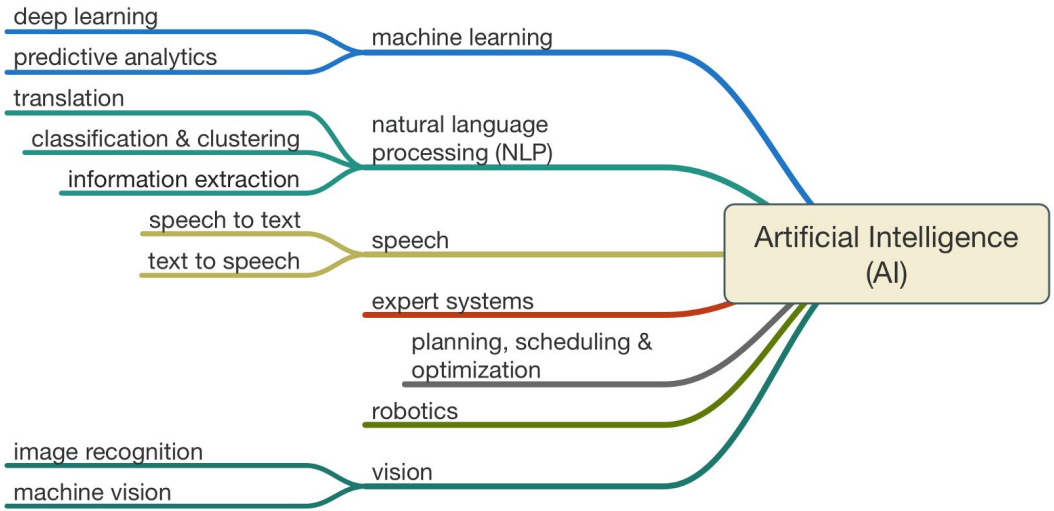
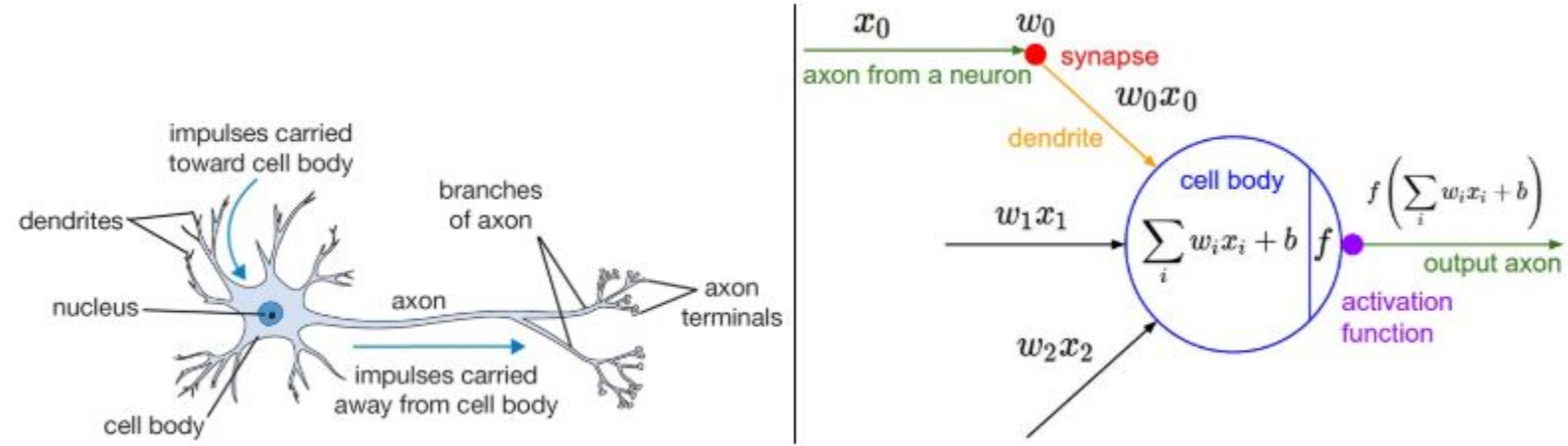
Deep Learning: Foundations and Practicing

