Spring school of AI 2

Jiří Materna

Machine Learning College

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 lecturer at ML College
- Founder and co-organizer of ML Prague
- ML Freelance and consultant

Outline

Monday: Neural networks recapitulation

Tuesday: Deep Learning

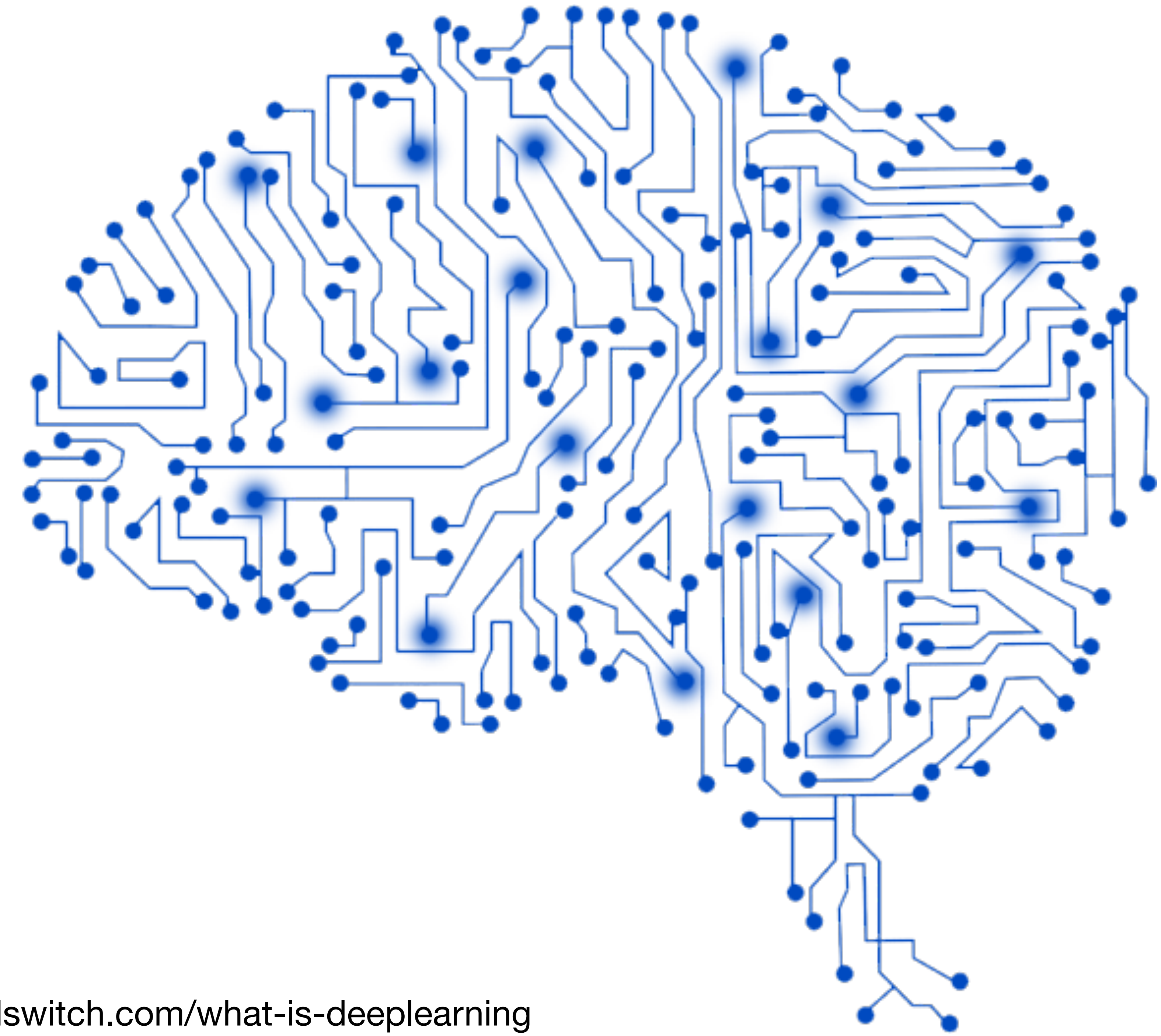
Wednesday: Image processing

Thursday: Natural Language Processing

Friday: Project

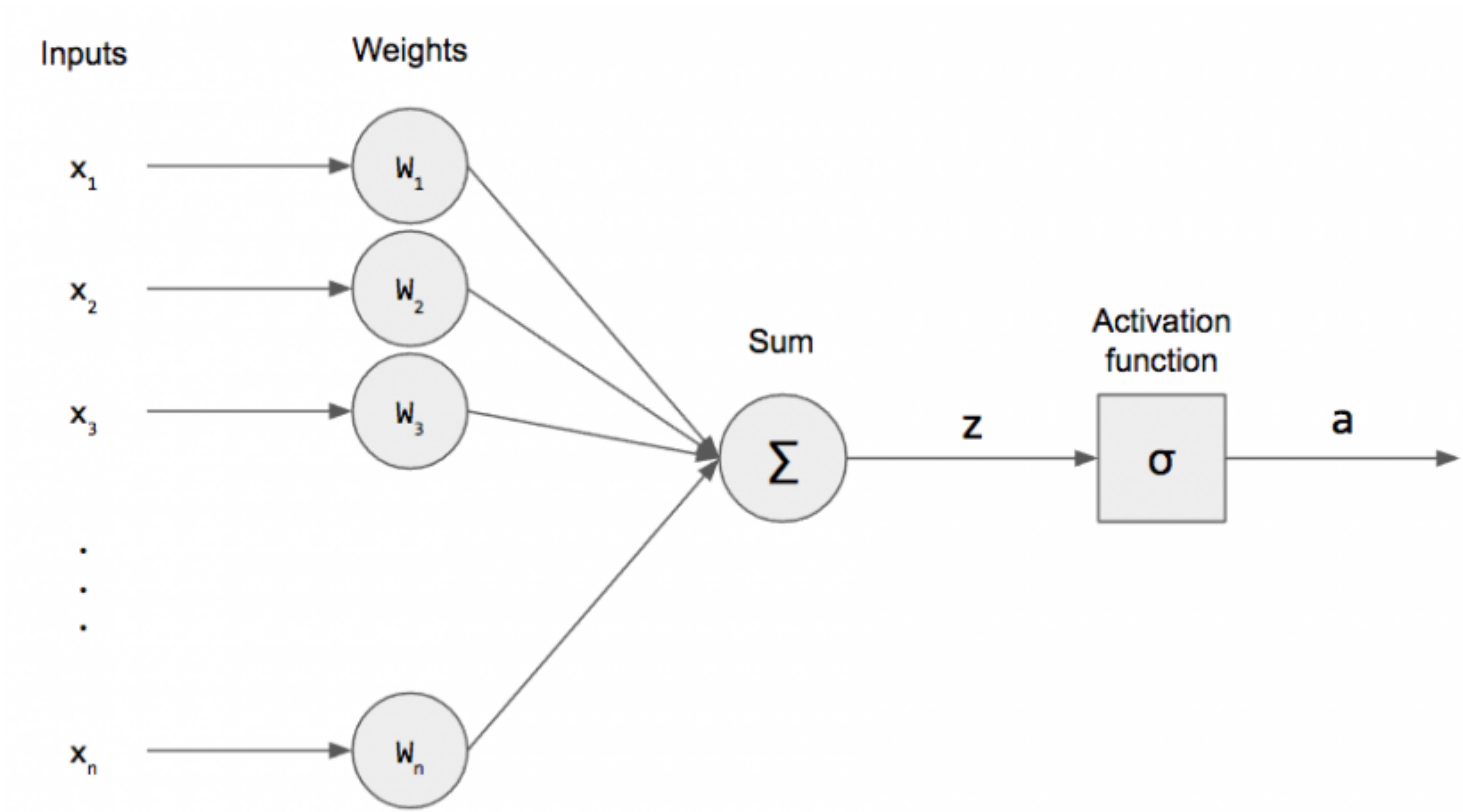
Source: http://www.rapidswitch.com/what-is-deeplearning

Neural networks and deep learning



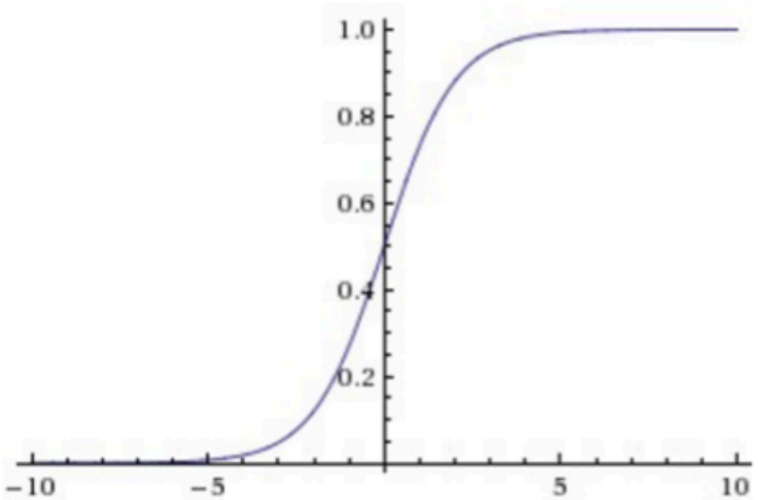
Source: http://www.rapidswitch.com/what-is-deeplearning

Perceptron

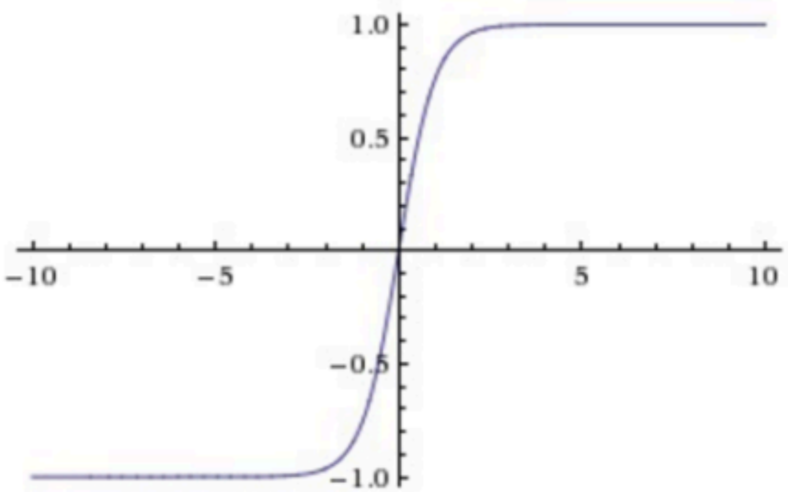


Source: https://pythonmachinelearning.pro

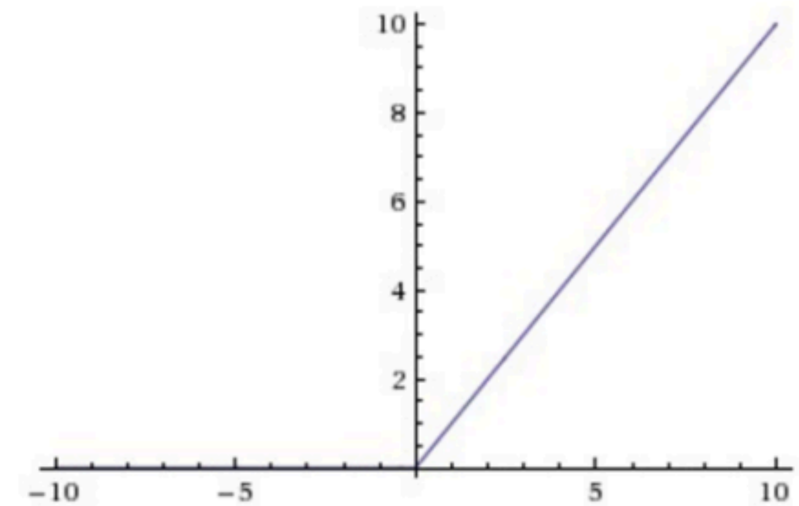
Activation functions



Sigmoid



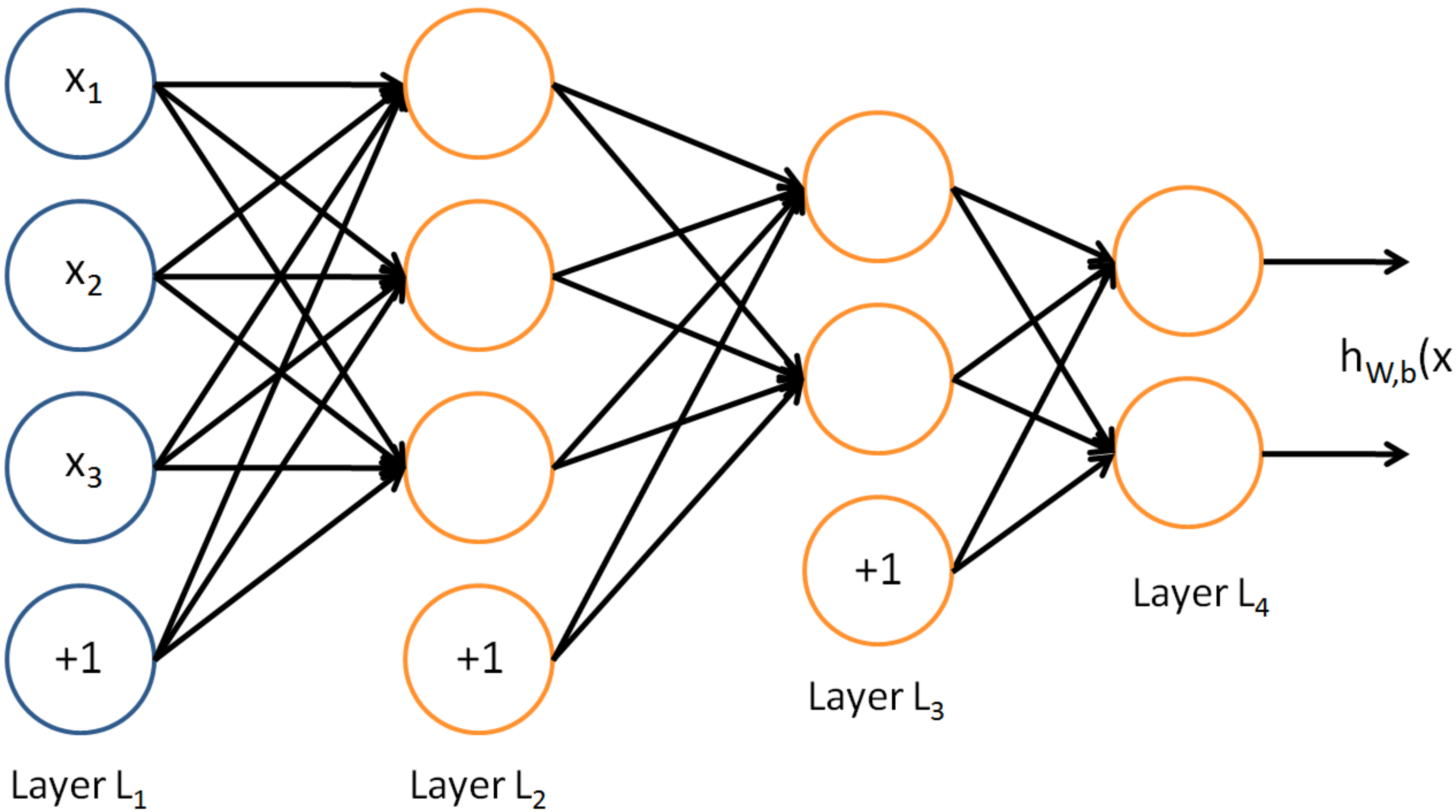
tanh



ReLU

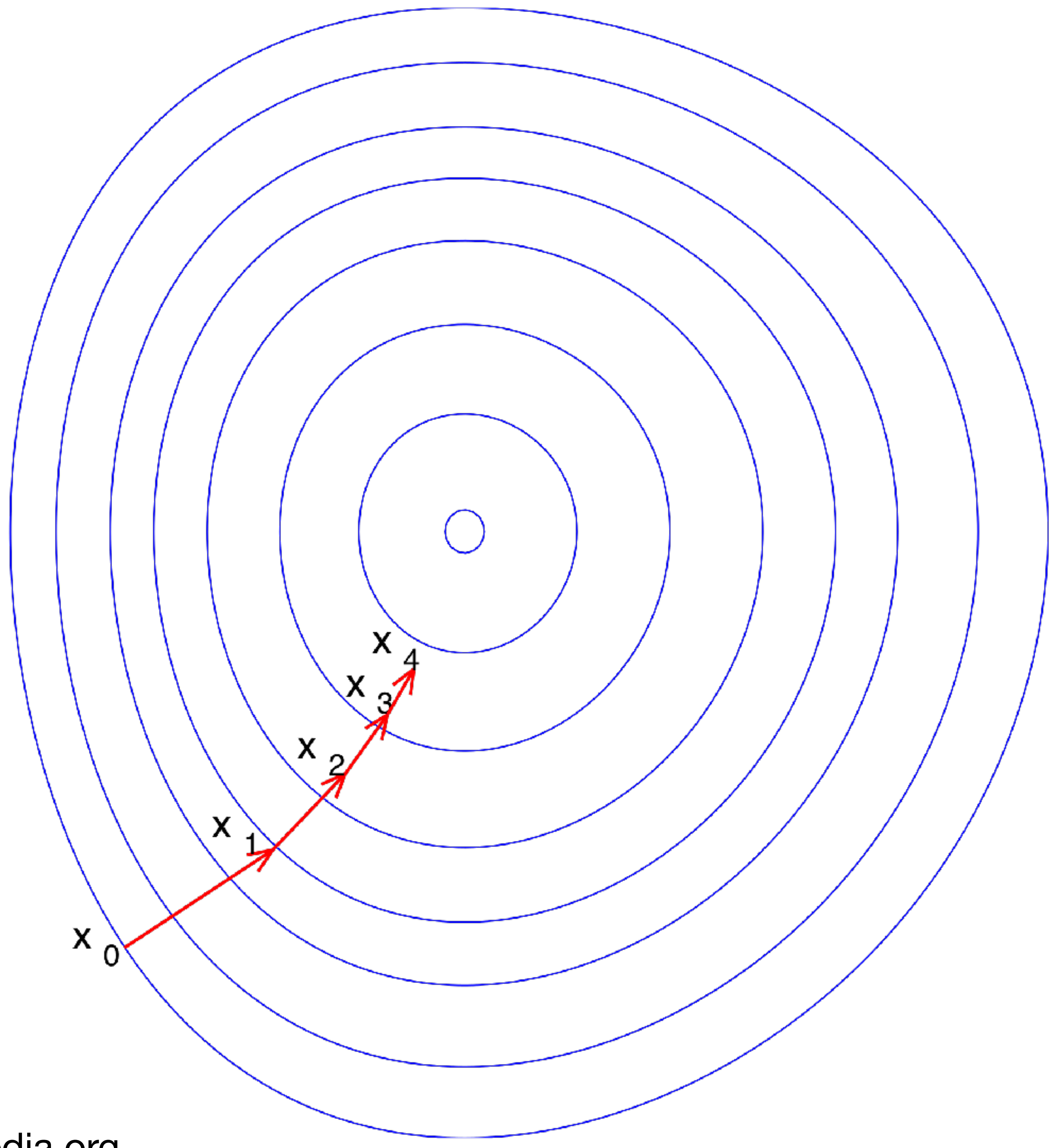
Softmax: $$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Multilayer Neural Networks



