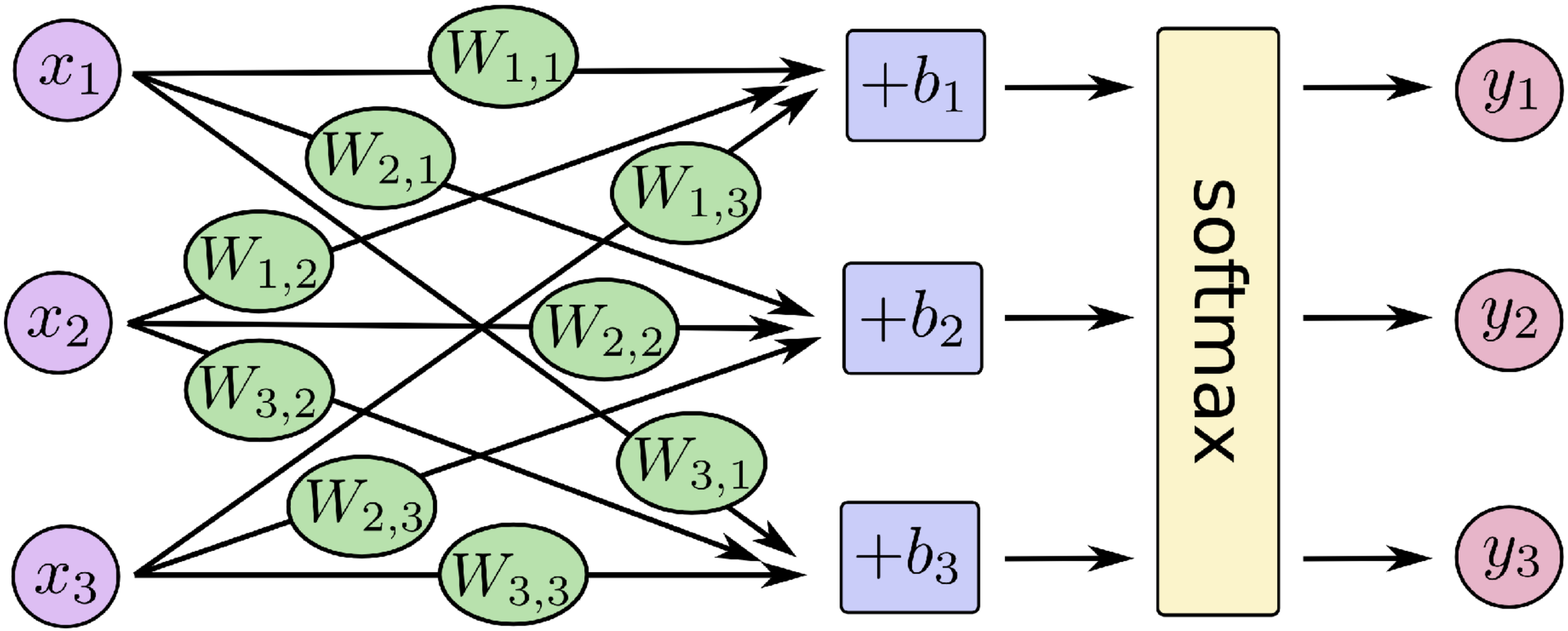
AutoML with AutoKeras

16-Autokeras.ipynb

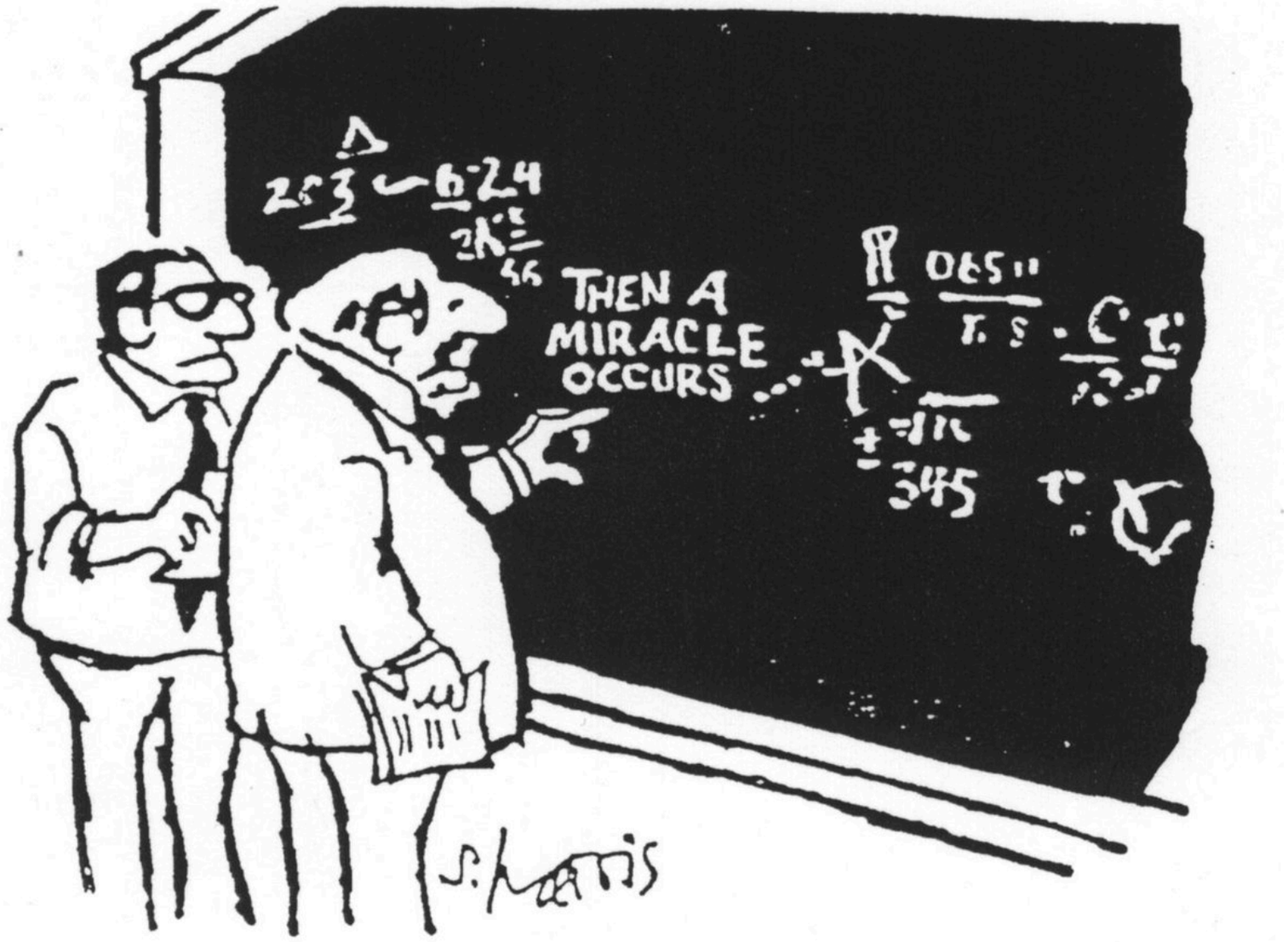
Confidence estimation



Confidence estimation