Introduction to Deep Learning

- One of the threads of IA and Machine Learning
- Automatic feature extractor

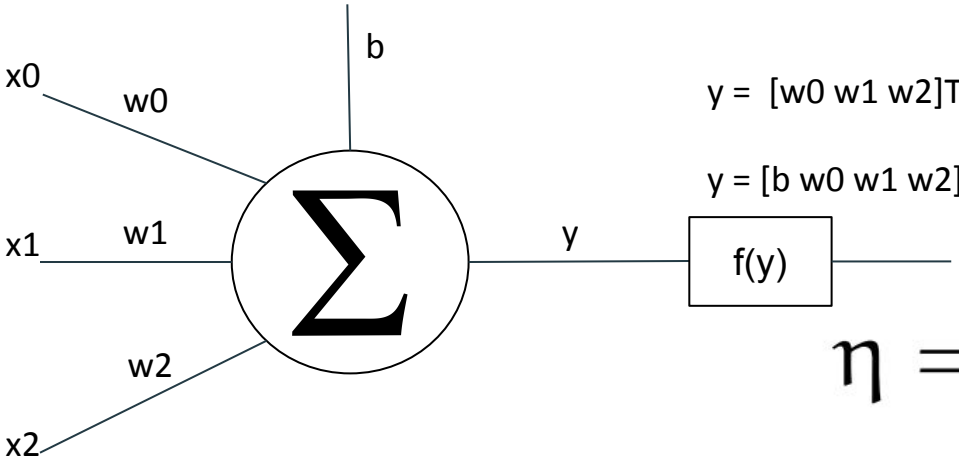


Perceptron (NN's basic unit)



A cartoon drawing of a biological neuron (left) and its mathematical model (right).

Neuron

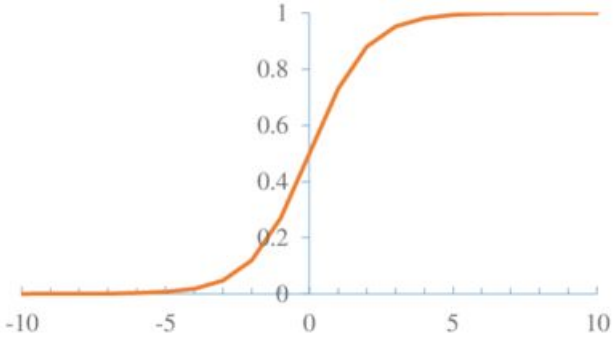


y = [w0 w1 w2]T X [x0 x1 x2] + b

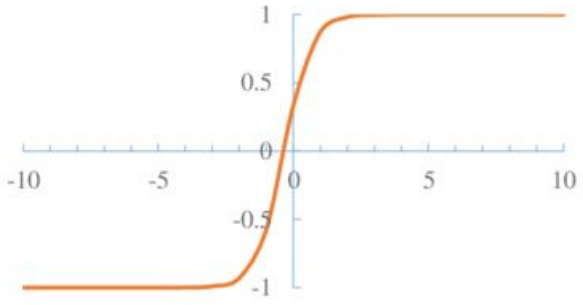
y = [b w0 w1 w2]T X [1 x0 x1 x2]

$$\eta = \beta^\top x + \beta_0$$

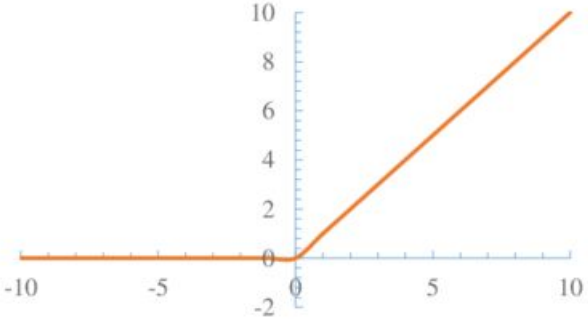
$$y = \eta + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)$$


1
0.8
0.6
0.4
0.2
0
-10
-5
0
5
10

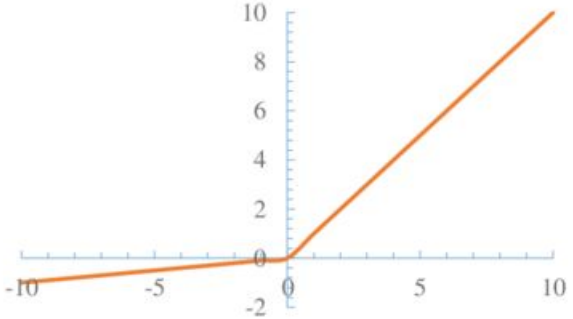
(a)