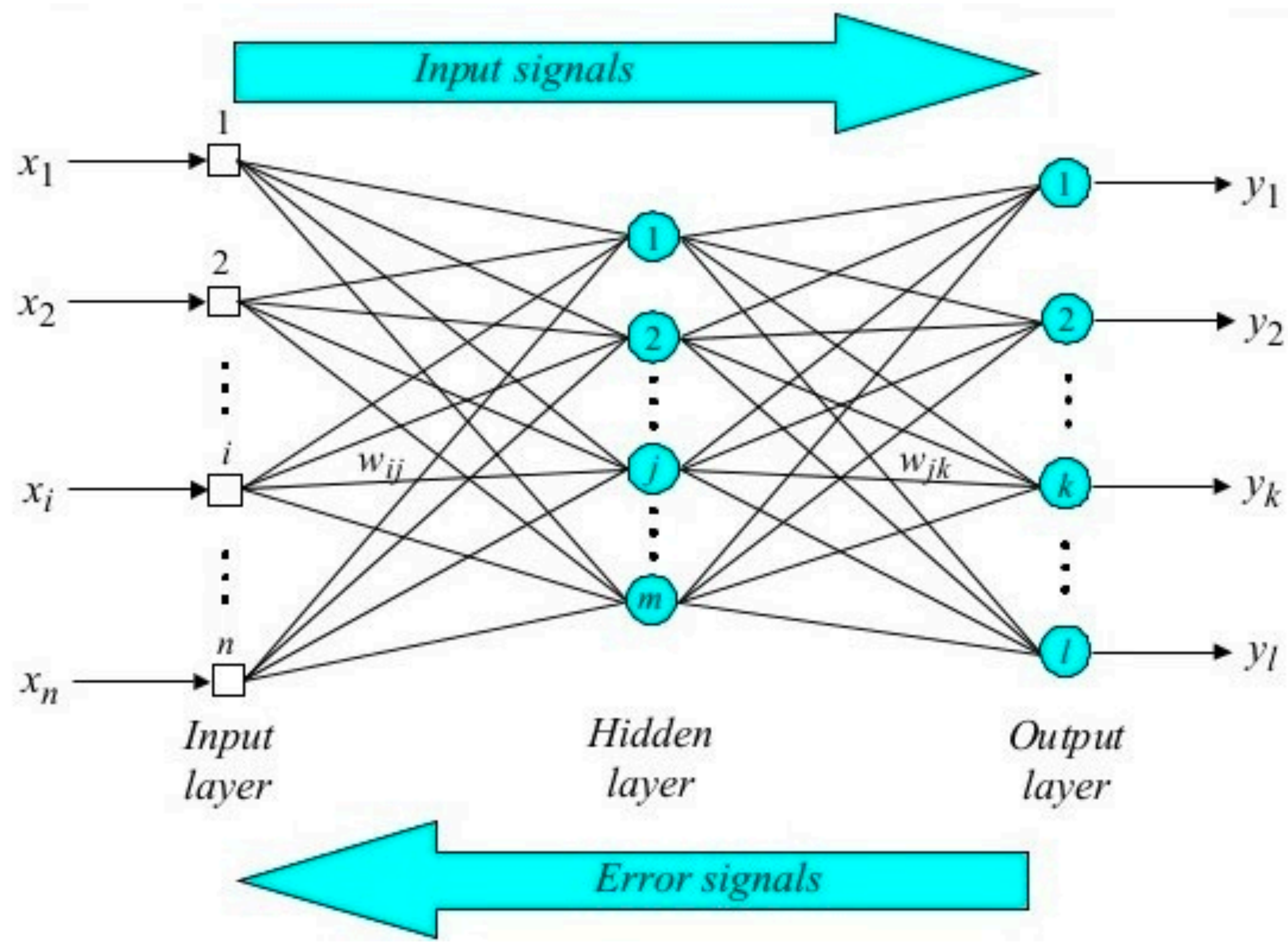
Source: http://ufldl.stanford.edu/

Steepest gradient descent



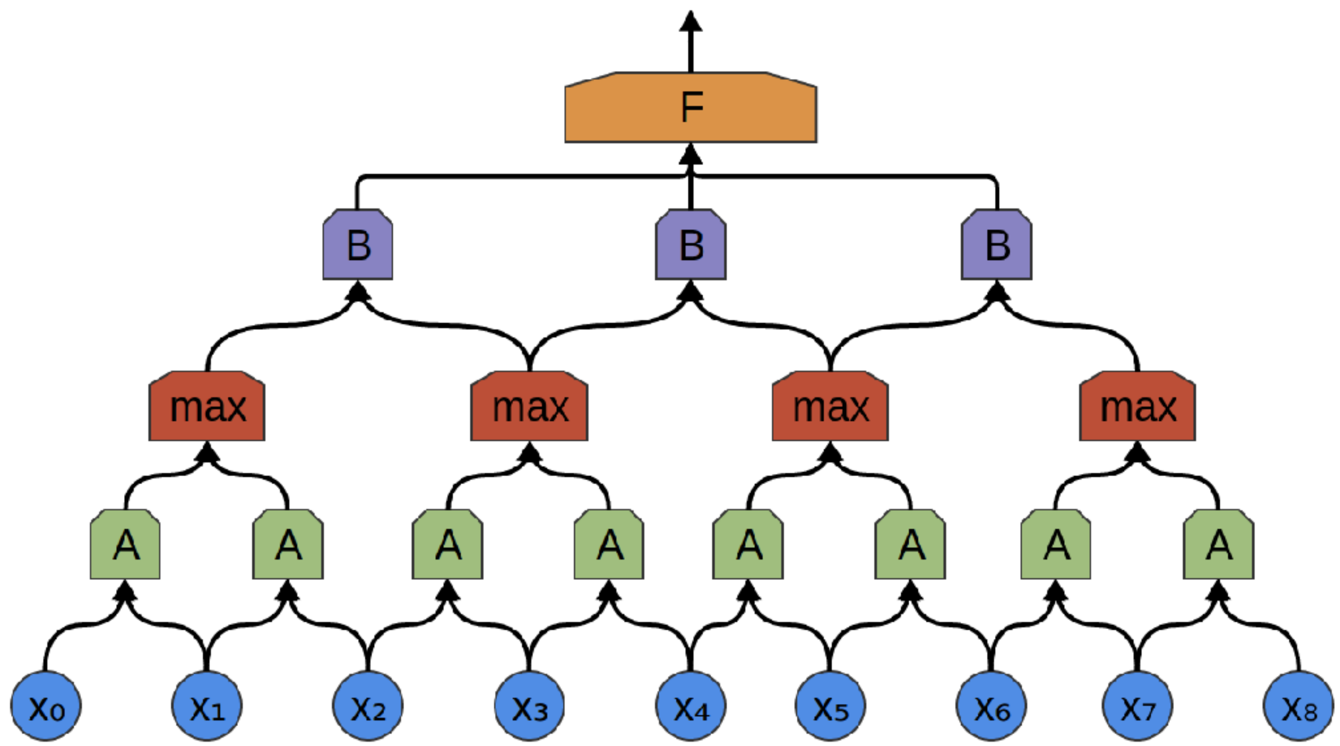
Source: https://www.wikipedia.org

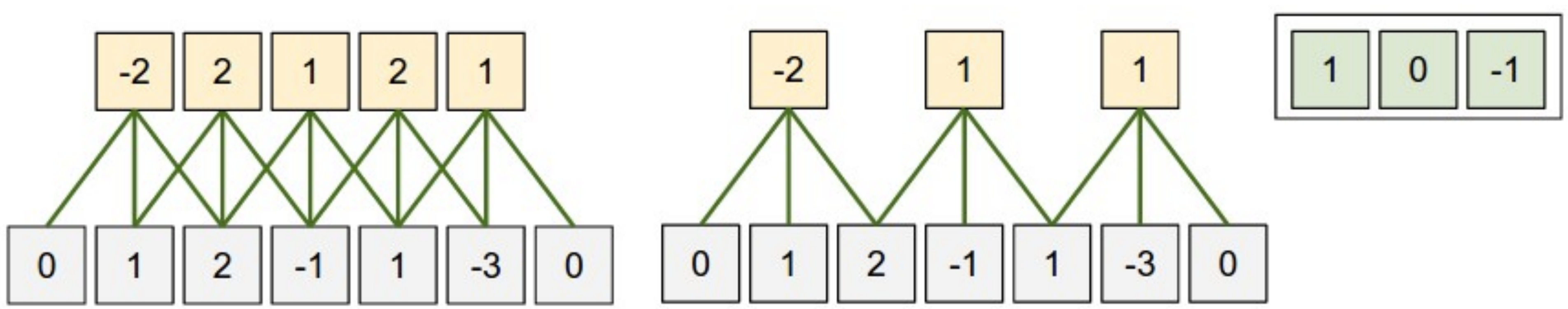
Back propagation



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

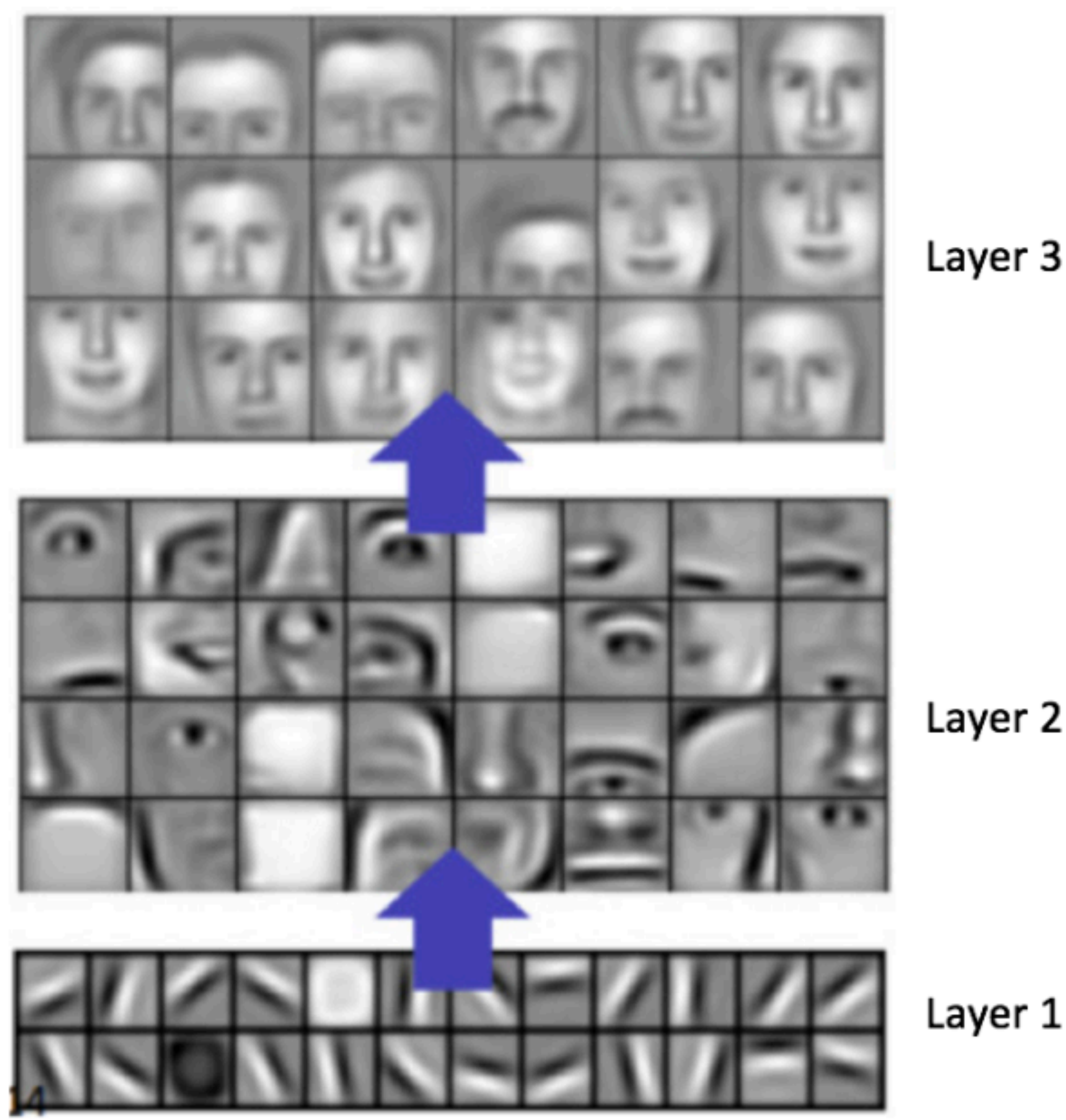
Convolution





Source: https://www.tensorflow.org

Weights visualization

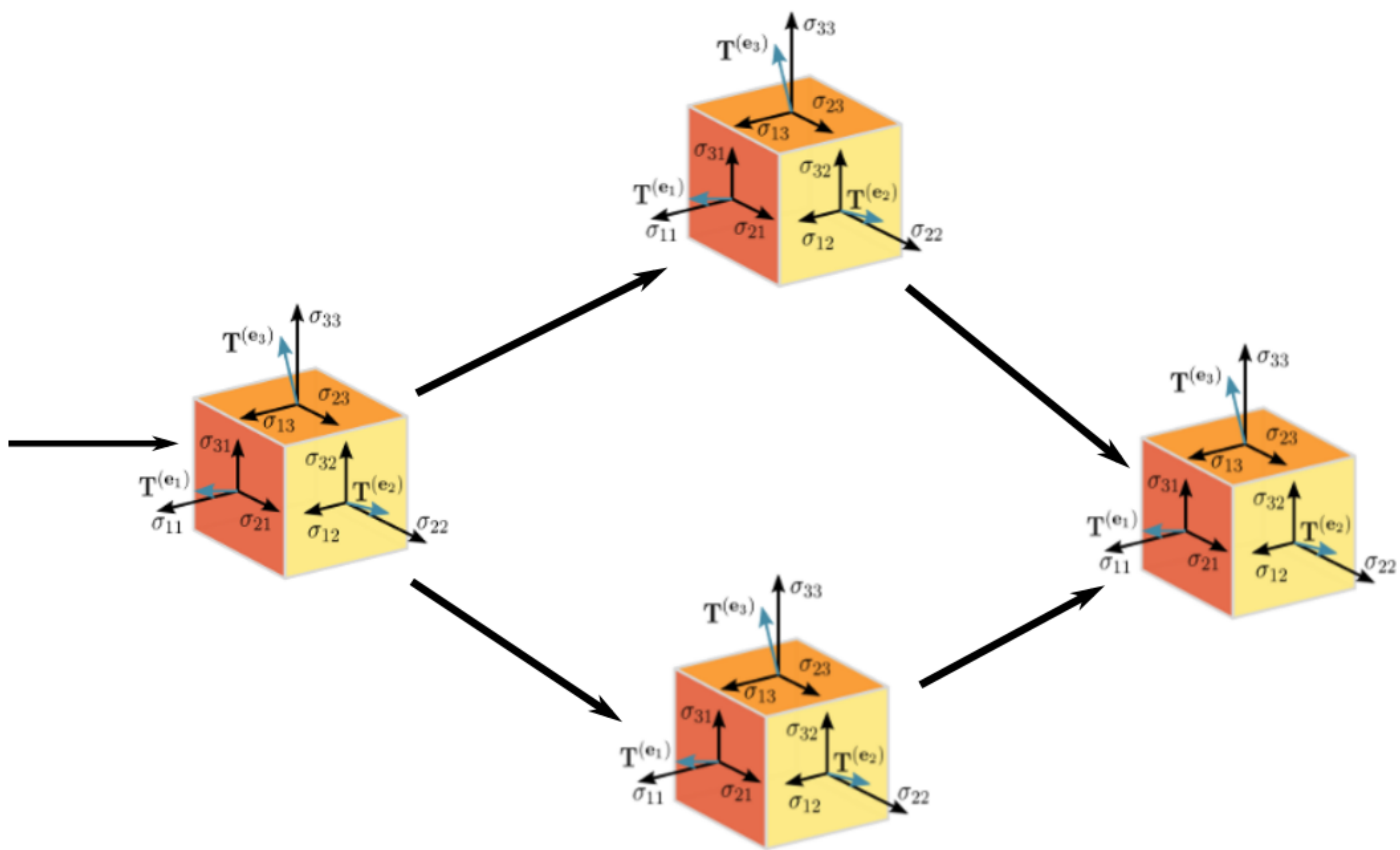


Important terms

- deep learning
- stochastic gradient descent
- batch and mini-batch learning
- epoch

What is not TensorFlow

What is TensorFlow?



Keras tutorial

01-Keras-introduction.ipynb

Implemenation of classification and regression tasks using NN

02-Classification-nn-assignment.ipynb

03-Regression-nn-assignment.ipynb

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

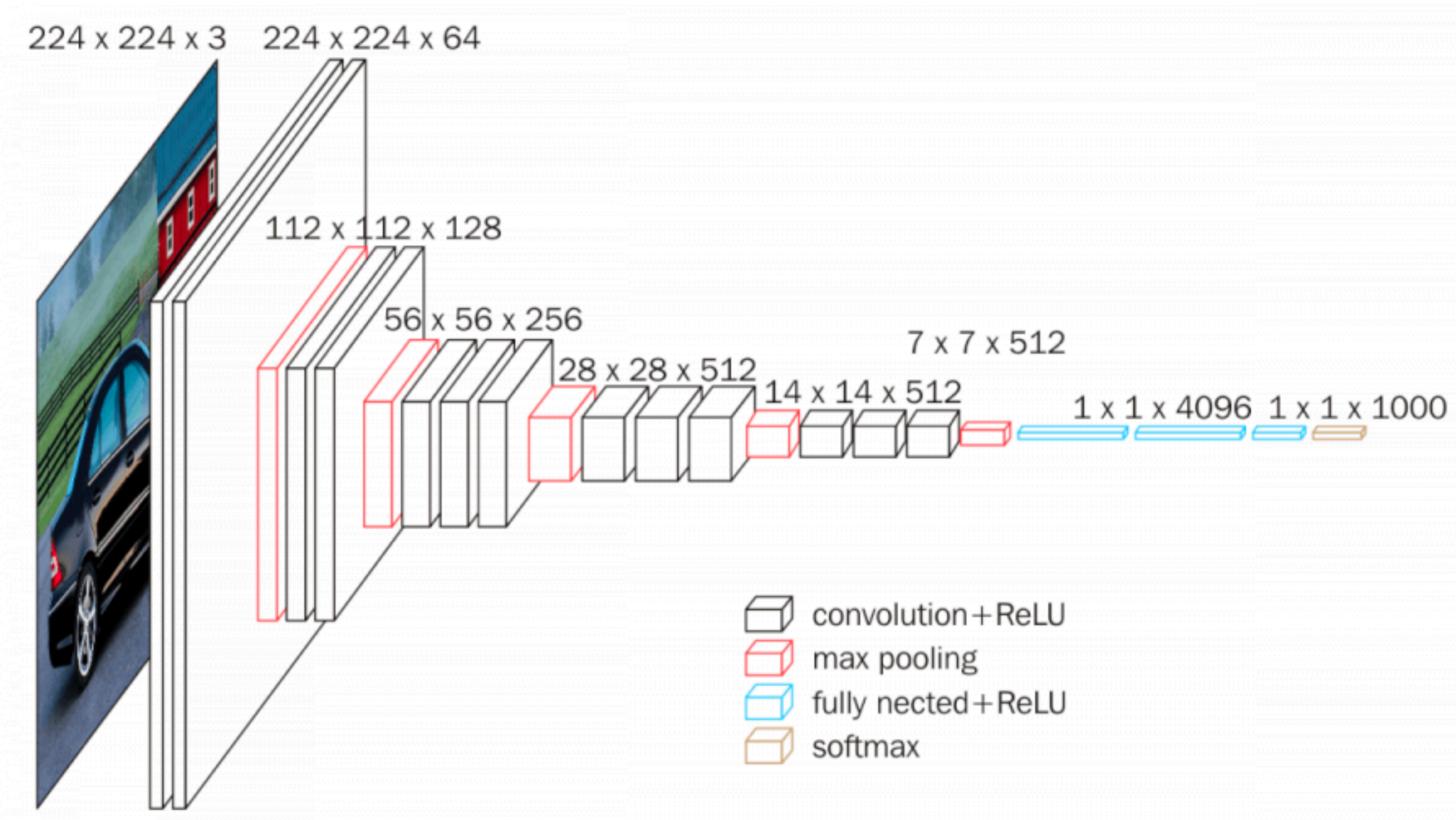
Autoencoder network and Restricted Blotzmann Machines

Transformer network

U-Net

Generative adversarial network

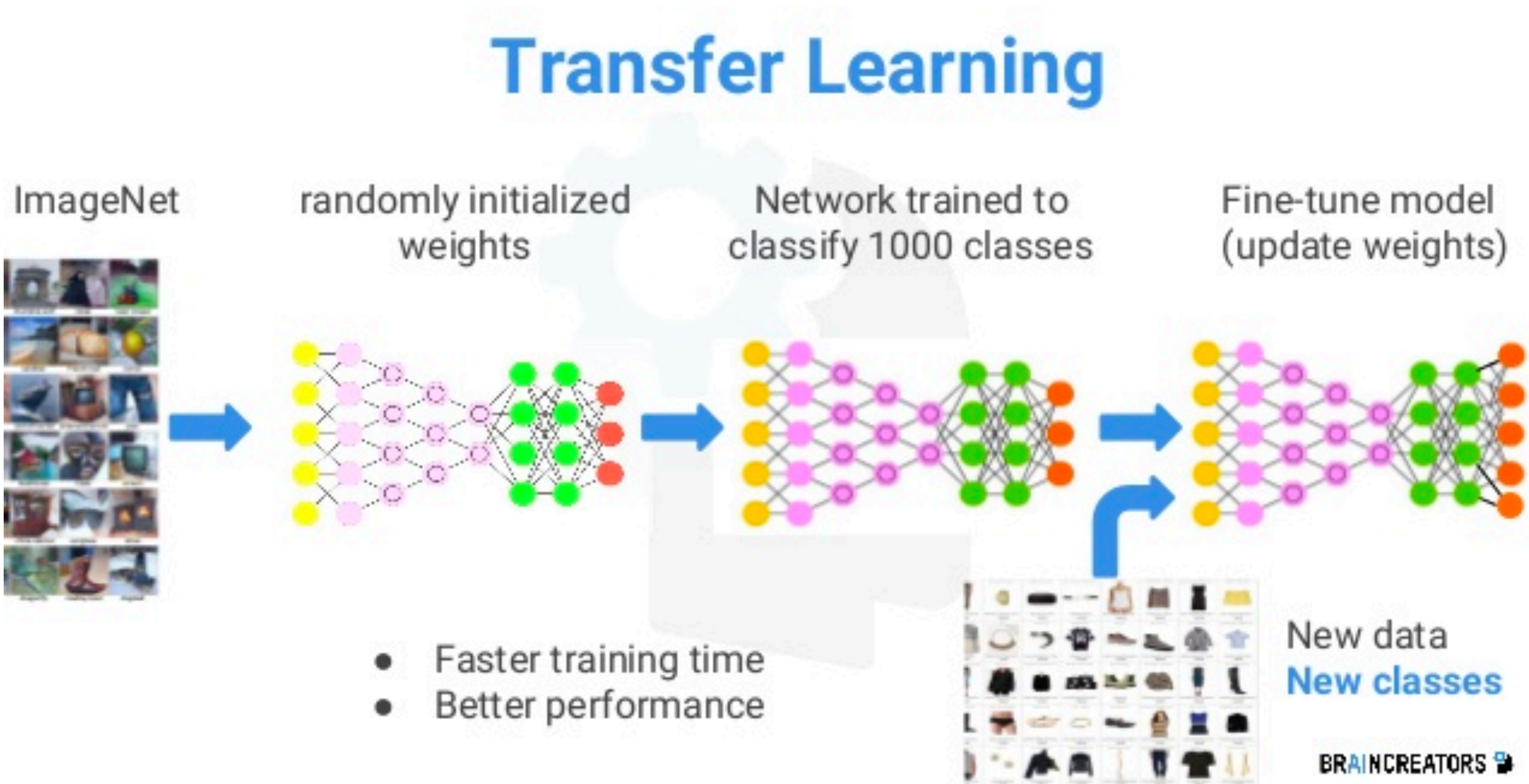
VGG 16


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

ResNet

residual connection
x
weight layer
relu
weight layer
relu
identity
Residual Network
Plain Network
image
pool, /2
avg pool
fc 1000

Finetuning

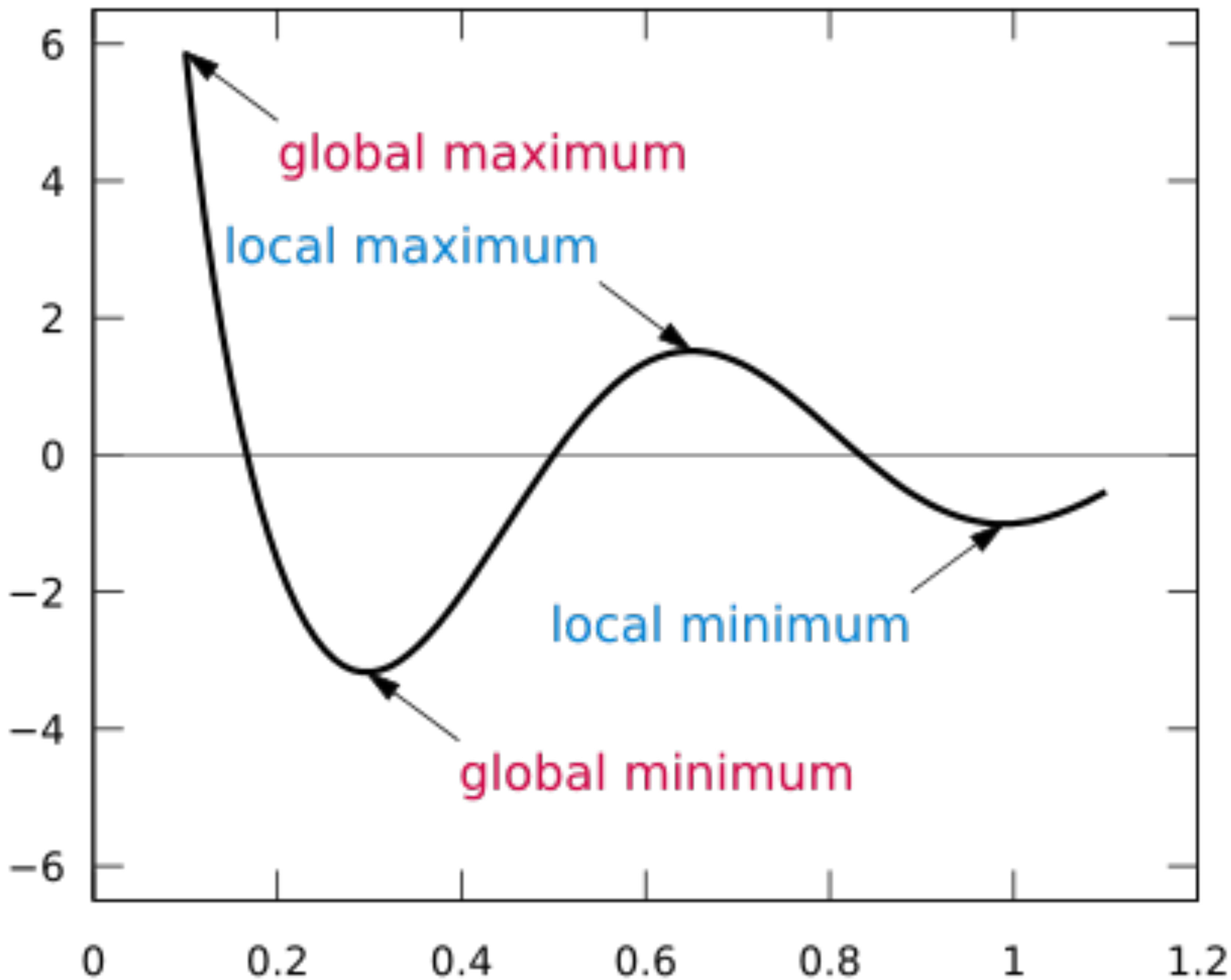


Transfer Learning
ImageNet
randomly initialized weights
Network trained to classify 1000 classes
Fine-tune model (update weights)
New data
New classes
Faster training time
Better performance
BRAINCREATORS