Neural network explainability

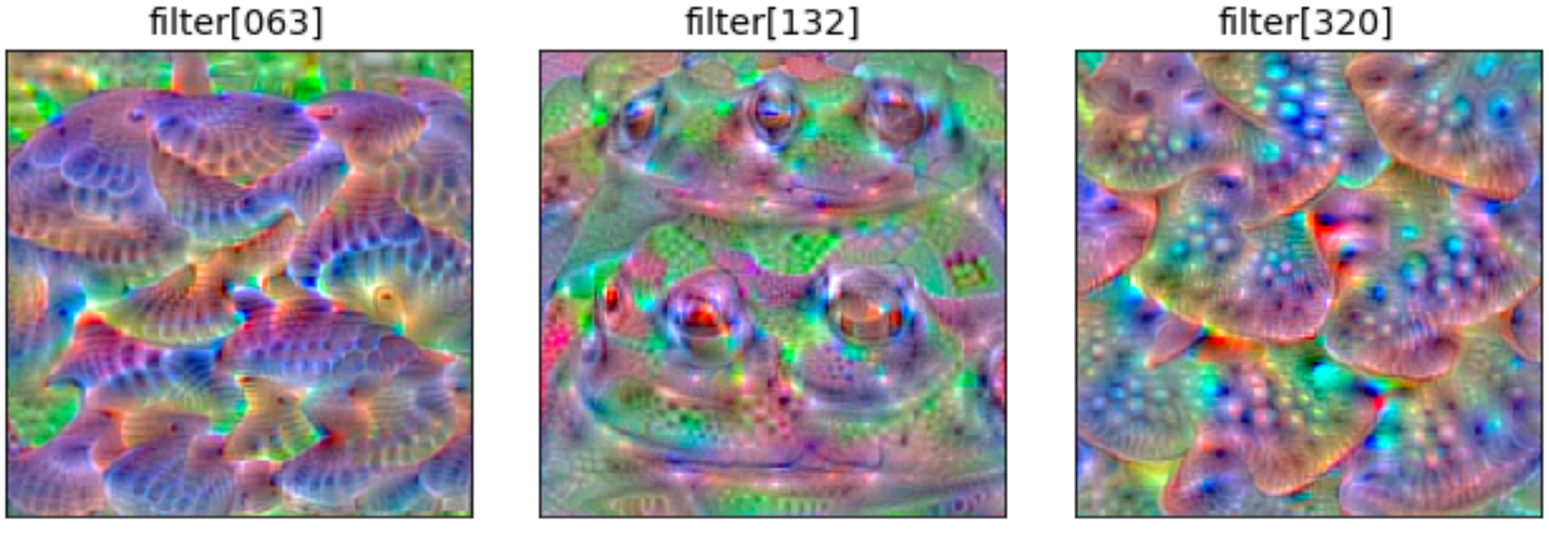


https://analyticsindiamag.com/deep-learning-is-not-so-much-a-black-box-anymore-and-thats-a-great-development/