1
0.5
0
-0.5
-1
-10
-5
0
5
10

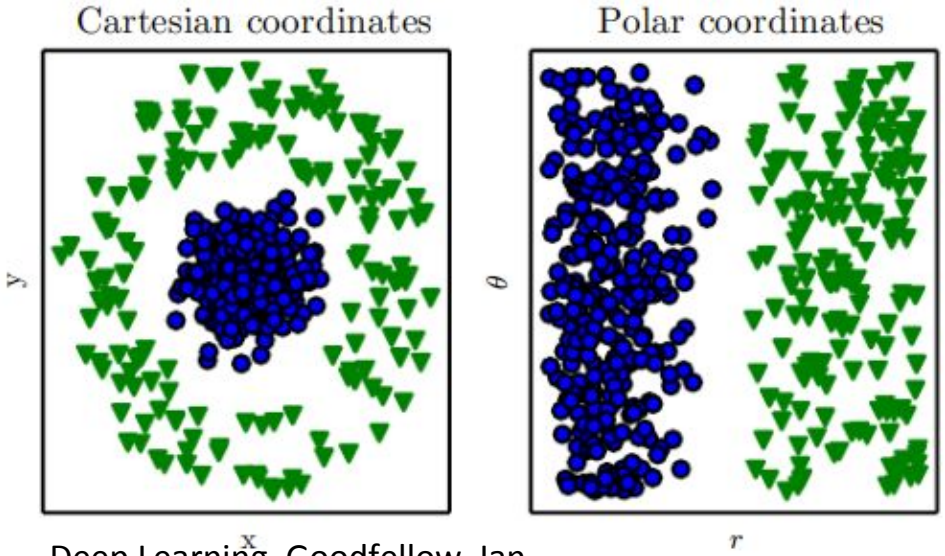
(b)


10
8
6
4
2
0
-2
-10
-5
0
5
10

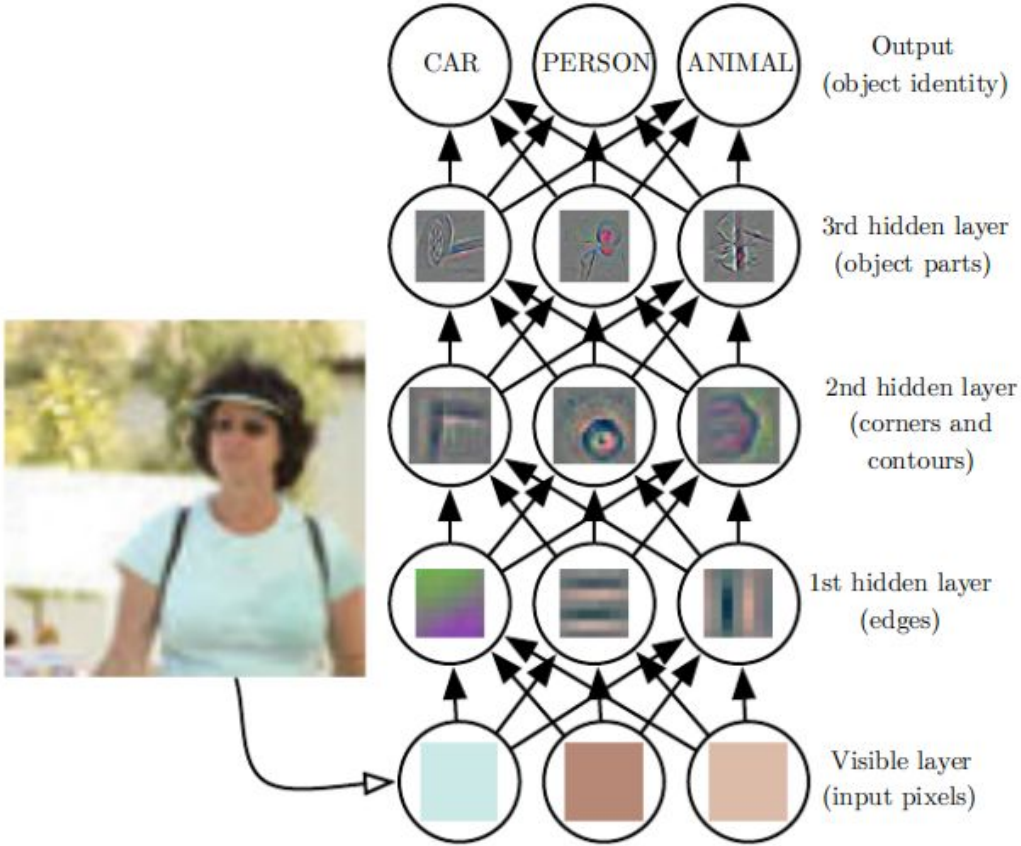
(c)