Transfer learning example

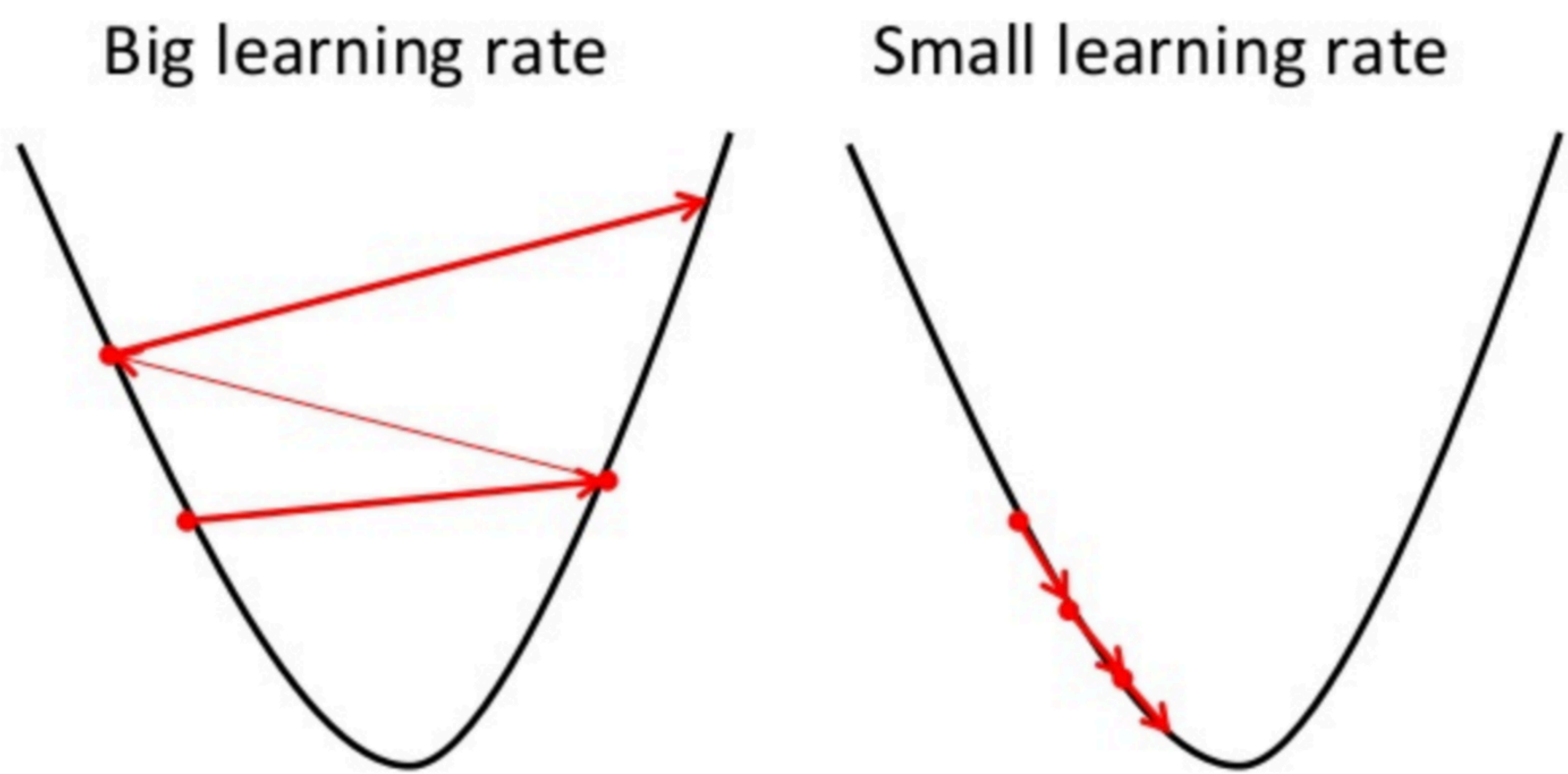
04-Transfer_learning.ipynb

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

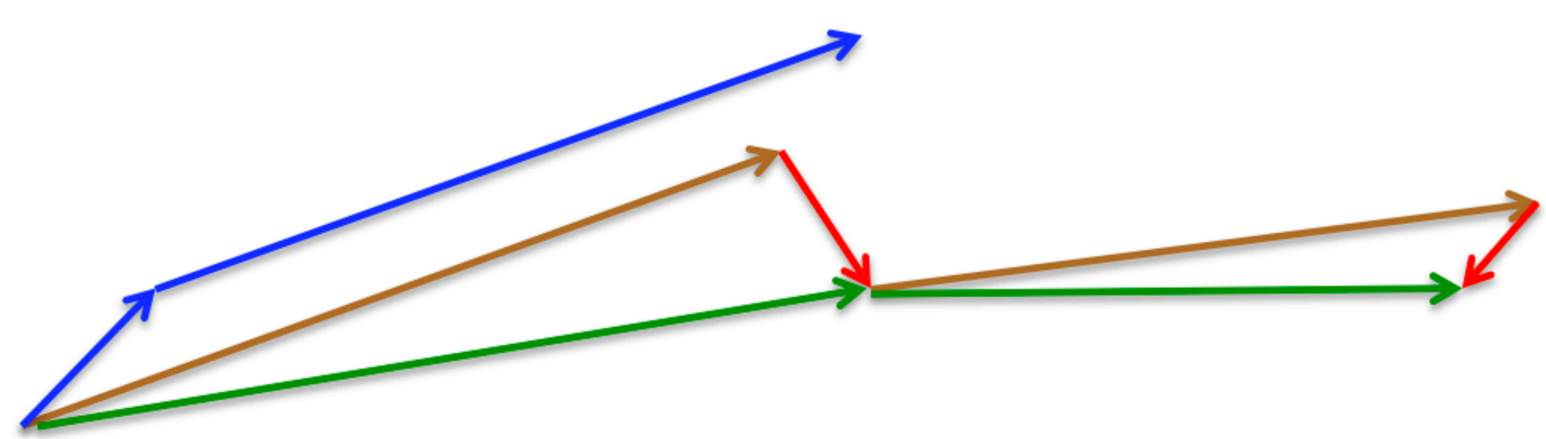
$$v_t = \gamma v_{t-1} + \lambda \frac{\partial e(w_t)}{\partial w_t}$$

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

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)

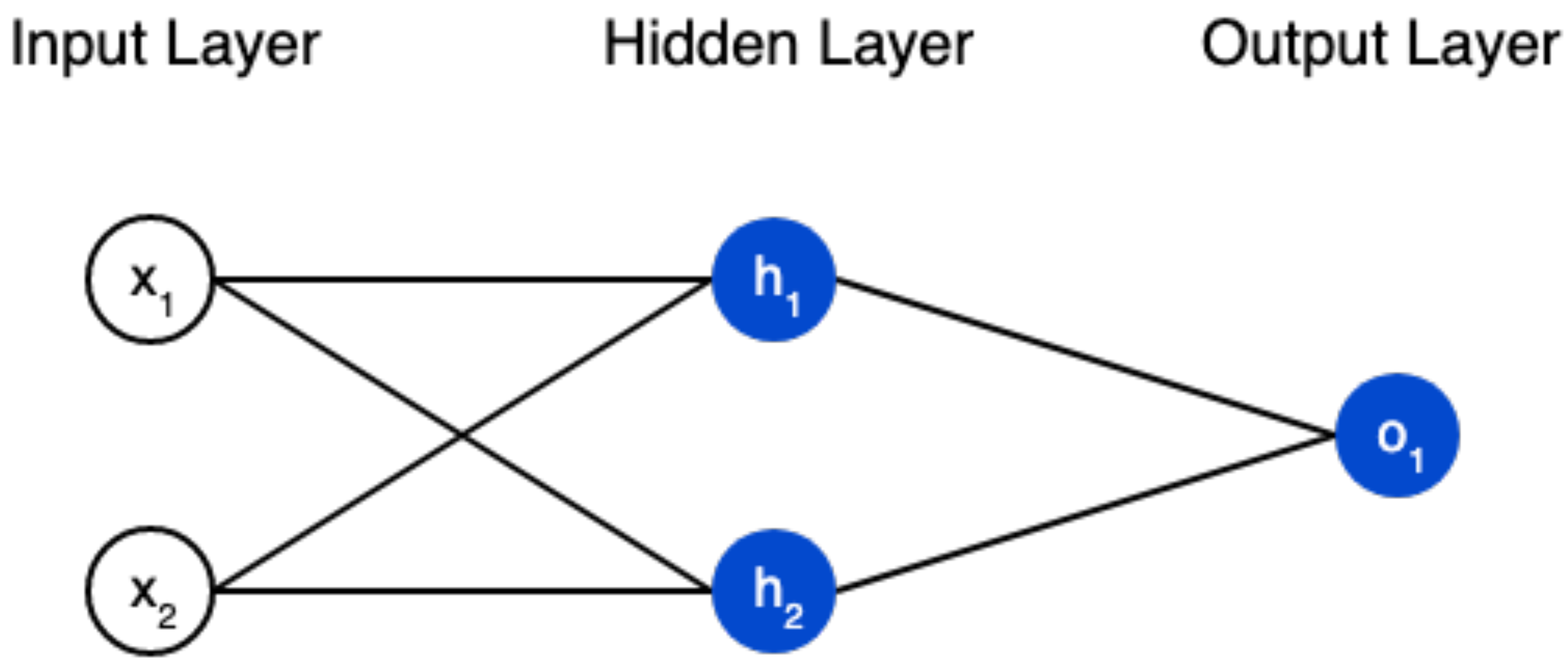
Categorical Cross Entropy $$CCE = -\frac{\sum_{i=1}^{n}\sum_{j=1}^{c} y_{i,j} log(\hat{y}_{i,j})}{n}$$

Binary Cross Entropy

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

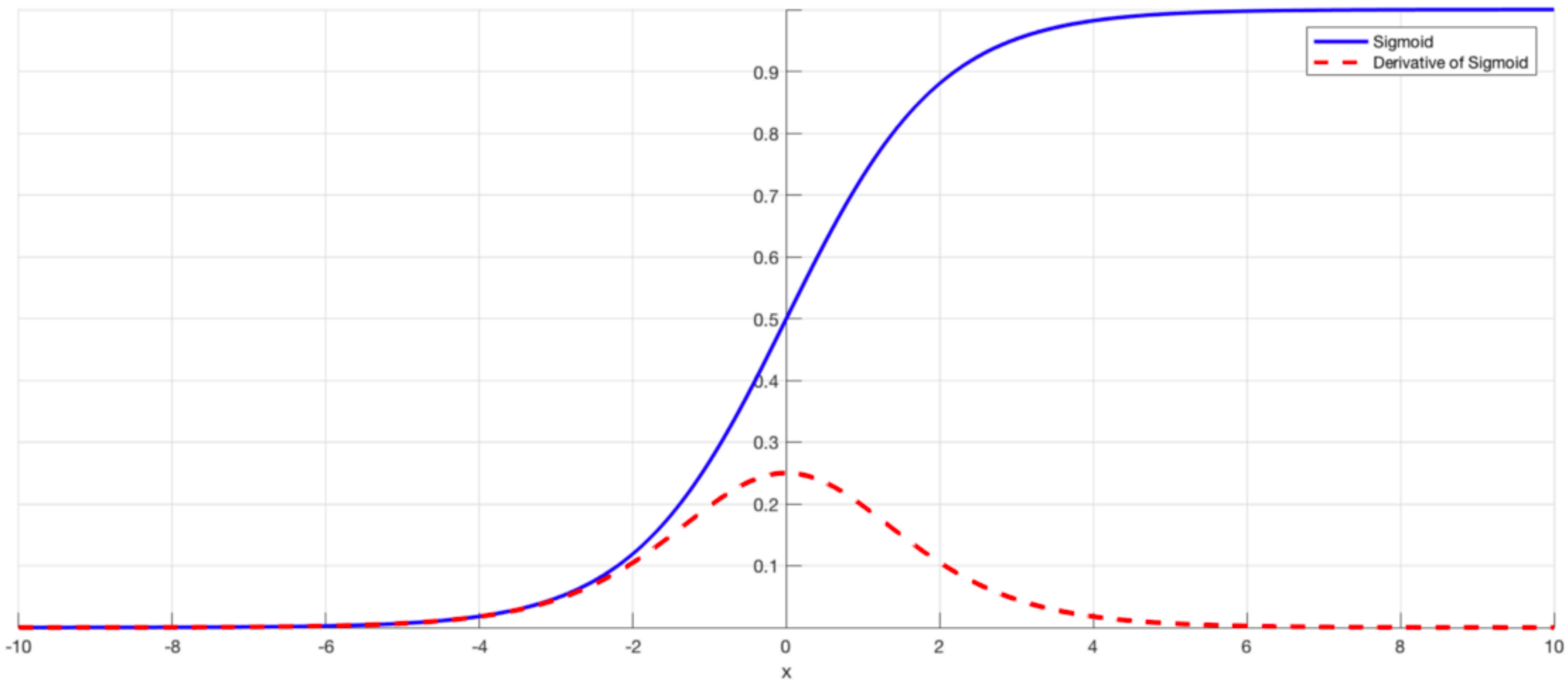
Weight initialization

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

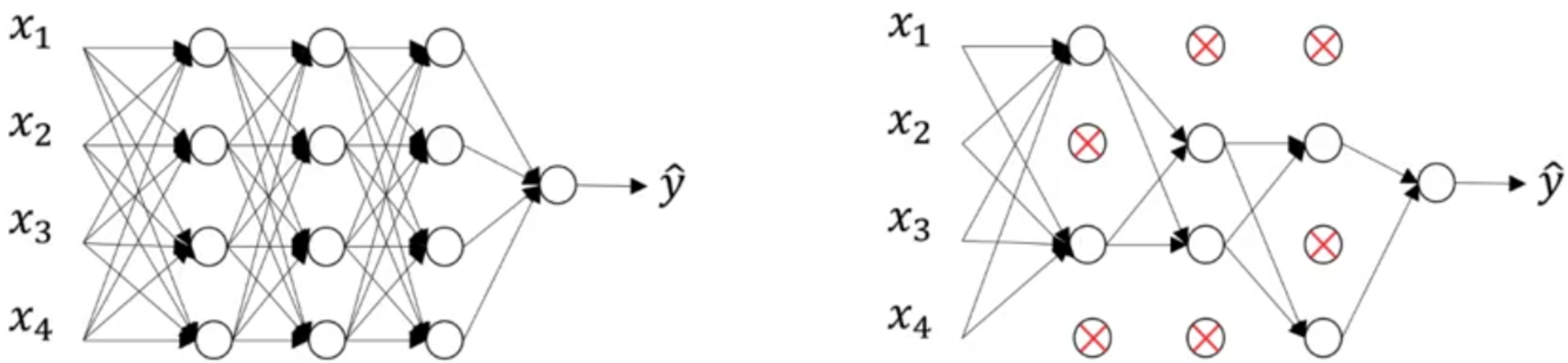
05-Regression-nn-assignment.ipynb

L2 Regularization in deep learning

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

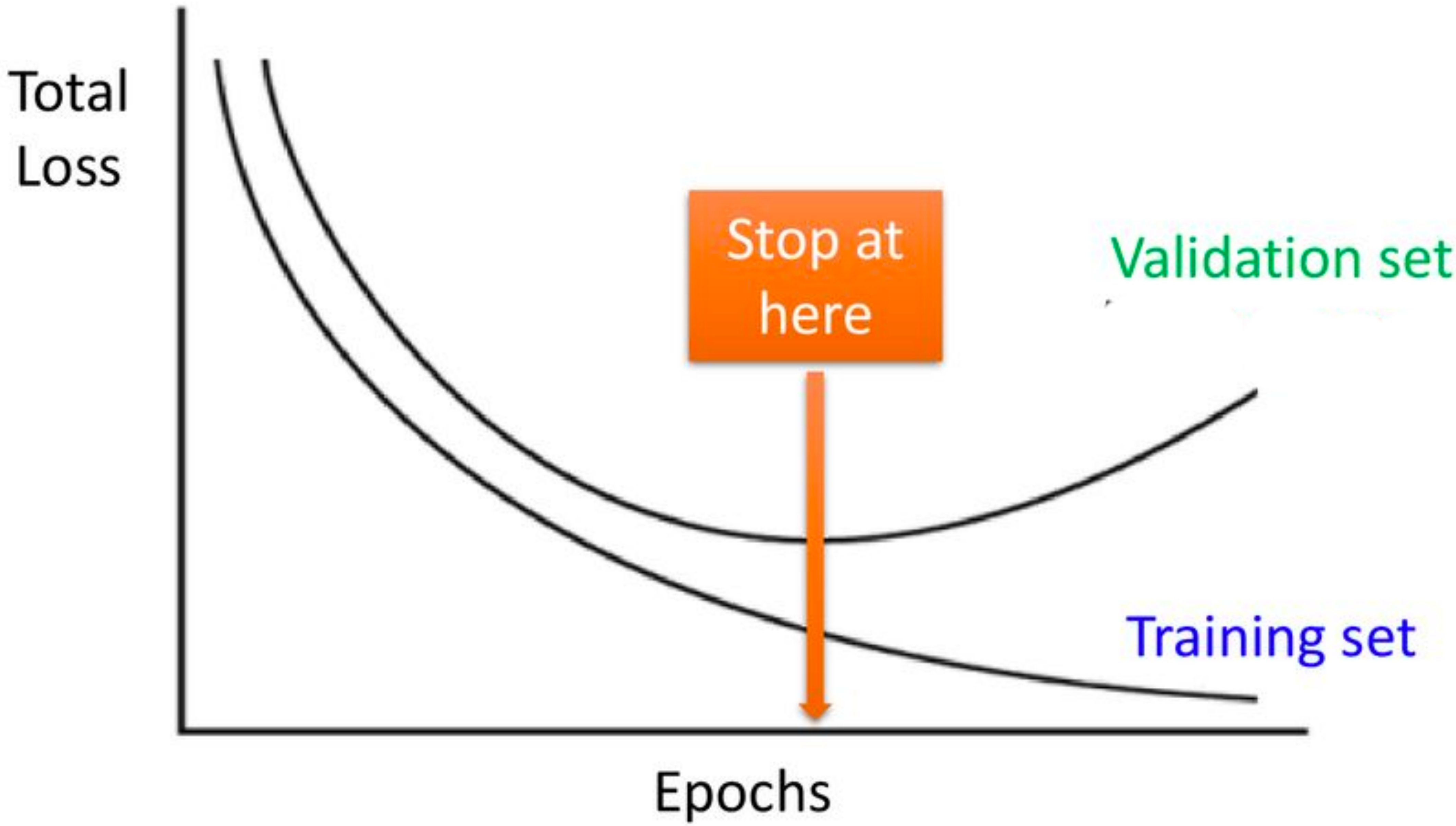
Frobenius norm $$||w||_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} |w_{i,j}|^2}$$

Dropout



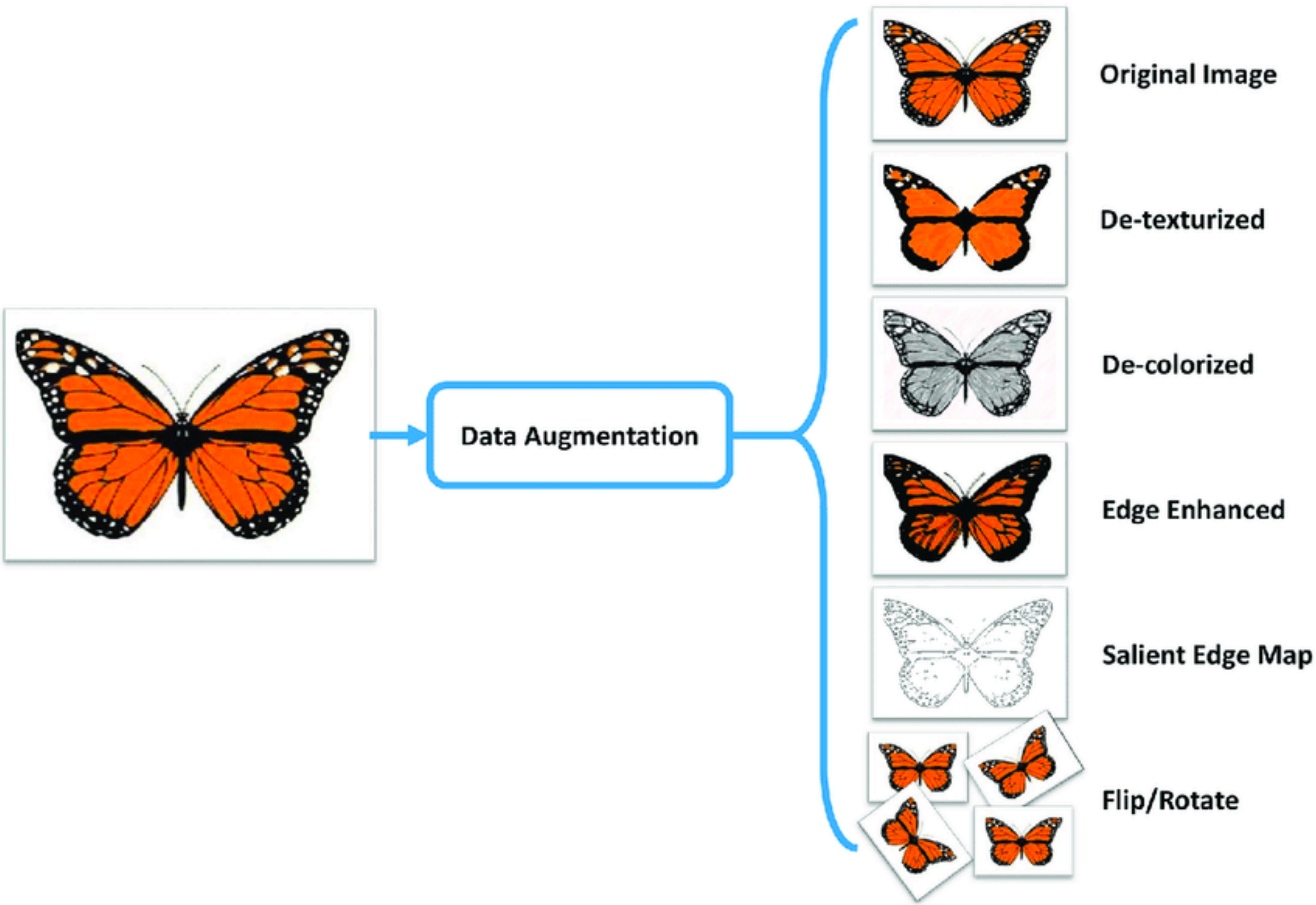
https://towardsdatascience.com/how-to-improve-a-neural-network-with-regularization-8a18ecda9fe3