Activation Maximization

Visualized output classification Layer

Visualized hidden convolutional layers

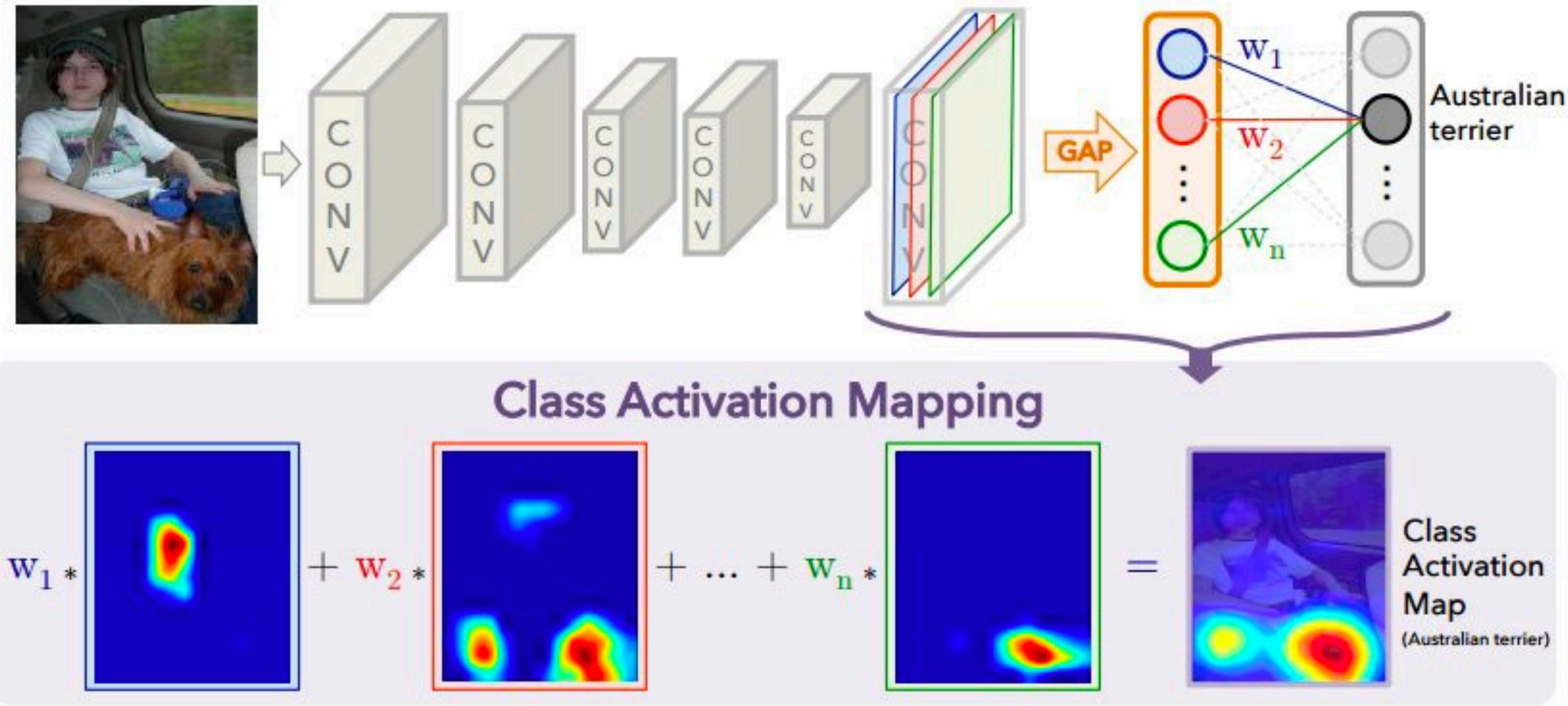


https://pypi.org/project/tf-keras-vis/

Grad-CAM heat maps

https://keras.io/examples/vision/grad_cam/

CAM heat maps



https://towardsdatascience.com/demystifying-convolutional-neural-networks-using-class-activation-maps-fe94eda4cef1

Convolutional Neural Networks Explainability in Keras

17-Explainability.ipynb

Complex programing assignment

18-House-pricing.ipynb

Thank you for your attention

e-mail: jiri@mlguru.com

Web: www.mlguru.com

Twitter: @JiriMaterna

Facebook: https://www.facebook.com/maternajiri

LinkedIn: https://www.linkedin.com/in/jirimaterna/