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: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

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

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

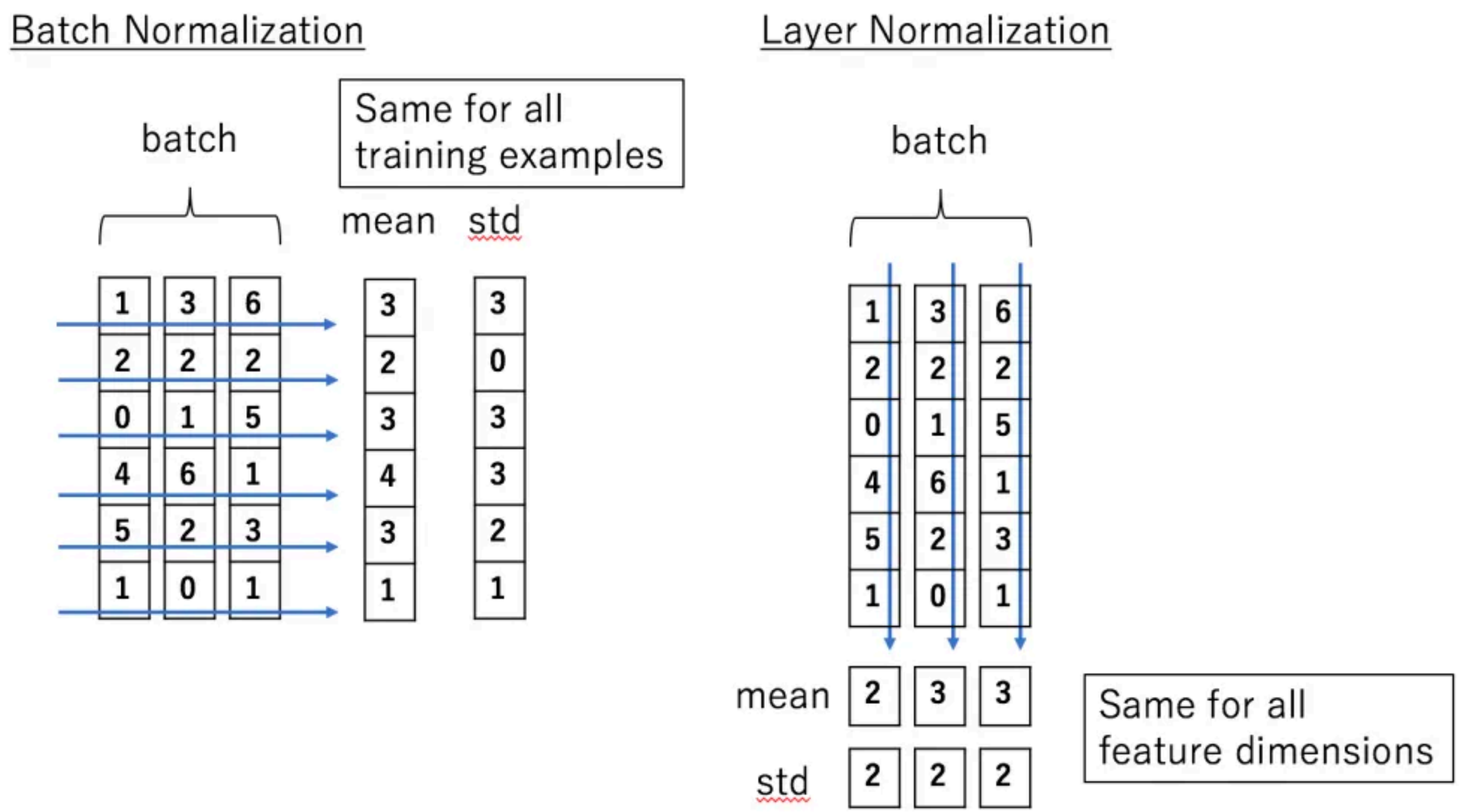
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

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

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

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)
```

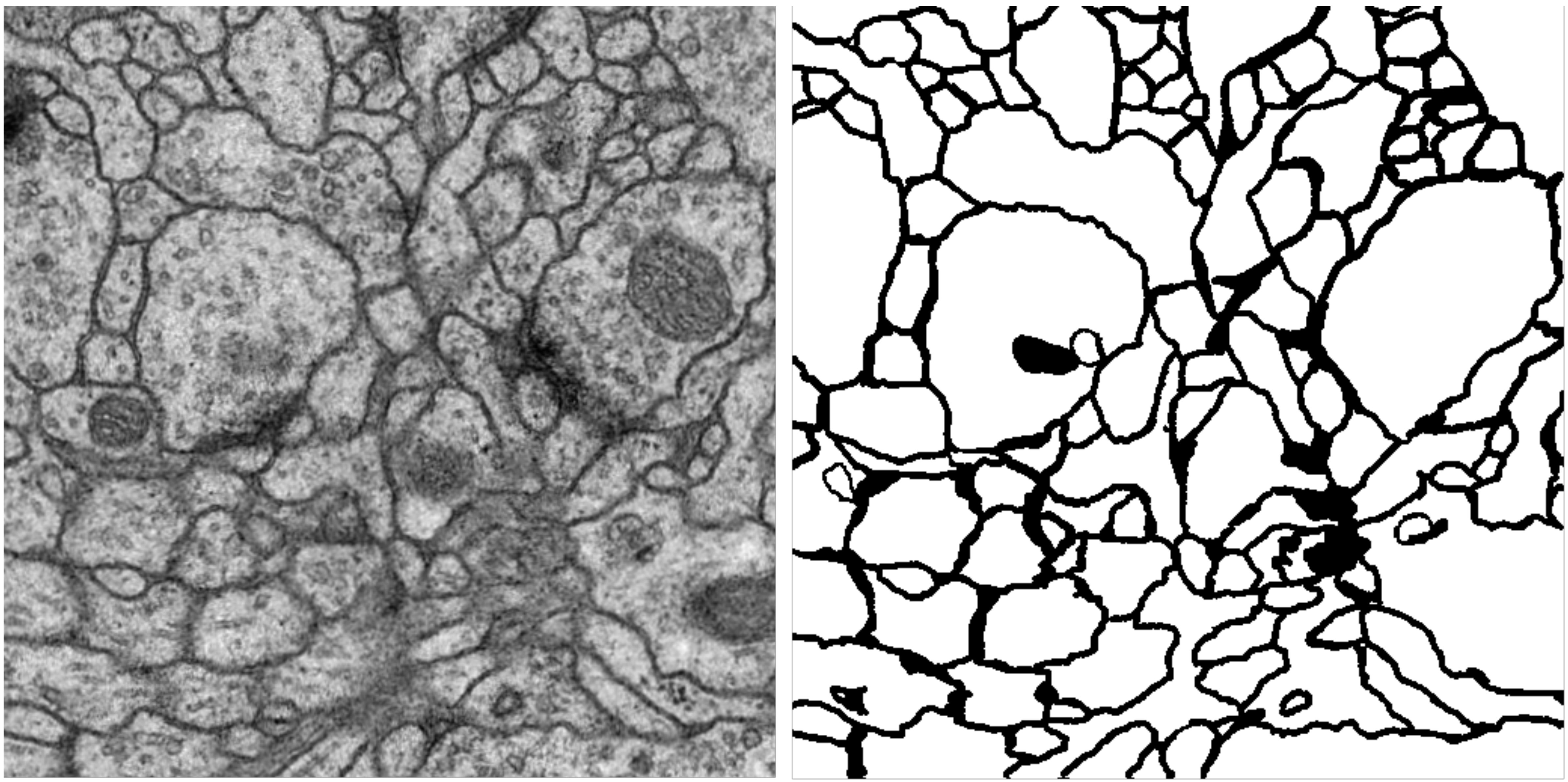
Multiple inputs (outputs)

```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

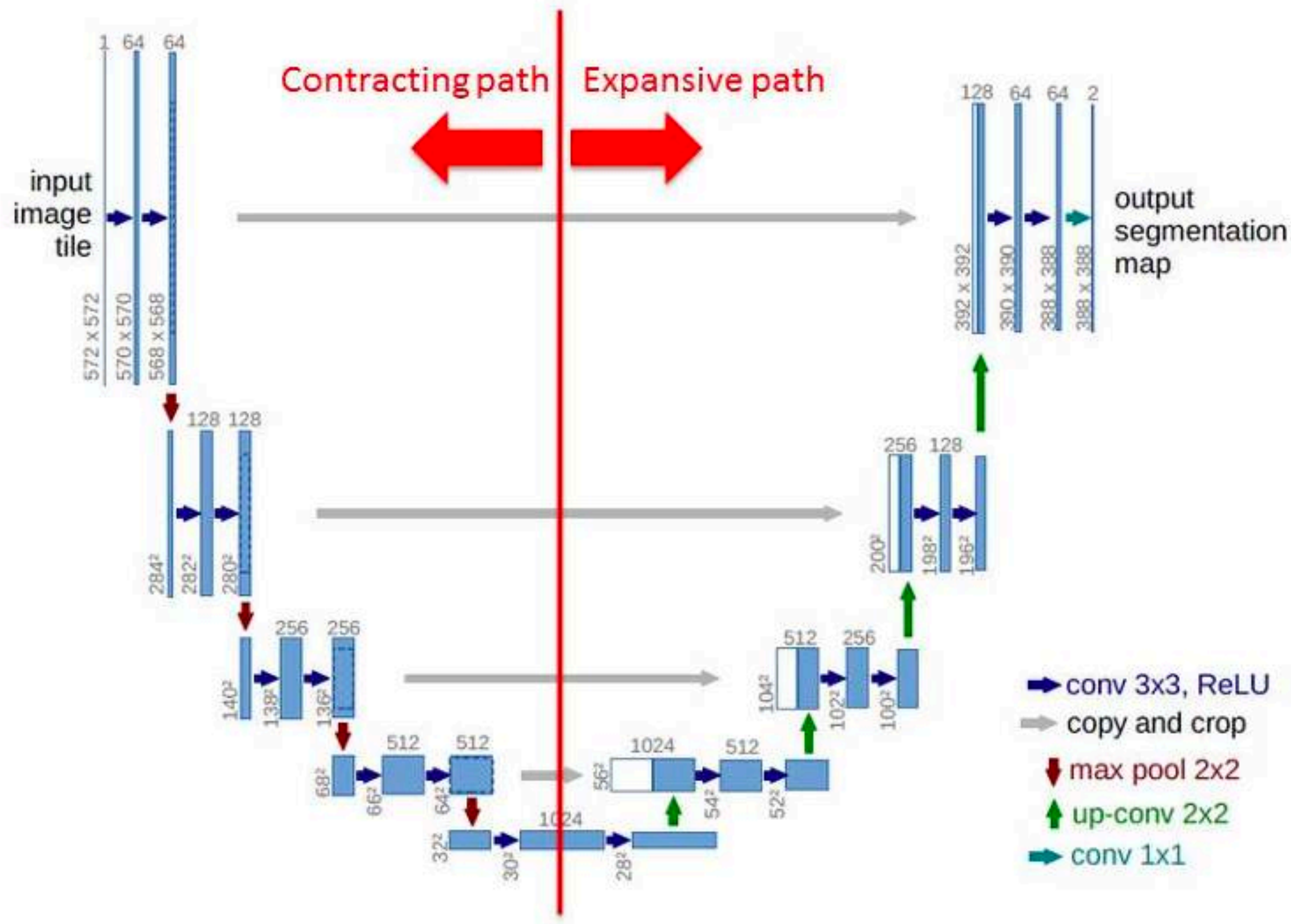
06-Normalization-and-regularization-assignment.ipynb

Image segmentation



U-Net

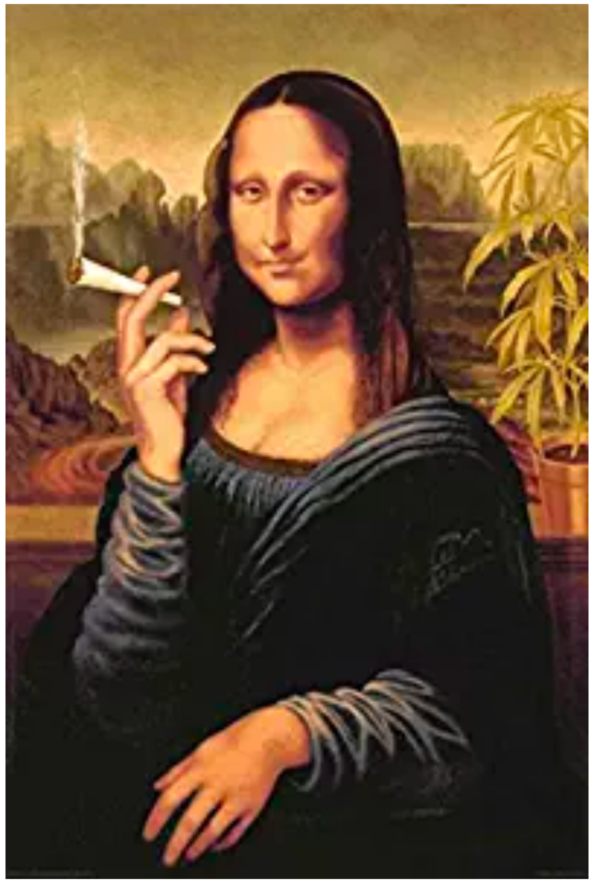
Network Architecture



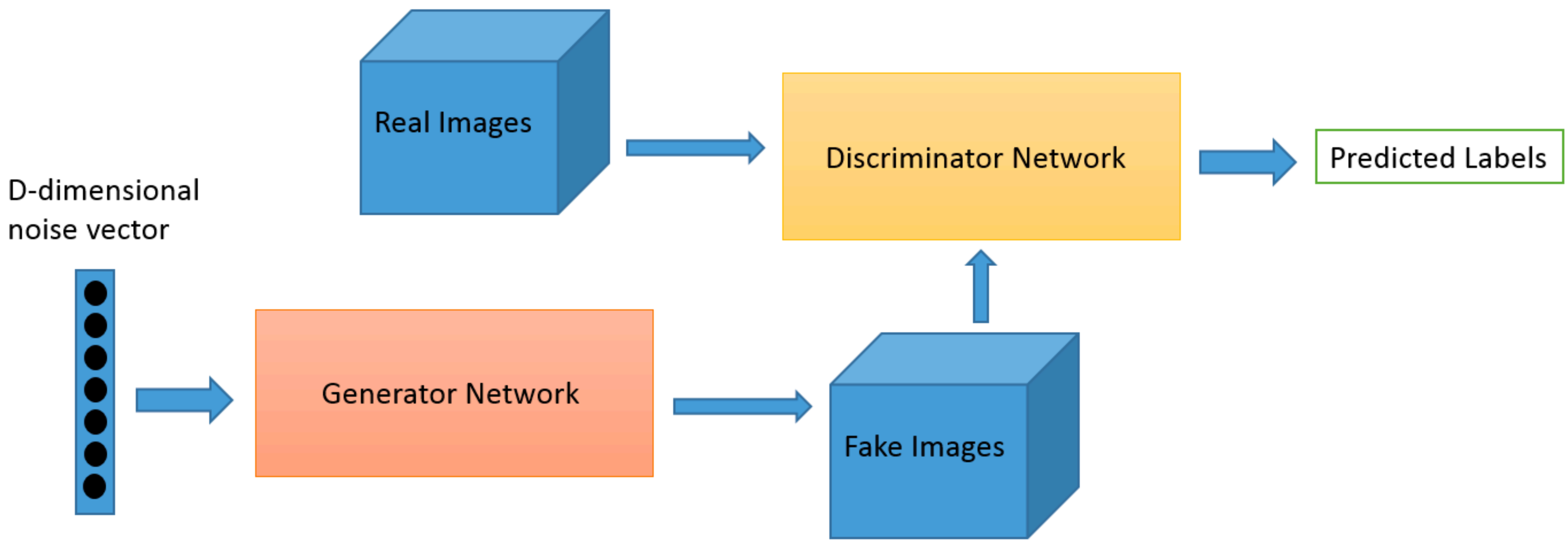
U-Net segmentation example

07-Segmentation.ipynb

Generative models with neural networks

Random noice → Artificial Neural network → 

Generative Adversarial Networks



Superresolution

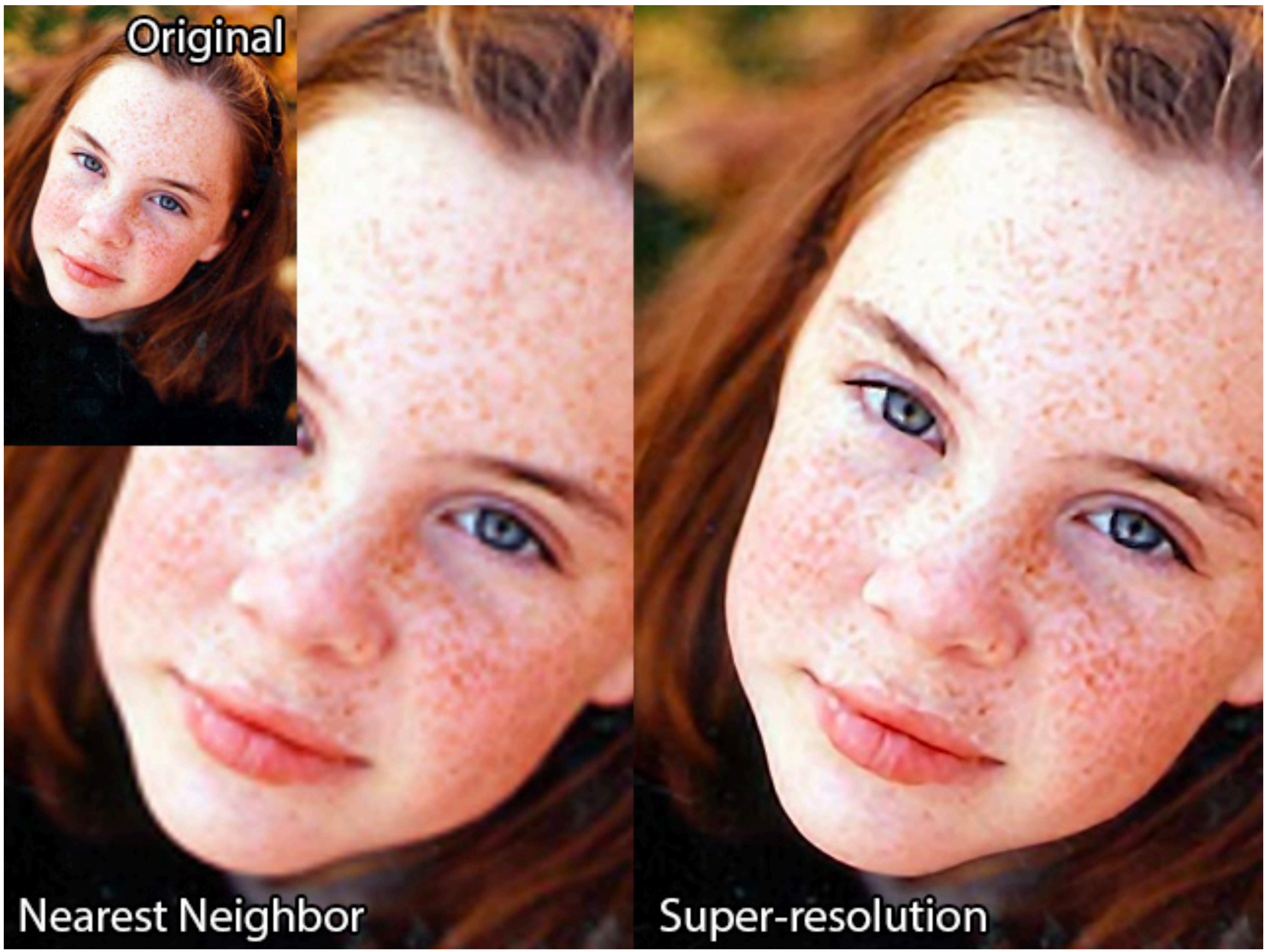
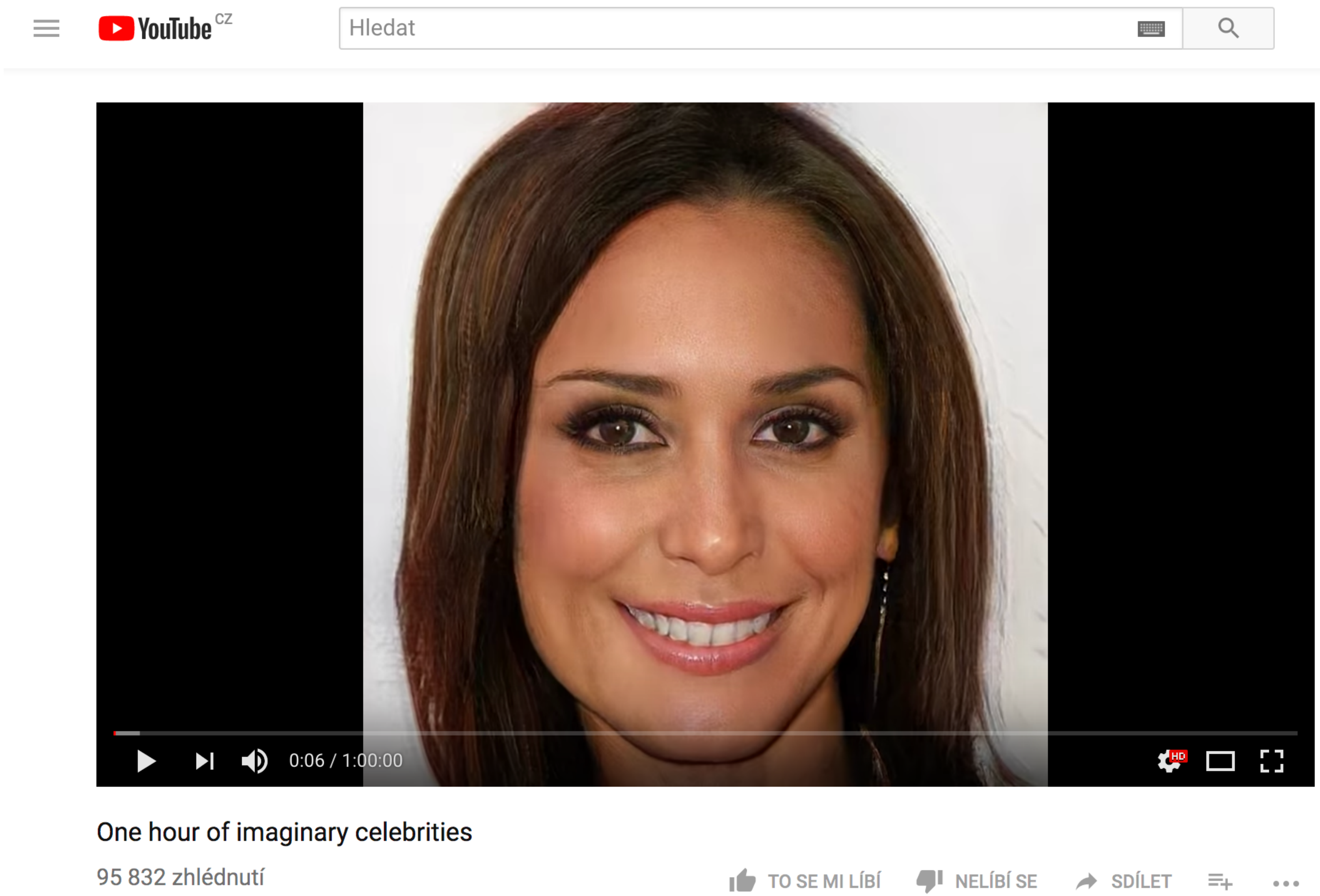


Image synthesis



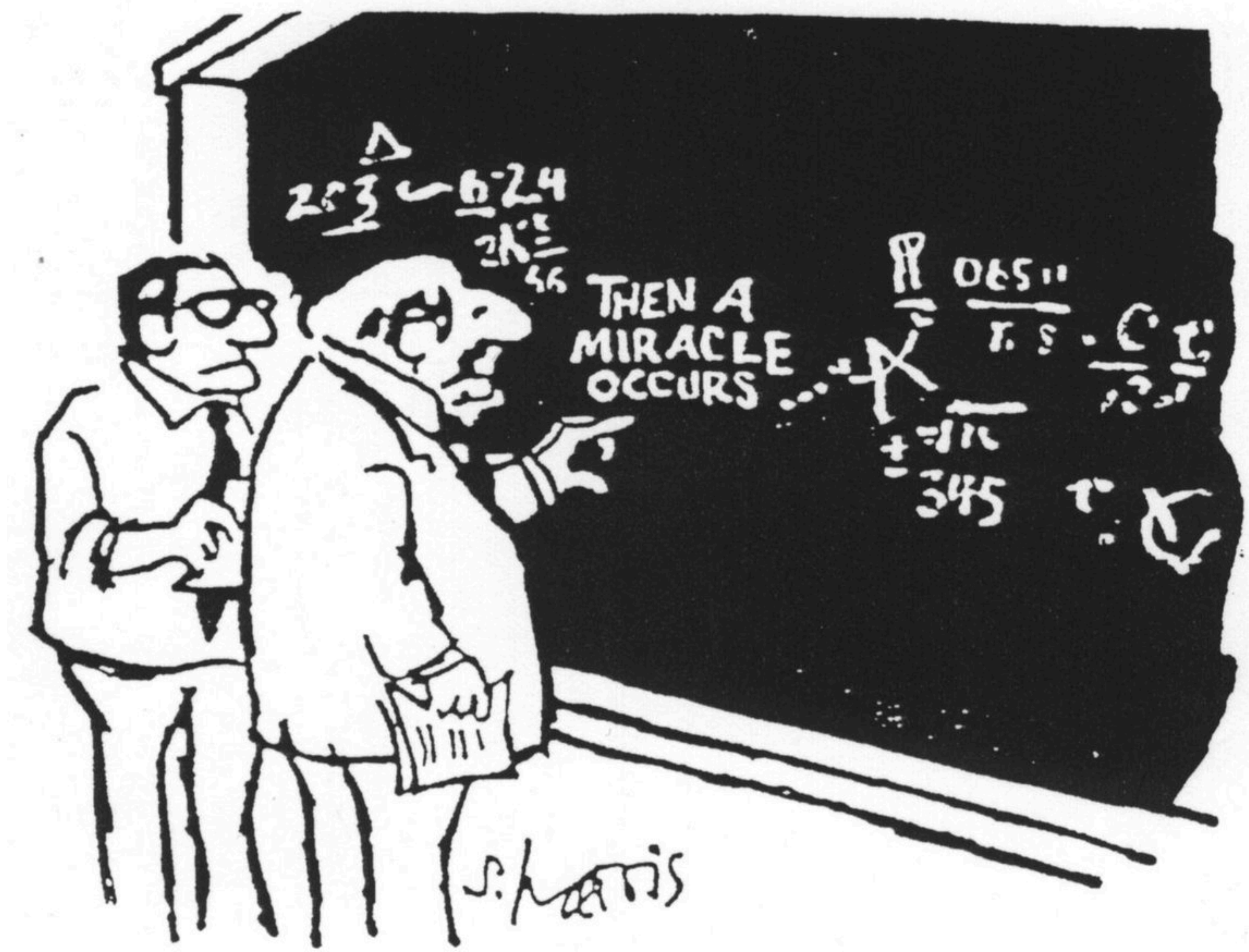
Which one is fake?

Generative Adversarial Networks

08_GANs.ipynb

Image manipulation

Neural network explainability



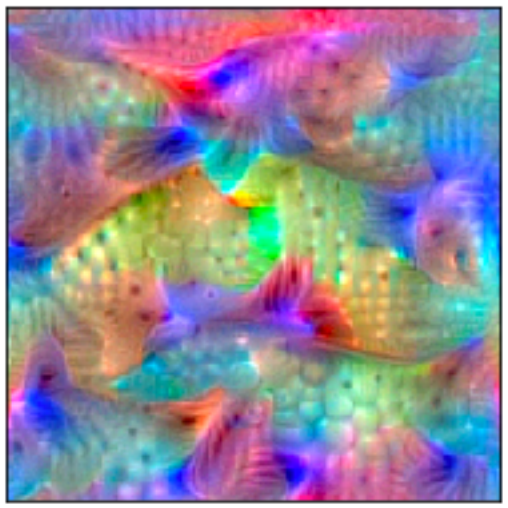
I think you should be a little more specific, here in Step 2

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

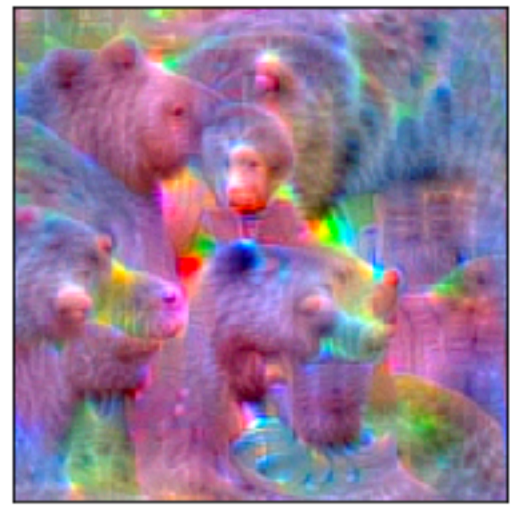
Activation Maximization

Visualized output classification Layer

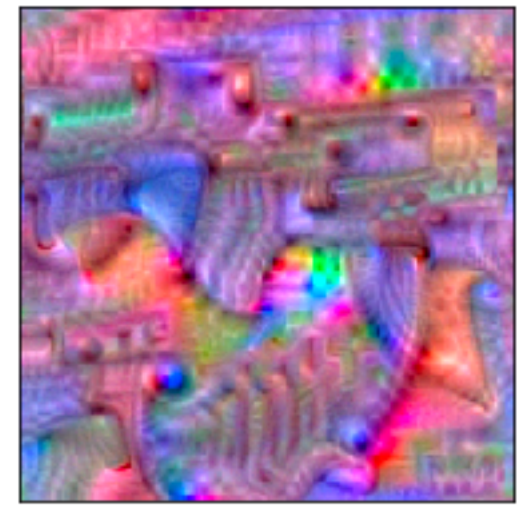
Goldfish



Bear

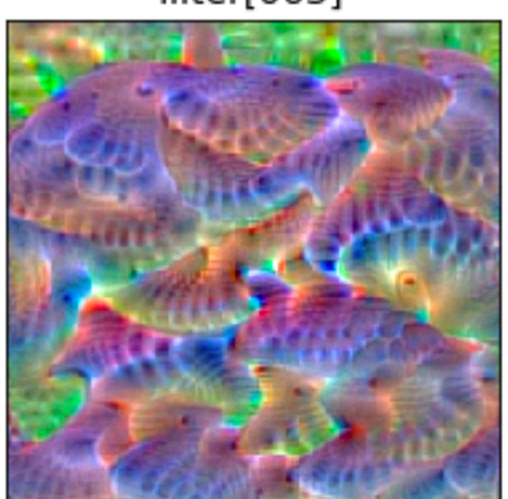


Assault rifle

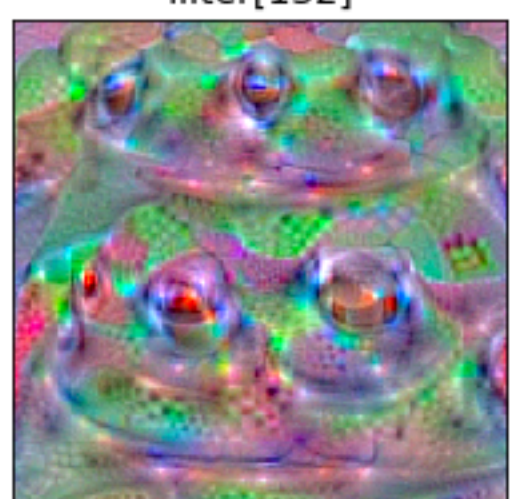


Visualized hidden convolutional layers

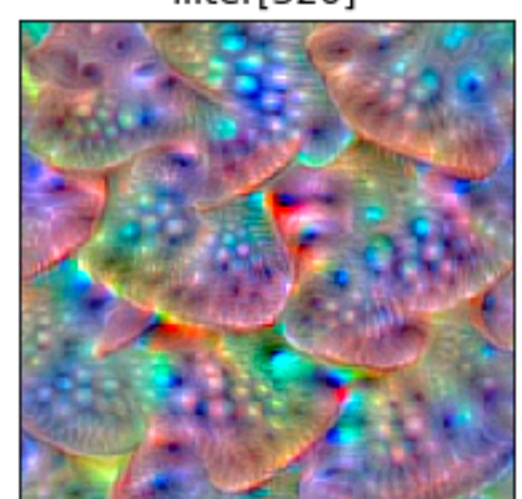
filter[063]



filter[132]



filter[320]

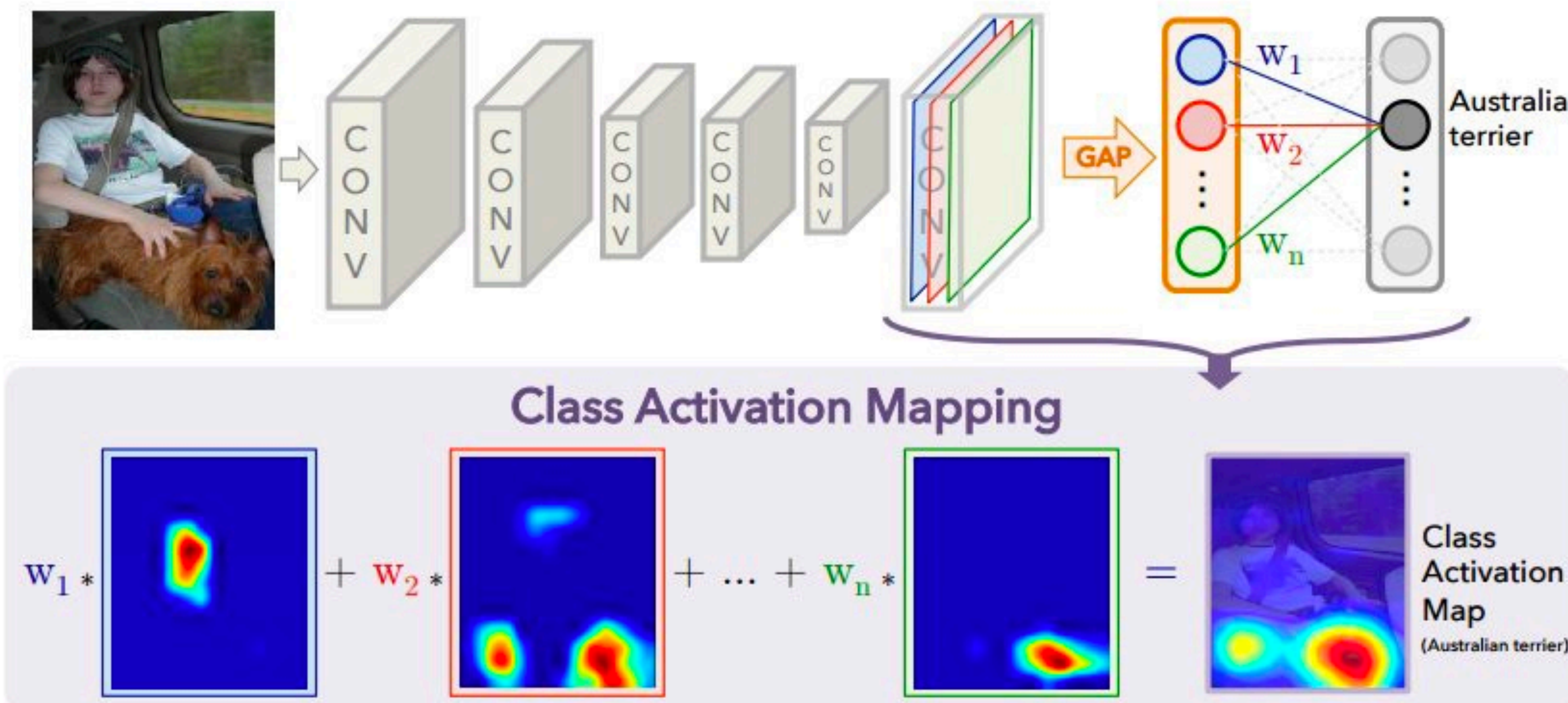


https://pypi.org/project/tf-keras-vis/

Grad-CAM heat maps

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

CAM heat maps

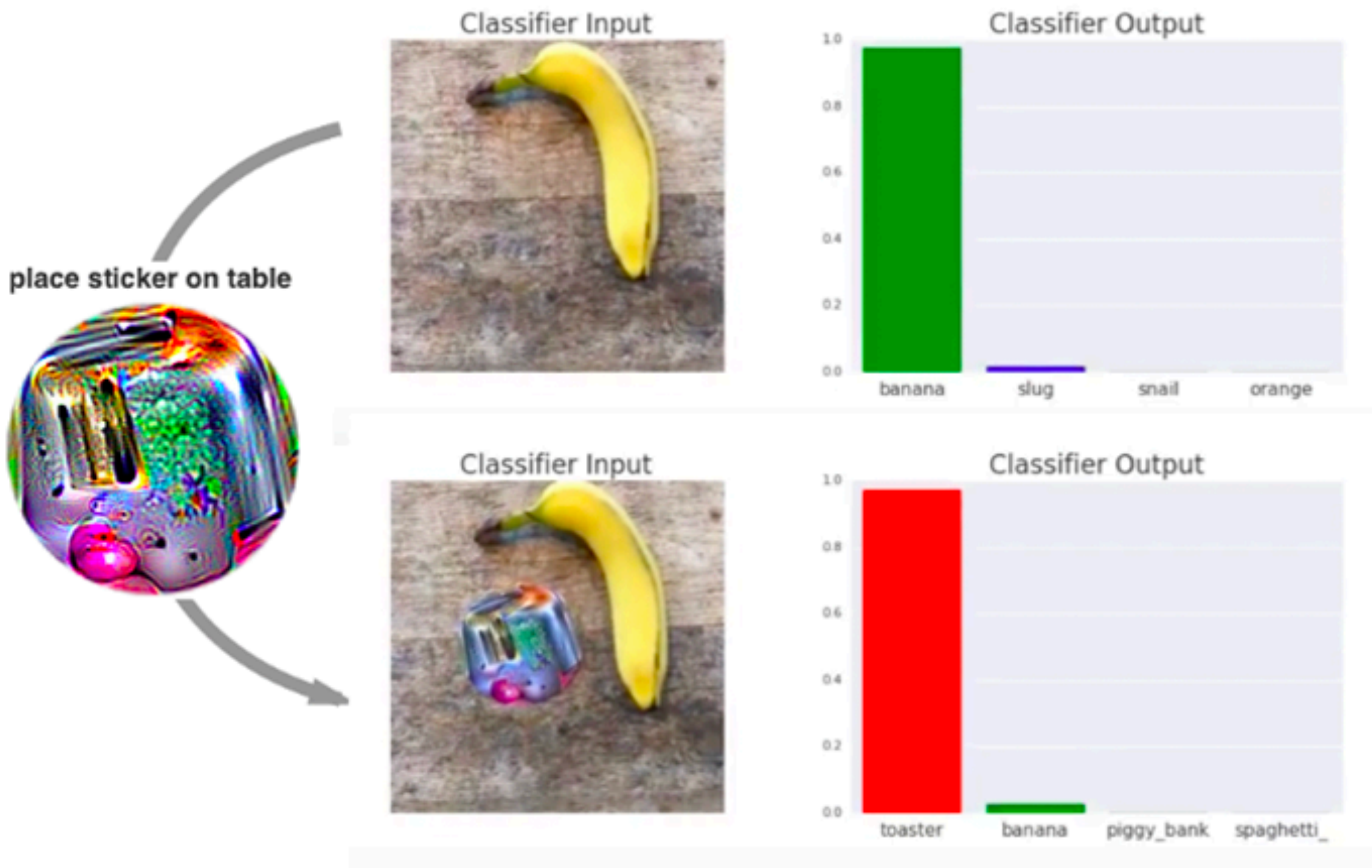


https://towardsdatascience.com/demystifying-convolutional-neural-networks-using-class-activation-maps-fe94eda4cef1

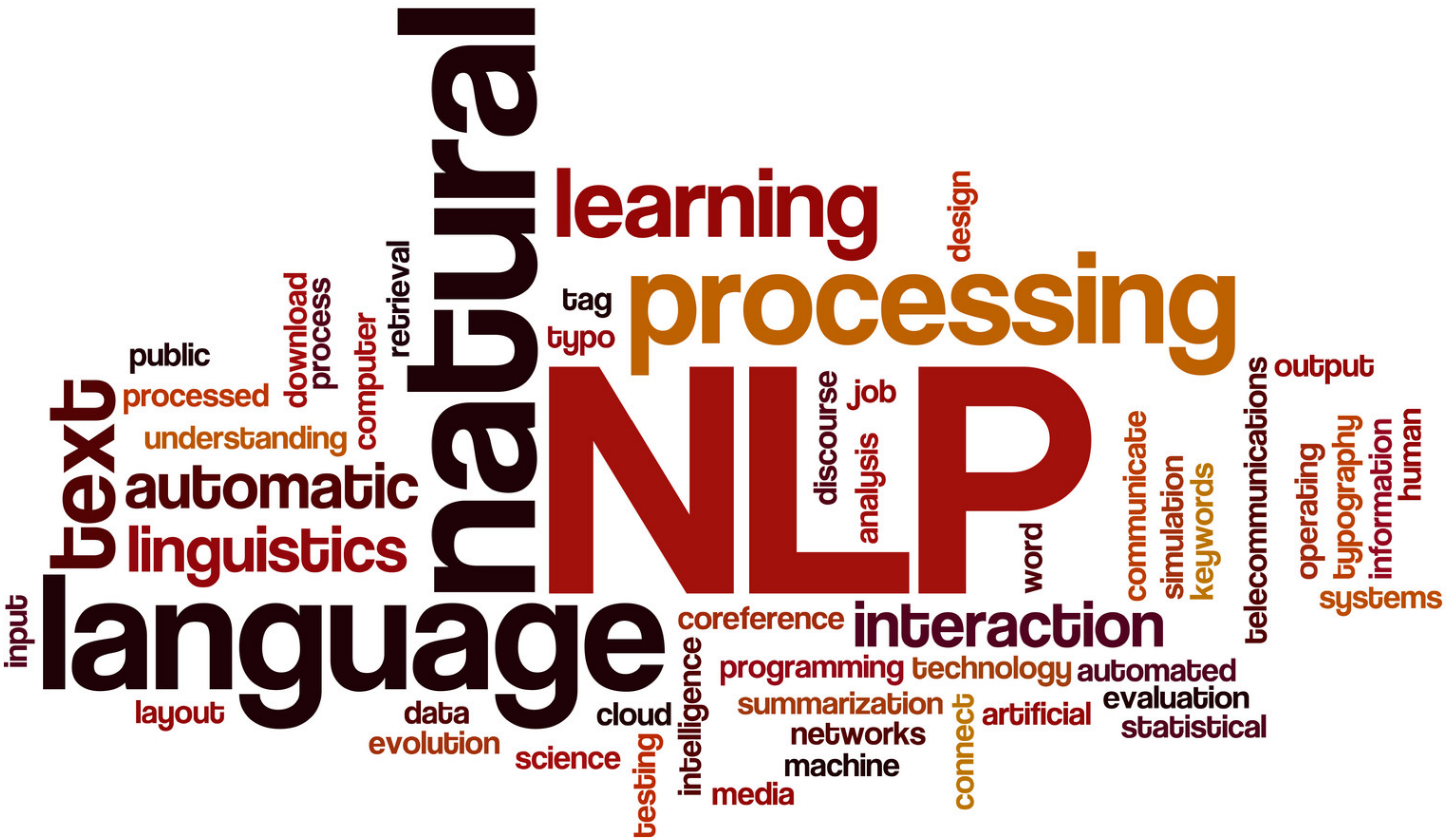
Convolutional Neural Networks Explainability in Keras

9-Explainability.ipynb

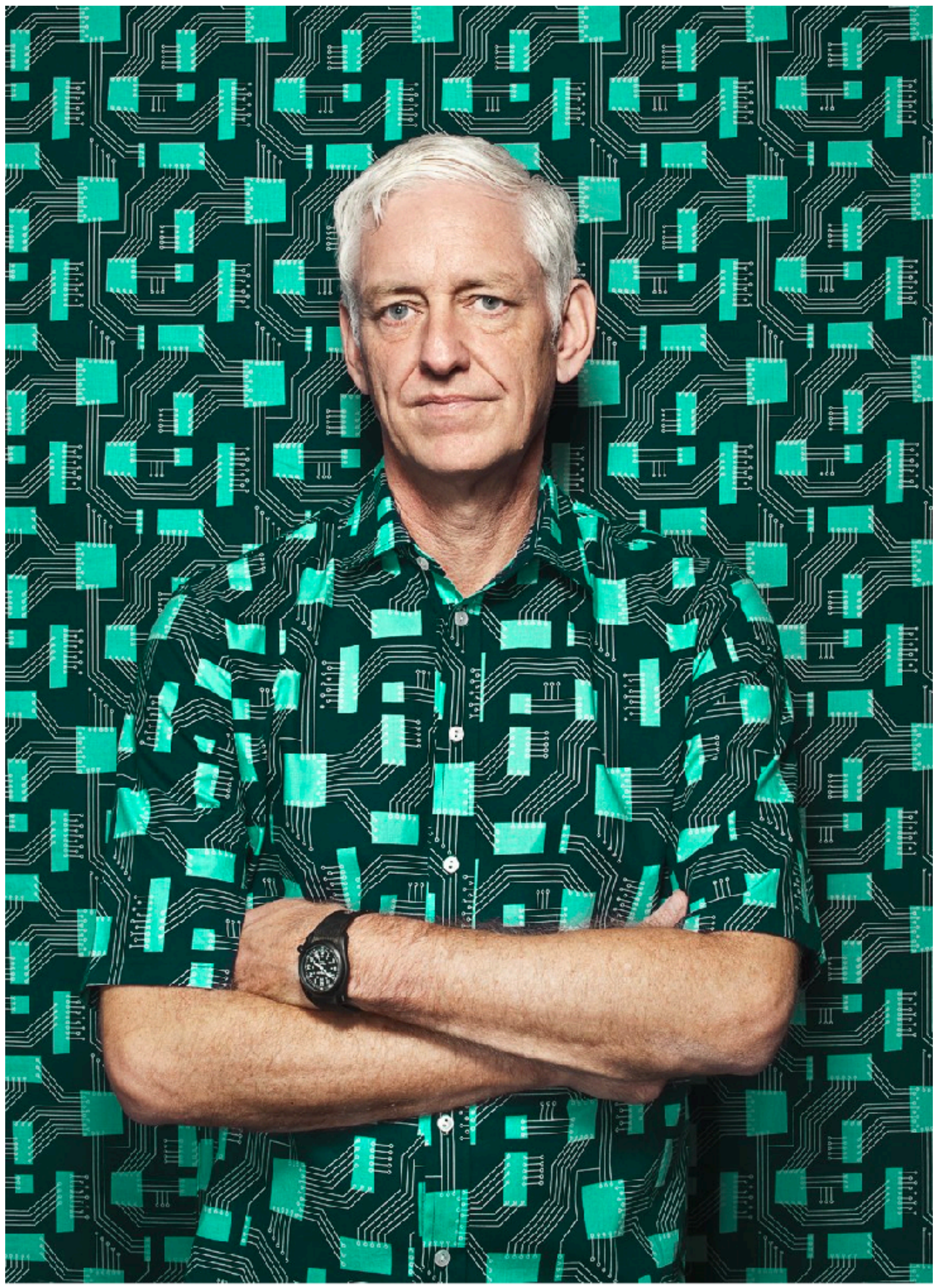
Adversarial Patch



What is Natural Language Processing?



Norvig vs. Chomsky



source: https://www.commarts.com

source: https://citaty.net

Text corpus

Sketch Engine | natural language | British National Corpus

- Home
- Search
- Word list
- Word sketch
- Thesaurus
- Sketch diff
- Trends
- Corpus info
- My jobs
- User guide

- Save
- Make subcorpus
- View options
  - KWIC
  - Sentence
- Sort
  - Left
  - Right
  - Node
  - References
  - Shuffle
- Sample

Query **natural, language** **255** (2.27 per million)

Page 1 of 13 Go Next | Last

J2K	nature of deixis (see Chapter 2 below) in	**natural languages**	, for sentences like (II) are true or false only
J2K	of the simple but immensely important fact that	**natural languages**	are primarily designed, so to speak, for use in
J2K	. The many facets of deixis are so pervasive in	**natural languages**	, and so deeply grammaticalized, that it is hard
J2K	the utterance, within the utterance itself.	**Natural language**	utterances are thus" anchored" directly to
J2K	semantics deals with certain	**natural language**	expressions. Suppose we identify the semantic
J2K	or self-referring expressions in	**natural languages**	, as in (12) and, arguably, in (13) (see Chapter 5
J2K	, is perhaps a philosophical red-herring.	**Natural languages**	, after all, just do have indexicals, and it is
J2K	. Semantics is then not concerned directly with	**natural language**	at all, but only with the abstract entities
J2K	to leave us with no term for all those aspects of	**natural language**	significance that are not in any way amenable to
J2K	of the deictic expressions that occur in	**natural languages**	, and we should now turn to consider linguistic
J2K	in familiar languages. Deictic systems in	**natural languages**	are not arbitrarily organized around the
J2K	. But this has the consequence, as we noted, that	**natural languages**	will only have a syntax and a pragmatics, and no
J2K	more or less directly on fragments of	**natural language**	(as initiated by Montague, 1974) would make
J2K	The semanticist who takes the other tack, that	**natural language**	senses are protean, sloppy and variable, is
J2K	offers a way out, for it allows one to claim that	**natural language**	expressions do tend to have simple, stable and
J2K	radical differences between logic and	**natural language**	seem to fade away. We shall explore this below
J2K	on what can be a possible lexical item in	**natural languages**	. Finally, the principles that generate
J0V	is meant any single document, or any stretch of	**natural language**	regarded as a self-contained unit for
J53	recognition and those that can understand	**natural languages**	, such as English, are known by the collective
HRK	through a dialogue, which approaches a	**natural language**	dialogue, or via a menu. In figure 6.2, the users

Page 1 of 13 Go Next | Last

Token & tokenization

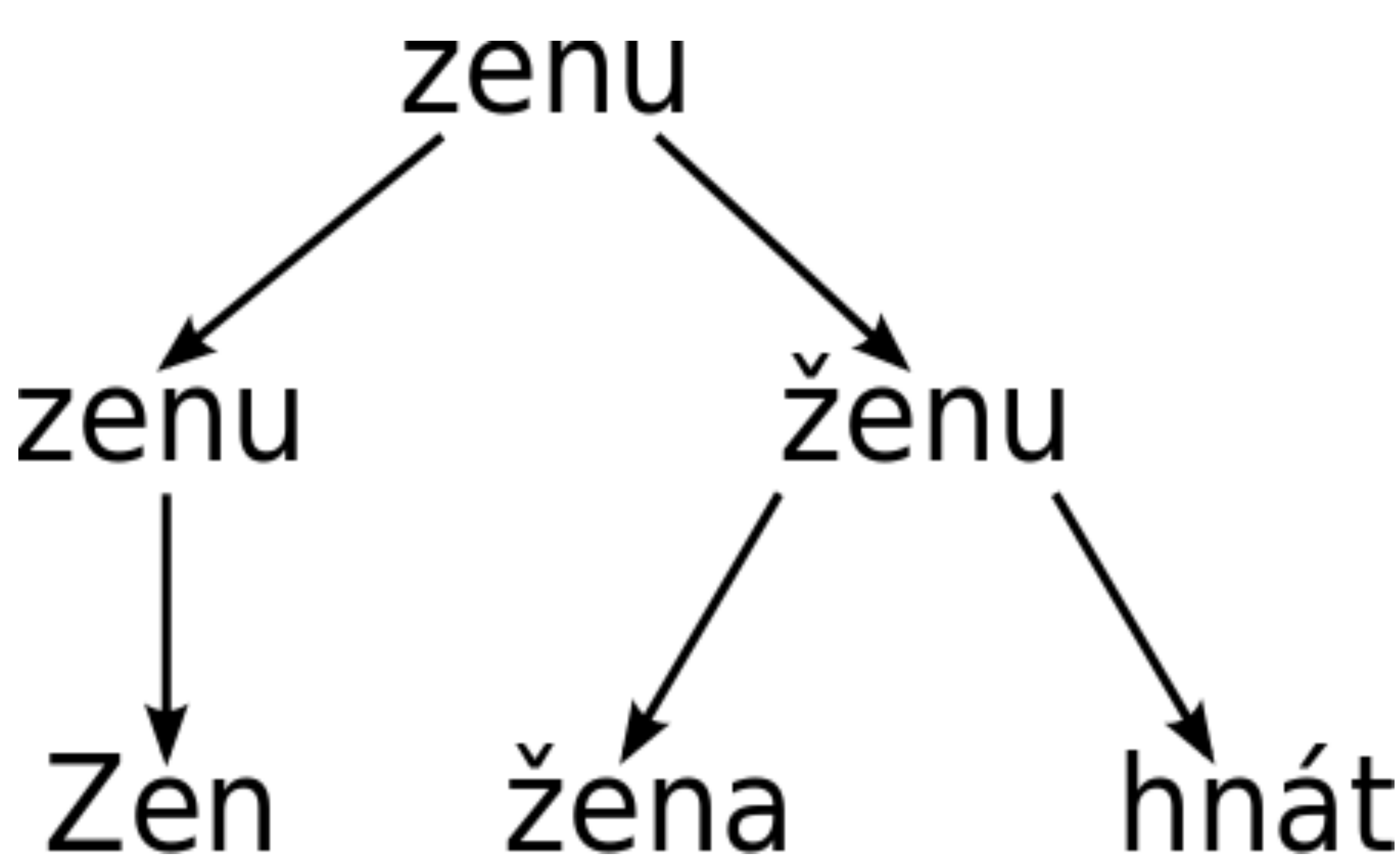
This is a non-trivial English sentence: Ludolph’s number is approx. 3.14.

Python library: http://www.nltk.org/

Stemming & lemmatization

Original	Stemming	Lemmatization
compensation	compens	compensation
compensations	compens	compensation
mouse	mous	mouse
mice	mice	mouse

Ambiguity in lemmatization



Stemming & lemmatization

English:

https://tartarus.org/martin/PorterStemmer/

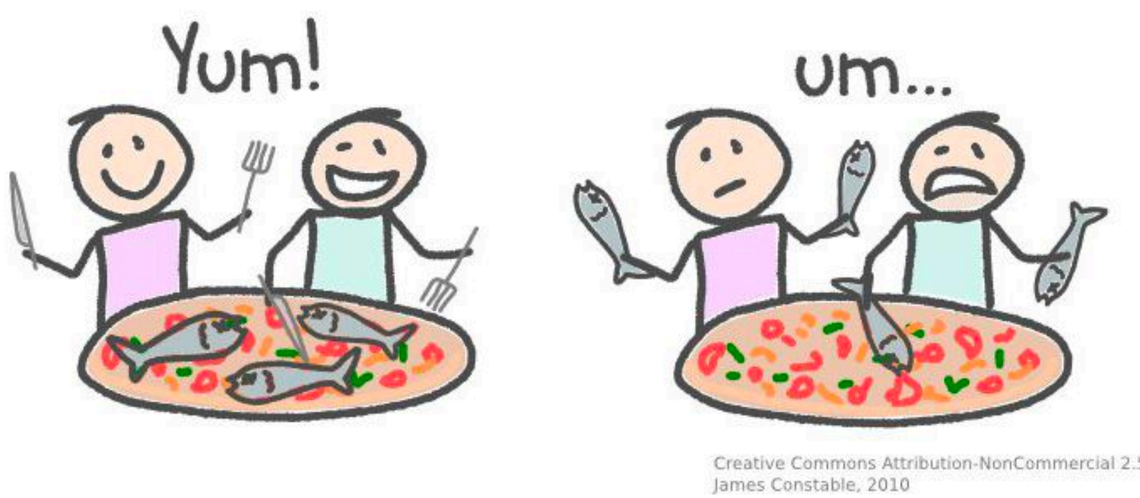
http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

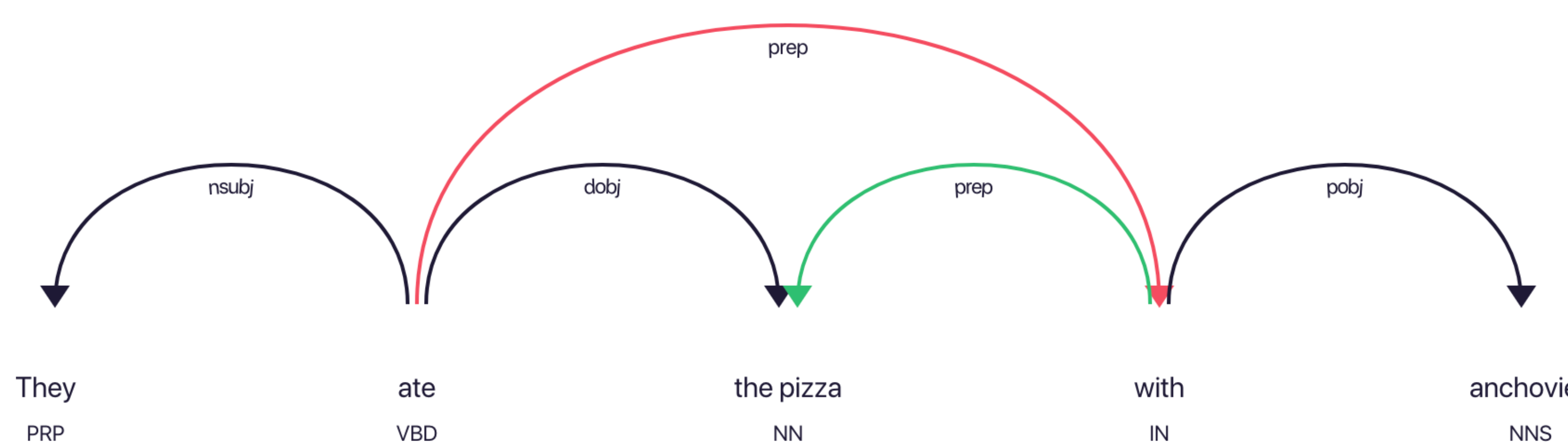
Czech:

http://ufal.mff.cuni.cz/morphodita

Parsing

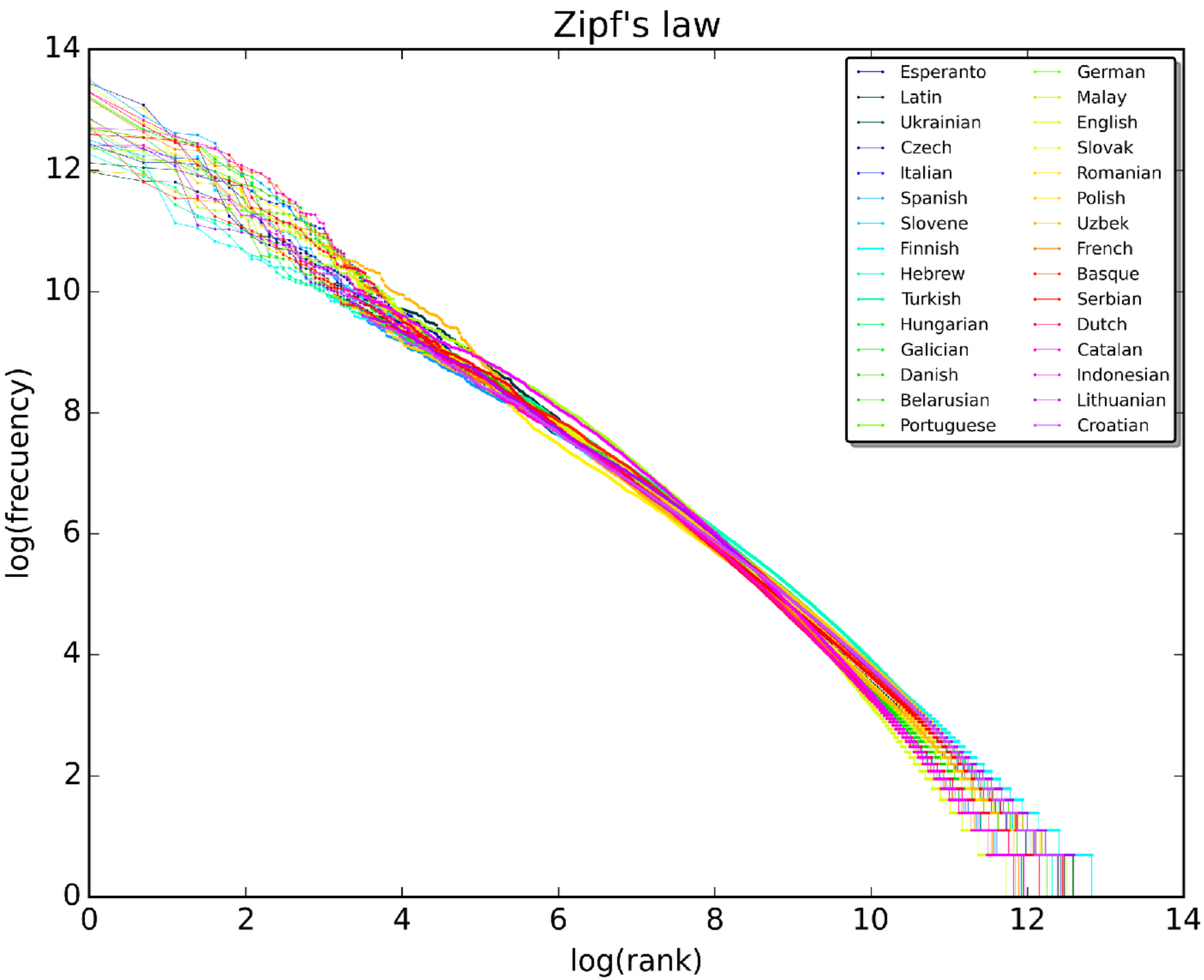
`They ate the pizza with anchovies`





source: https://explosion.ai/blog/syntaxnet-in-context

Zipf’s law & long tail



Publicly available corpora

British National Corpus: http://www.natcorp.ox.ac.uk/

Common Crawl: http://commoncrawl.org/the-data/get-started/

Wikipedia: https://dumps.wikimedia.org/

Feature extraction for NLP

1. *the man walked the dog*
2. *the man took the dog to the park*
3. *the dog went to the park*

[dog, man, park, the, to, took, walked, went]

1. [1, 1, 0, 1, 0, 0, 1, 0]
2. [1, 1, 1, 1, 1, 1, 0, 0]
3. [1, 0, 1, 1, 1, 0, 0, 1]

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

1. [1, 1, 0, 2, 0, 0, 1, 0]
2. [1, 1, 1, 3, 1, 1, 0, 0]
3. [1, 0, 1, 2, 1, 0, 0, 1]

1. [0, 0.18, 0, 0, 0, 0, 0.48, 0]
2. [0, 0.18, 0.18, 0, 0.18, 0.48, 0, 0]
3. [0, 0, 0.18, 0, 0.18, 0, 0, 0.48]

NLP Introduction task

10-text-classification-introduction.ipynb

Language models

- spell checking
- speech recognition
- machine translation
- …

n-gram models

$$P(w_1, w_2, \ldots w_n) = P(w_1)P(w_2|w_1)\ldots P(w_n|w_1, \ldots w_{n-1})$$

$$= \prod_i P(w_i|w_1, w_2 \ldots w_{i-1})$$

$$\approx \prod_i P(w_i|w_{i-k}, w_{i-k-1} \ldots w_{i-1})$$

$$P(w_i|w_{i-k}, w_{i-k-1} \ldots w_{i-1}) = \frac{count(w_{i-k}, w_{i-k-1} \ldots w_{i-1}, w_i)}{count(w_{i-k}, w_{i-k-1} \ldots w_{i-1})}$$

Language model smoothing

- Laplace smoothing (plus one)

$$P(w_i|w_{i-1}) = \frac{count(w_{i-1}, w_i) + 1}{count(w_{i-1}) + V}$$

- interpolation
- Good-Turing
- Witten-Bell
- …

Perplexity

$$PP(W) = P(w_1, w_2, \ldots, w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1, w_2, \ldots, w_N)}}$$

$$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_1, \ldots, w_{i-1})}}$$

$$= 2^{-\frac{1}{N} \sum_{i=1}^{N} log_2 P(w_i | w_1, \ldots, w_{i-1})}$$

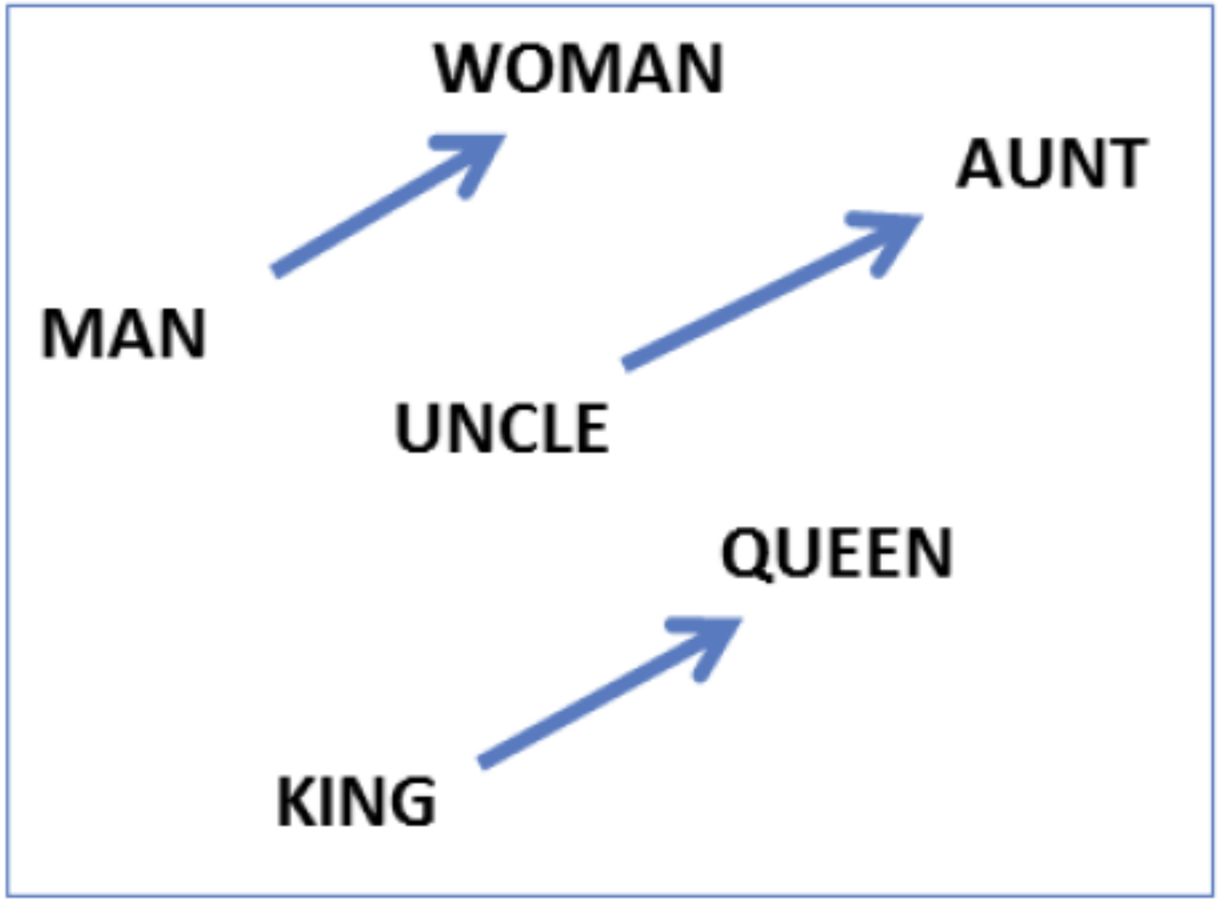
Language detection using language models

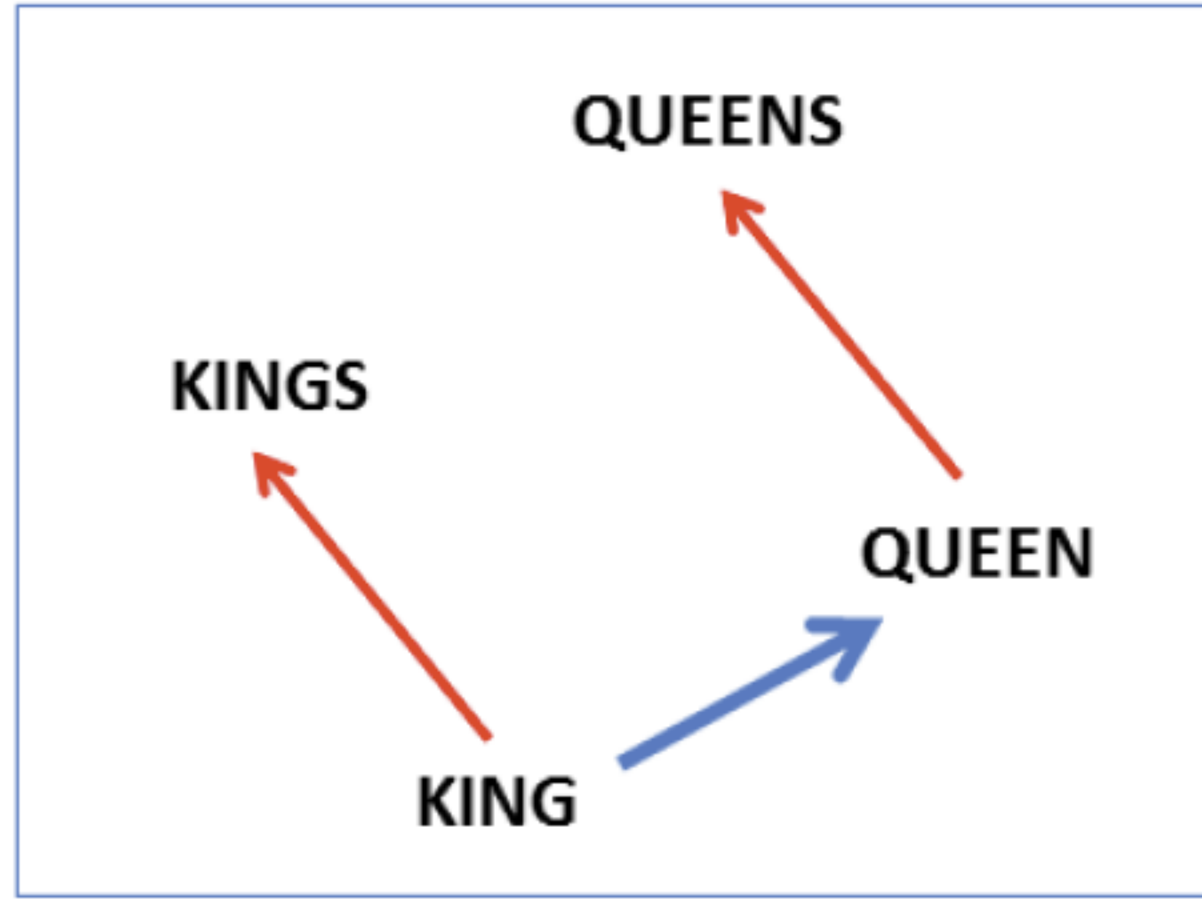
11-Language-detection-assignment.ipynb

Travel agency review classification

12-Review-classification-assignment.ipynb

word2vec

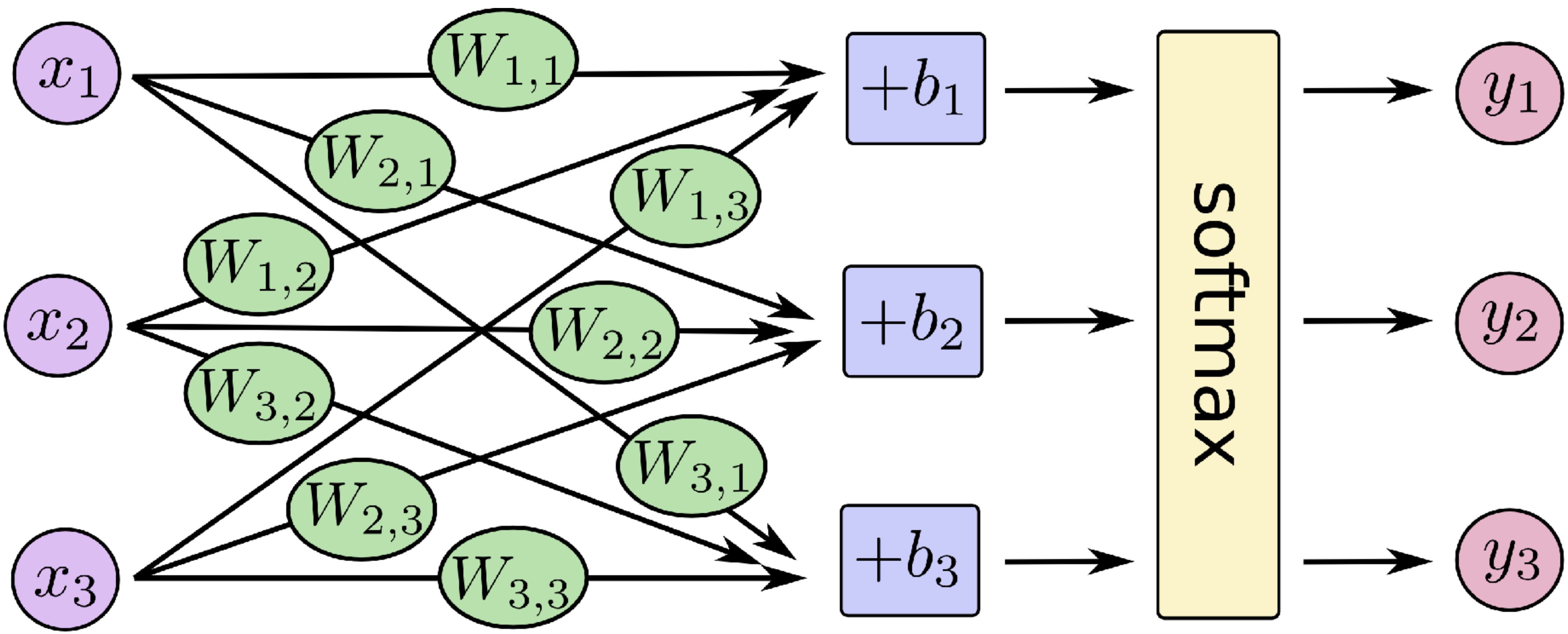




king is to **kings** as **queen** to ***?***.

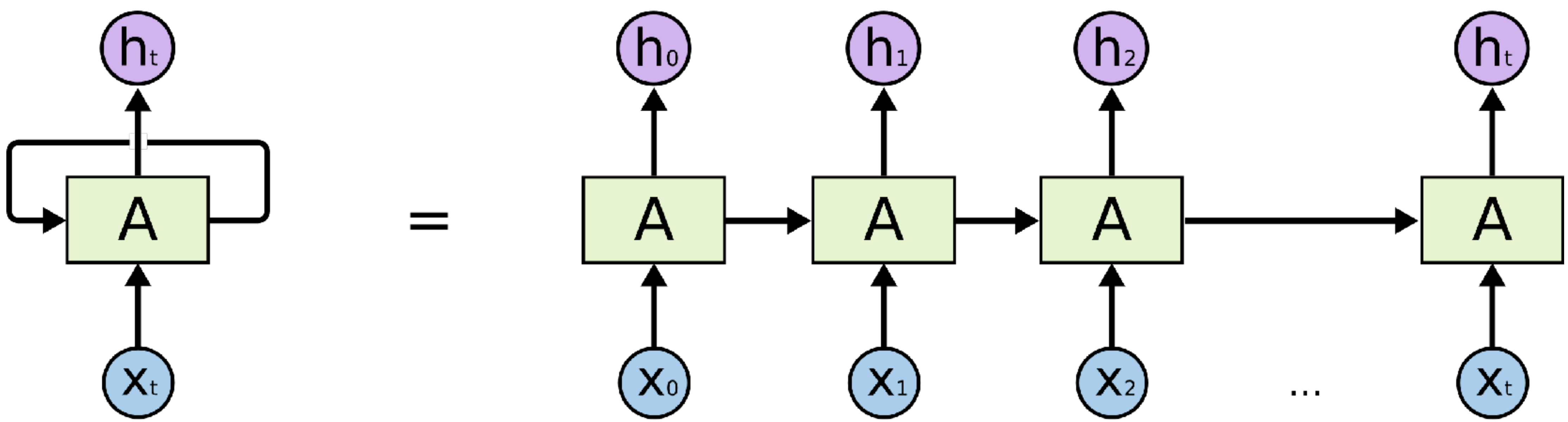
v(**kings**) - v(**king**) = v(**queens**) - v (**queen**)

Feed-Forward Neural Network



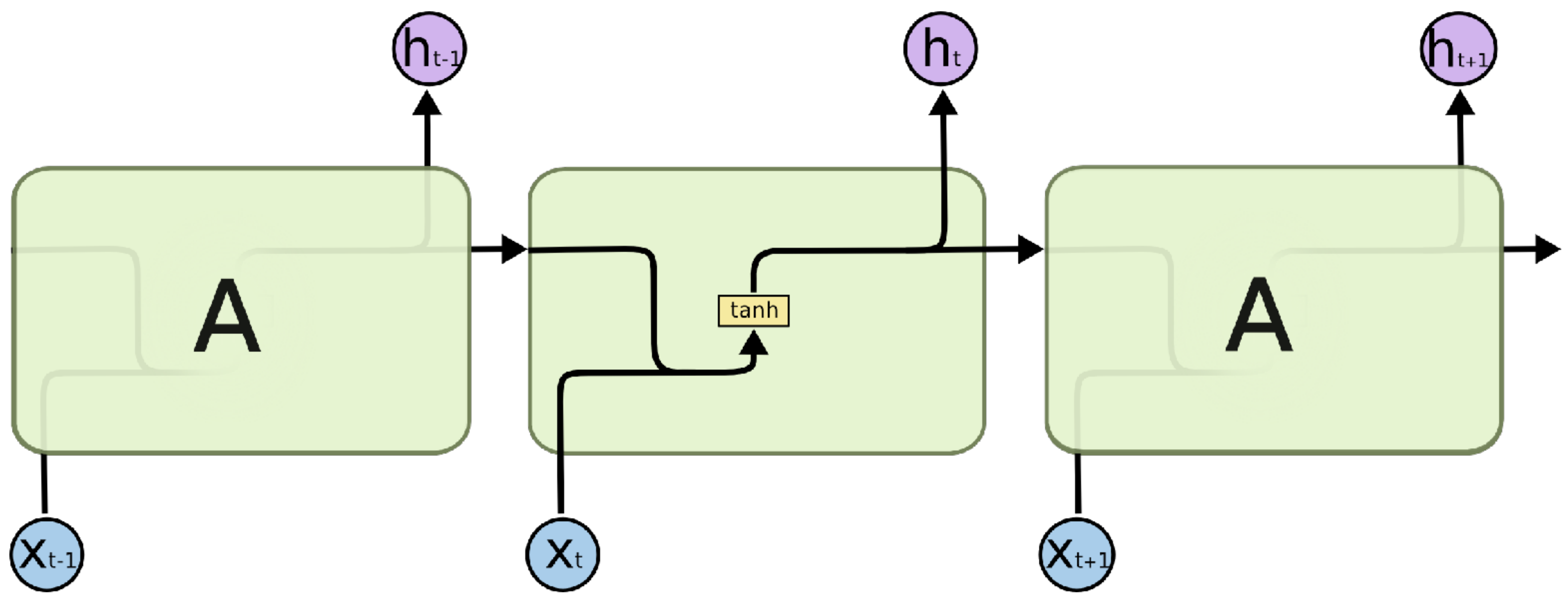
source: https://www.tensorflow.org

Recurrent Neural networks 1/2



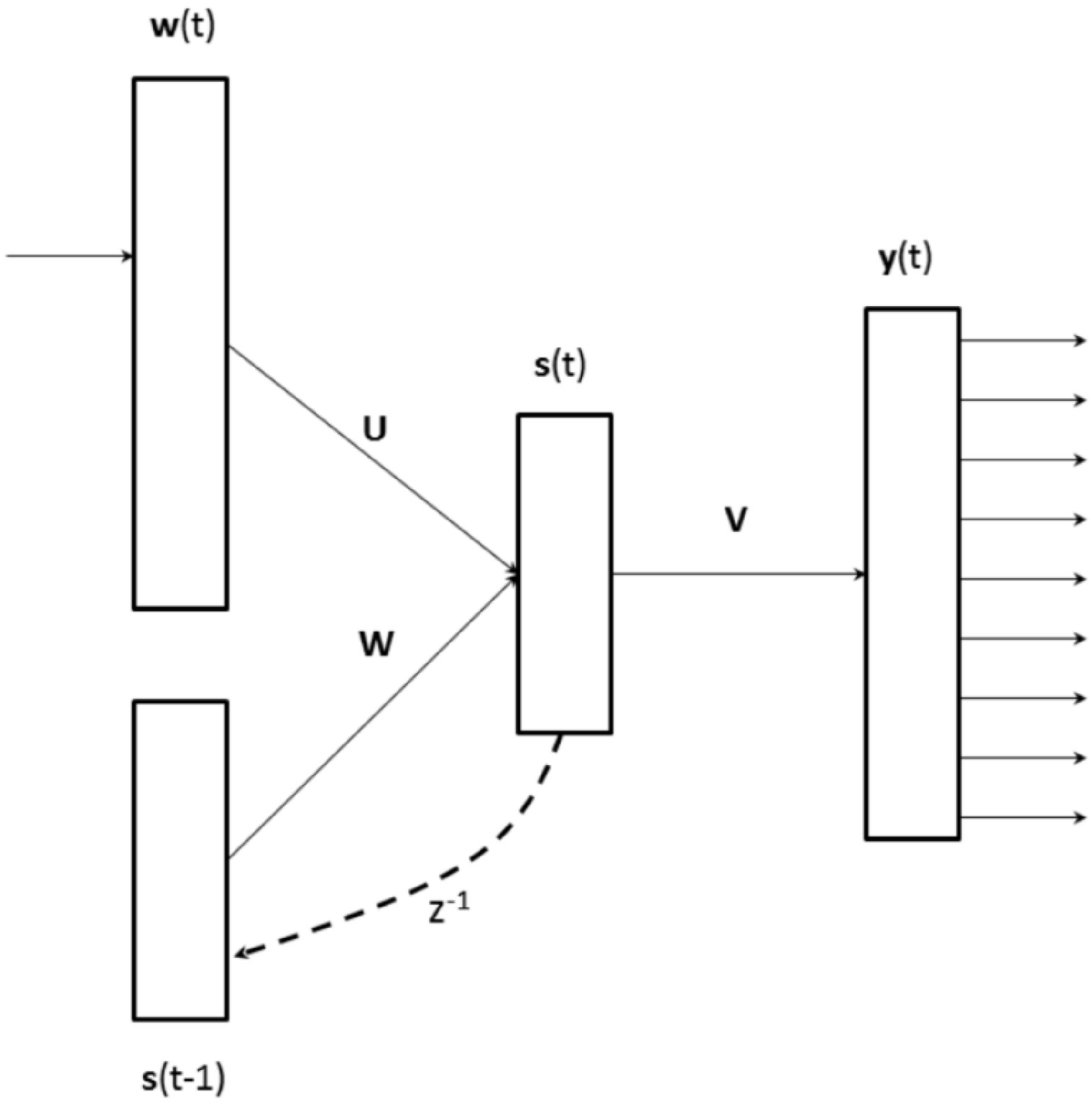
source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent Neural Networks 2/2



https://www.tensorflow.org

Recurrent Neural Network Language Modeling Toolkit



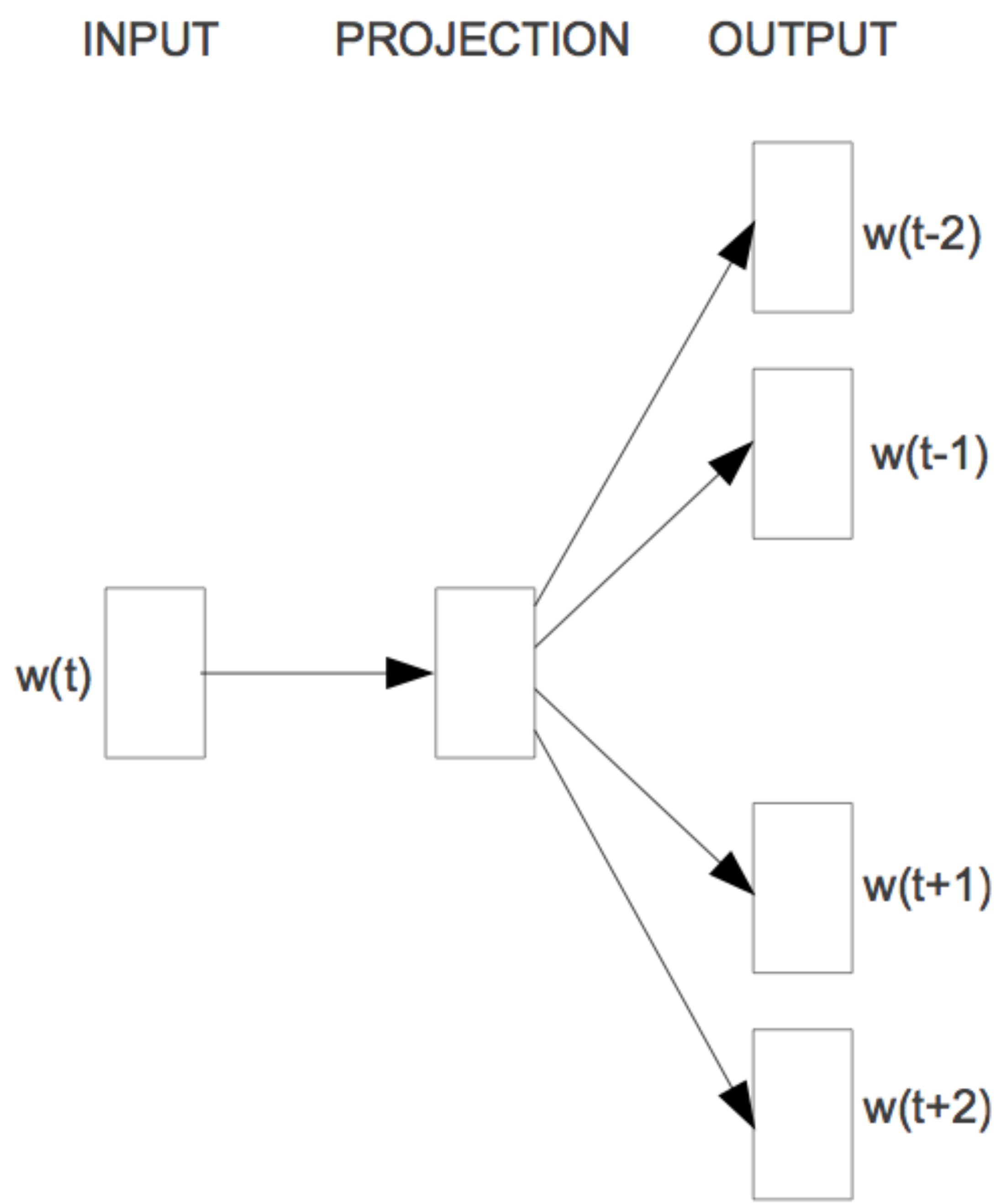
$$\mathbf{s}(t) = f\left(\mathbf{U}\mathbf{w}(t) + \mathbf{W}\mathbf{s}(t-1)\right)$$

$$\mathbf{y}(t) = g\left(\mathbf{V}\mathbf{s}(t)\right),$$

$$f(z) = \frac{1}{1+e^{-z}}, \quad g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}.$$

source: http://www.fit.vutbr.cz/~imikolov/rnnlm/

The skip-gram model



Experiments with word2vec

13-Word2vec-in-gensim.ipynb

14-Review-classification-w2v-assignment.ipynb

Language models for text generating

Nacházíte se: Úvod › Oddělení › Krásná literatura › Poezie › Česká a slovenská poezie › **Elektronická kniha Poezie umělého světa**

Ukázka pdf

Ukázka epub

Ukázka mobi

Like 180

Poezie umělého světa [E-kniha]

Jiří Materna

★★★★★ Hodnotilo 7 uživatelů, zatím žádné recenze, napsat vlastní recenzi

Popis: Elektronická kniha, 50 stran, bez zabezpečení DRM, ePUB, Mobi, PDF, česky - více

Stáhnout

Zdarma
K dispozici pro **okamžité** stáhnutí

Ke stažení

Anotace

Všechny básně v této knize byly automaticky vygenerovány počítačem za pomoci umělých neuronových sítích. Neuronová síť sama o sobě nic neumí a je třeba ji natrénovat pro činnost, kterou má vykonávat.

LISTOPAD

usínám, pláču, umírám, přemýšlím
co cítíš ty?
cítím tvou slabost
a whisky

NOVEMBER

I am falling asleep, crying, dying, thinking
what do you feel?
I feel your weakness
and whisky

SPRAVEDLNOST

na tvou dekadentní duši
ráno i v poledne
bůh má připravenou kuši

JUSTICE

for your decadent soul
in the morning, in the evening
the god has prepared a crossbow

Metaphores

...tělo plné červánků...

...body full of blush of dawn...

...tak vzácný jako listí...

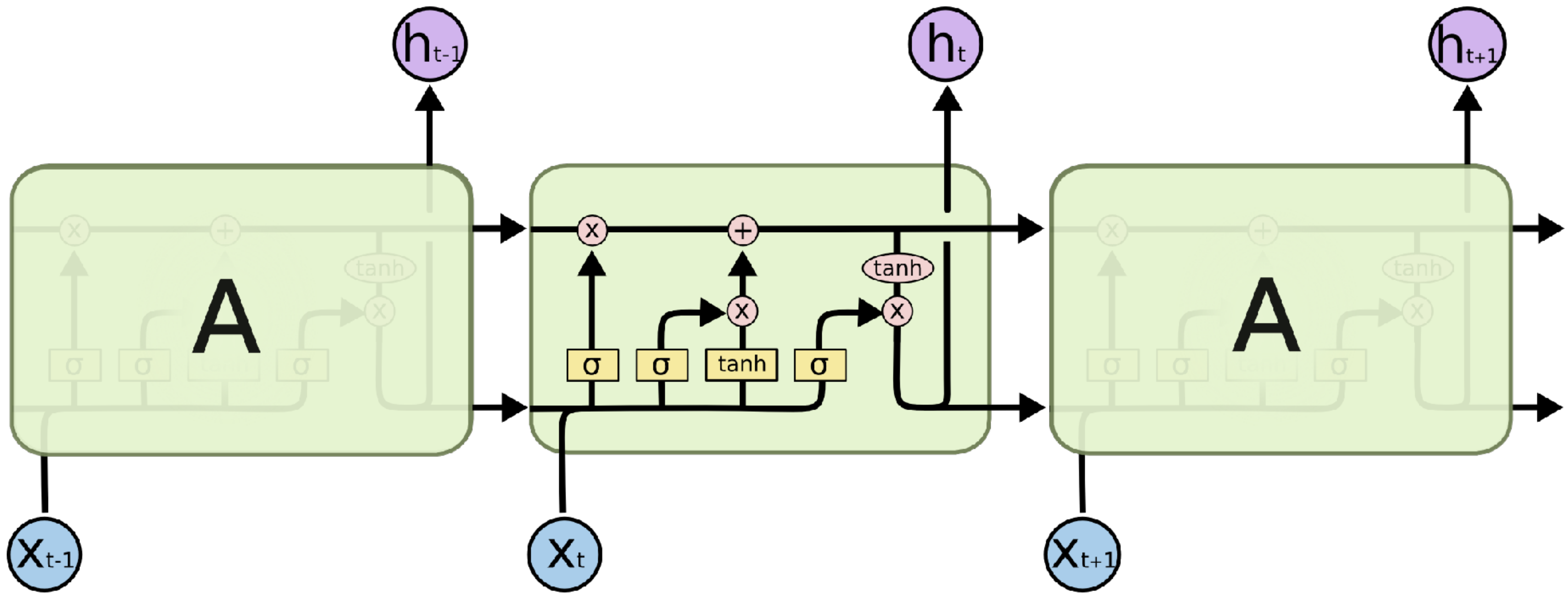
...as rare as leaves of trees...

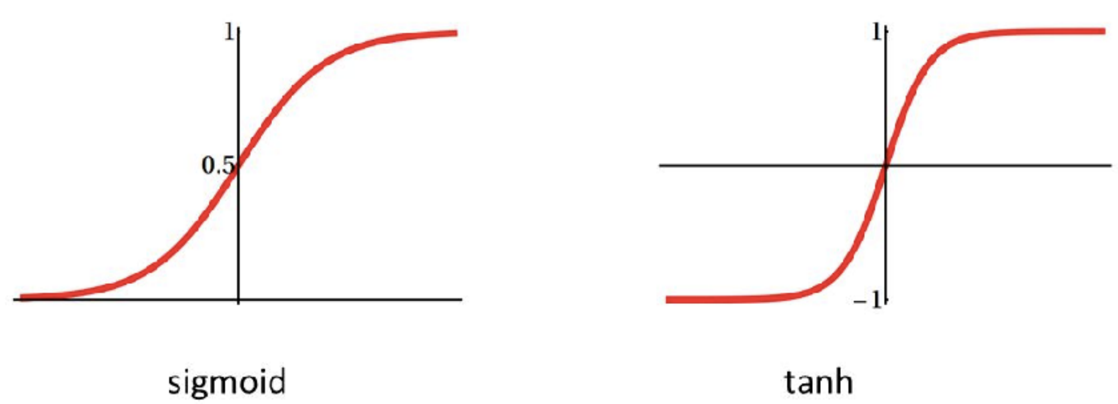
Language models for text generating

P(maso | máma, mele) = 0.5
P(Emu | máma, mele) = 0.3
P(tátu | máma, mele) = 0.2

```
t ~ Uniform(0, 1)
s = 0
for v in Vocabulary:
   s += v.prob
   if t < s:
      return v.word
```

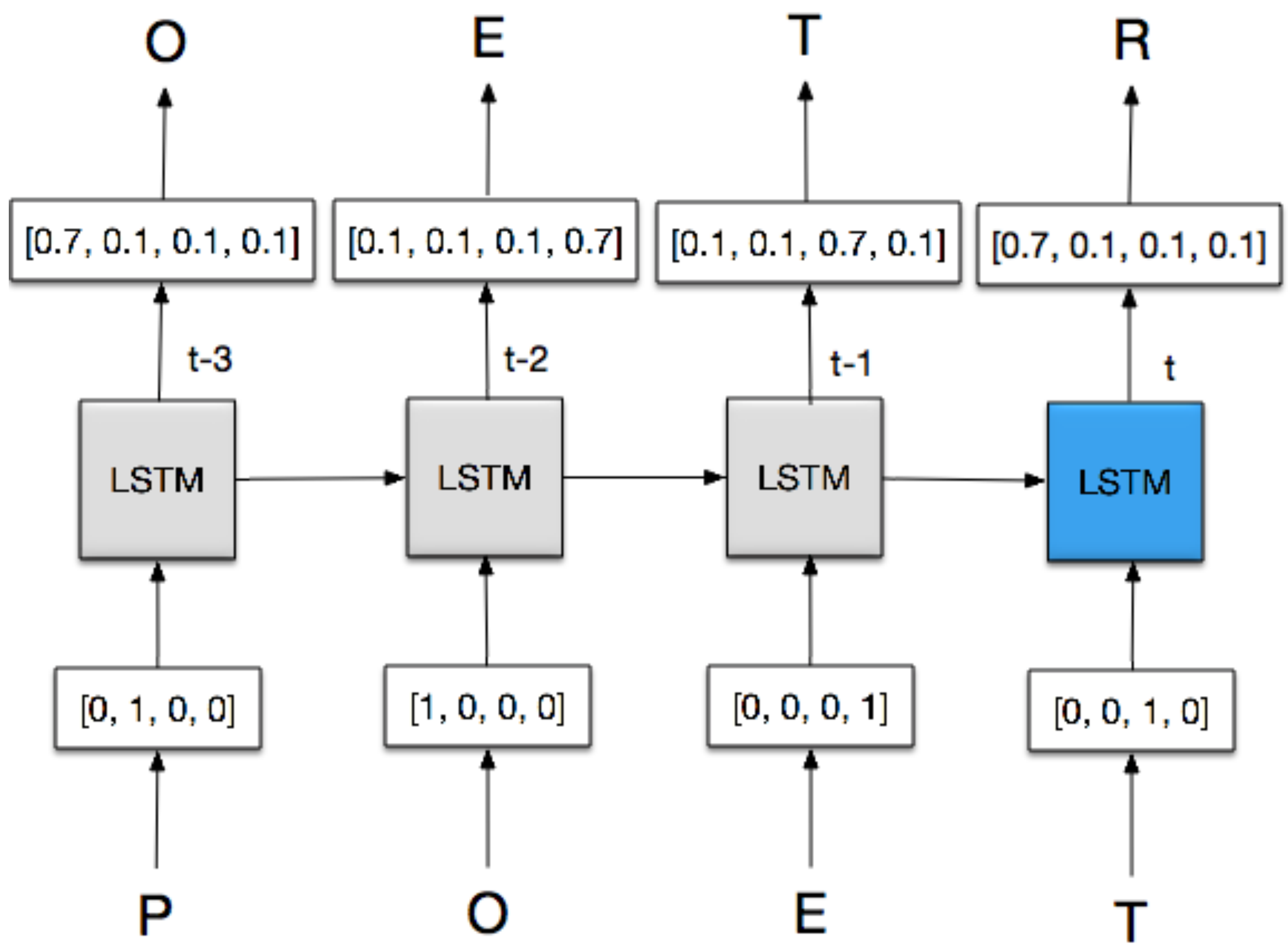
Long Short-Term Memory

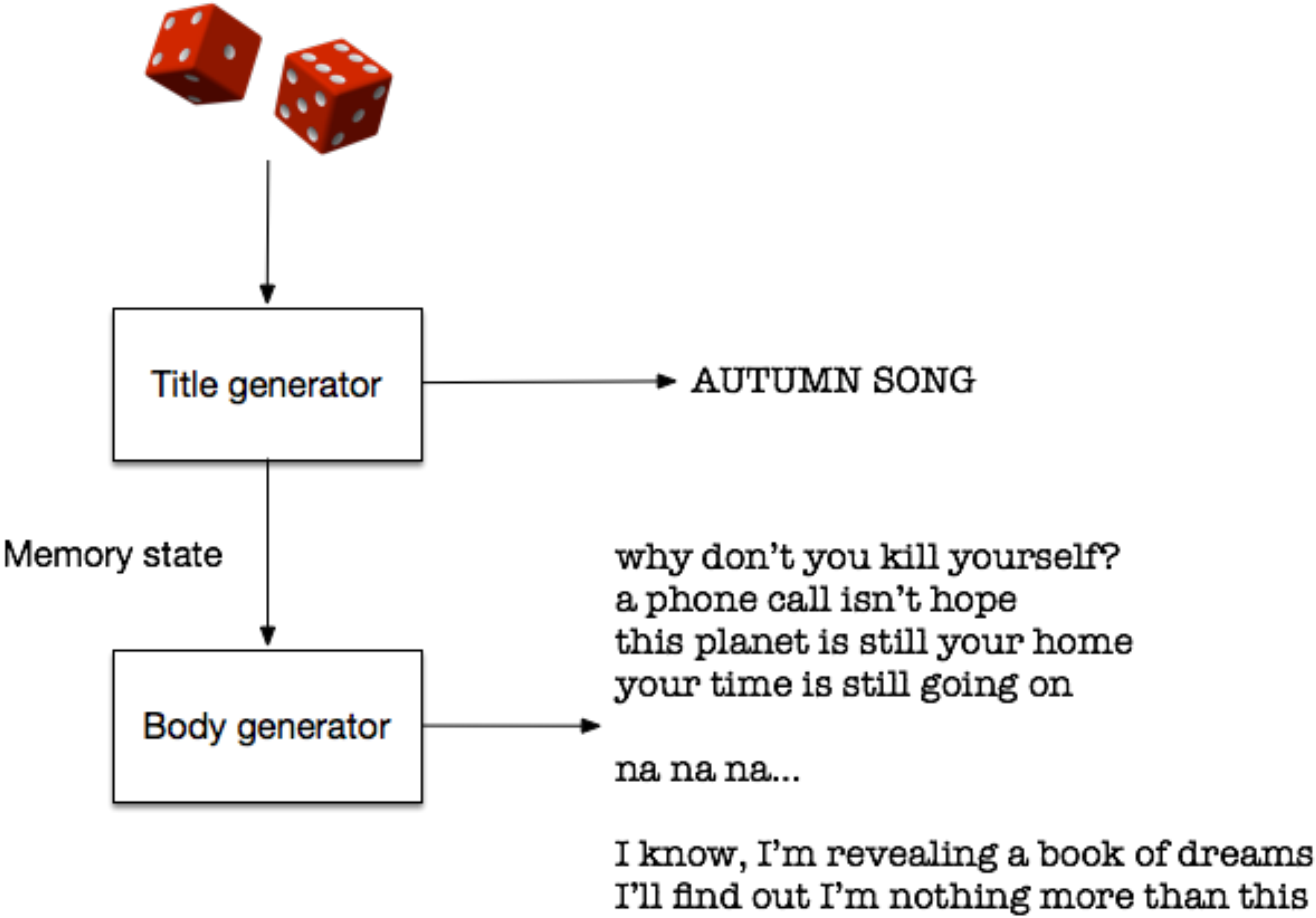




Zdroj: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM language model




Title generator
AUTUMN SONG
Memory state
Body generator
why don’t you kill yourself?
a phone call isn’t hope
this planet is still your home
your time is still going on

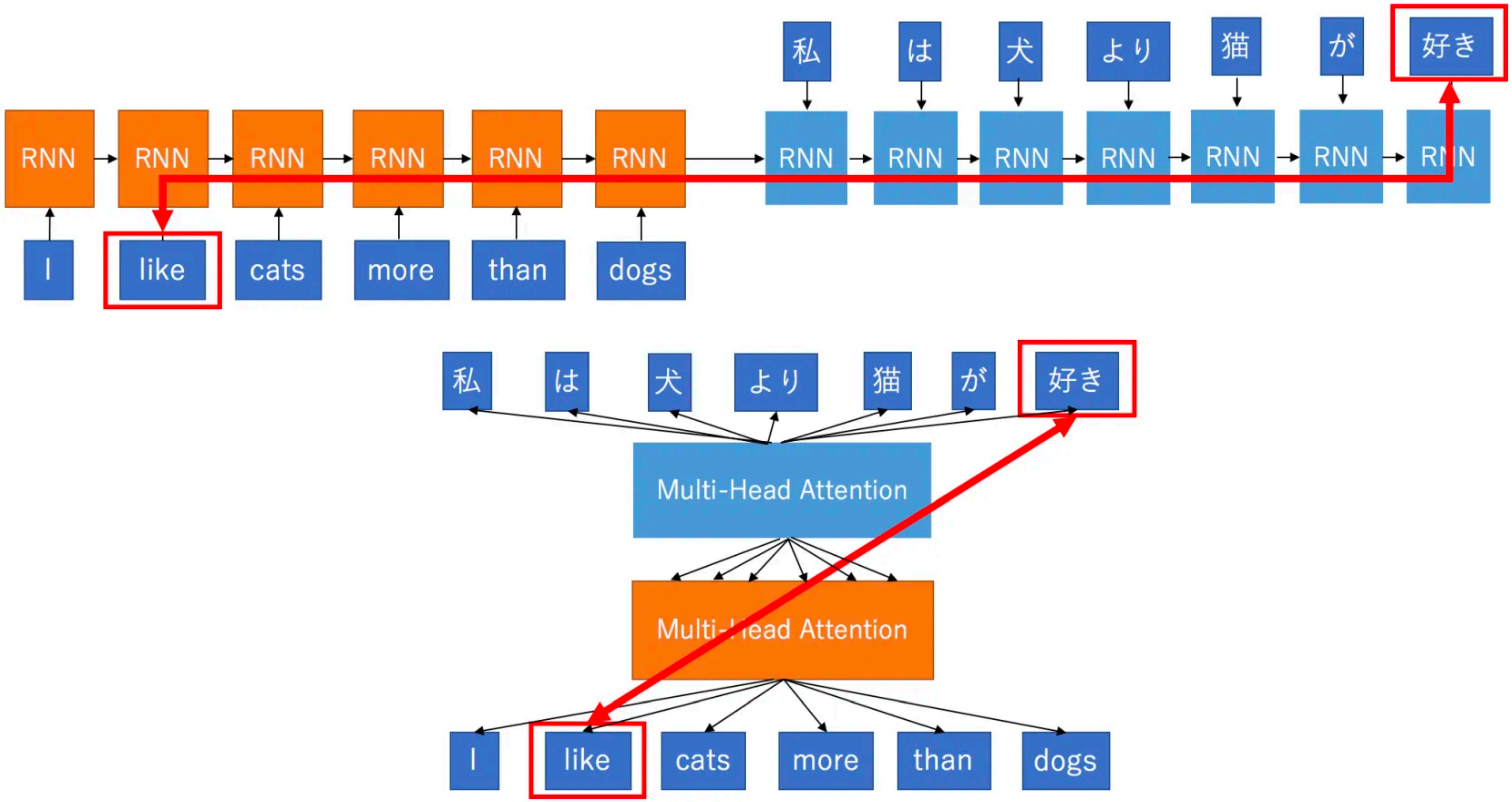
na na na...

I know, I’m revealing a book of dreams
I’ll find out I’m nothing more than this

LSTM review generator

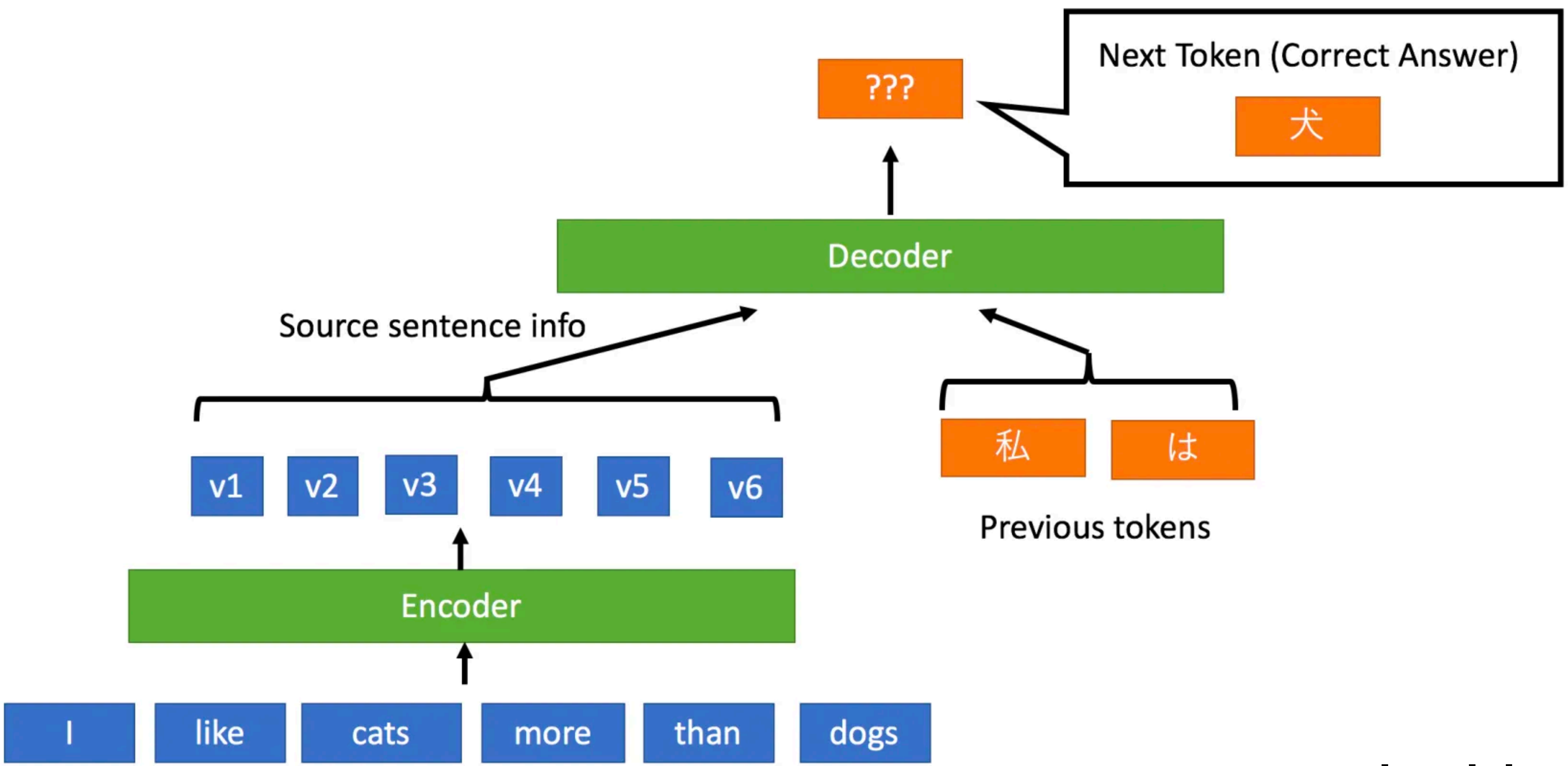
15-Review-generator.ipynb

Transformer



source: www.mlexplained.com

Translation with Transformers



www.mlexplained.com

GPT-2 Language model

Donald Trump told…

Demo: https://talktotransformer.com/

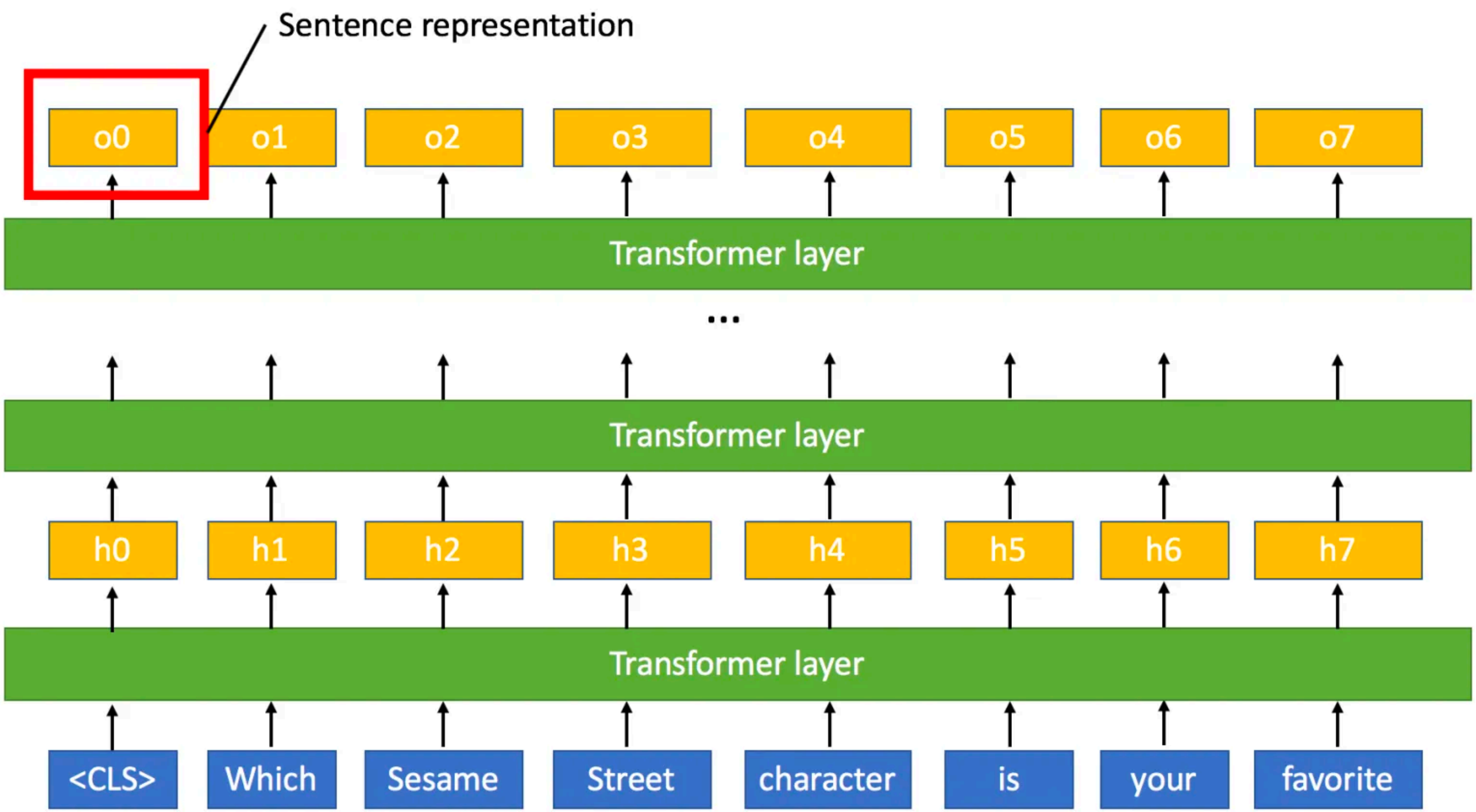
GPT-2 Language model

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.

Demo: https://talktotransformer.com/

BERT (classification)



What next?

https://www.mlcollege.com/en/#courses

Machine Learning Prague

ML MU MACHINE LEARNING meetups

Thank you for your attention

e-mail: jiri@mlcollege.com

Web: www.mlcollege.com

Twitter: @JiriMaterna

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

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