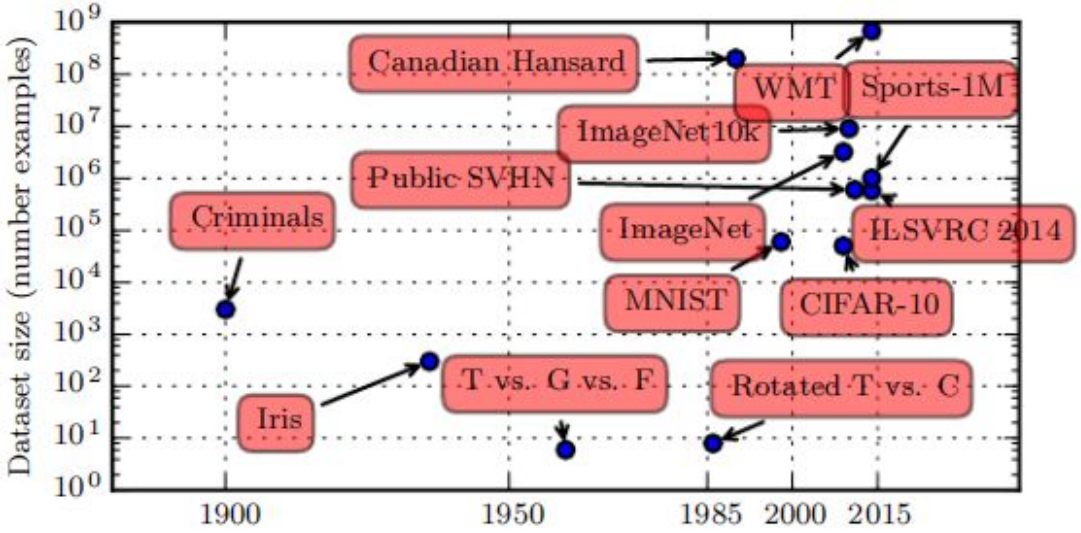

10
8
6
4
2
0
-2
-10
-5
0
5
10

(d)



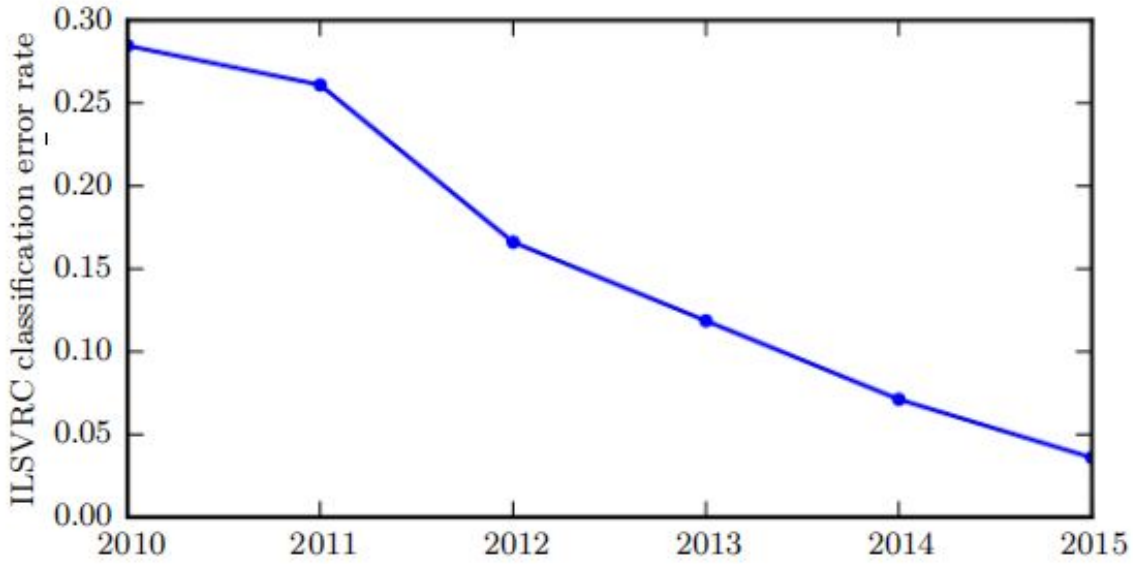
Deep Learning. Goodfellow, Ian


CAR
PERSON
ANIMAL
Output
(object identity)
3rd hidden layer
(object parts)
2nd hidden layer
(corners and
contours)
1st hidden layer
(edges)
Visible layer
(input pixels)

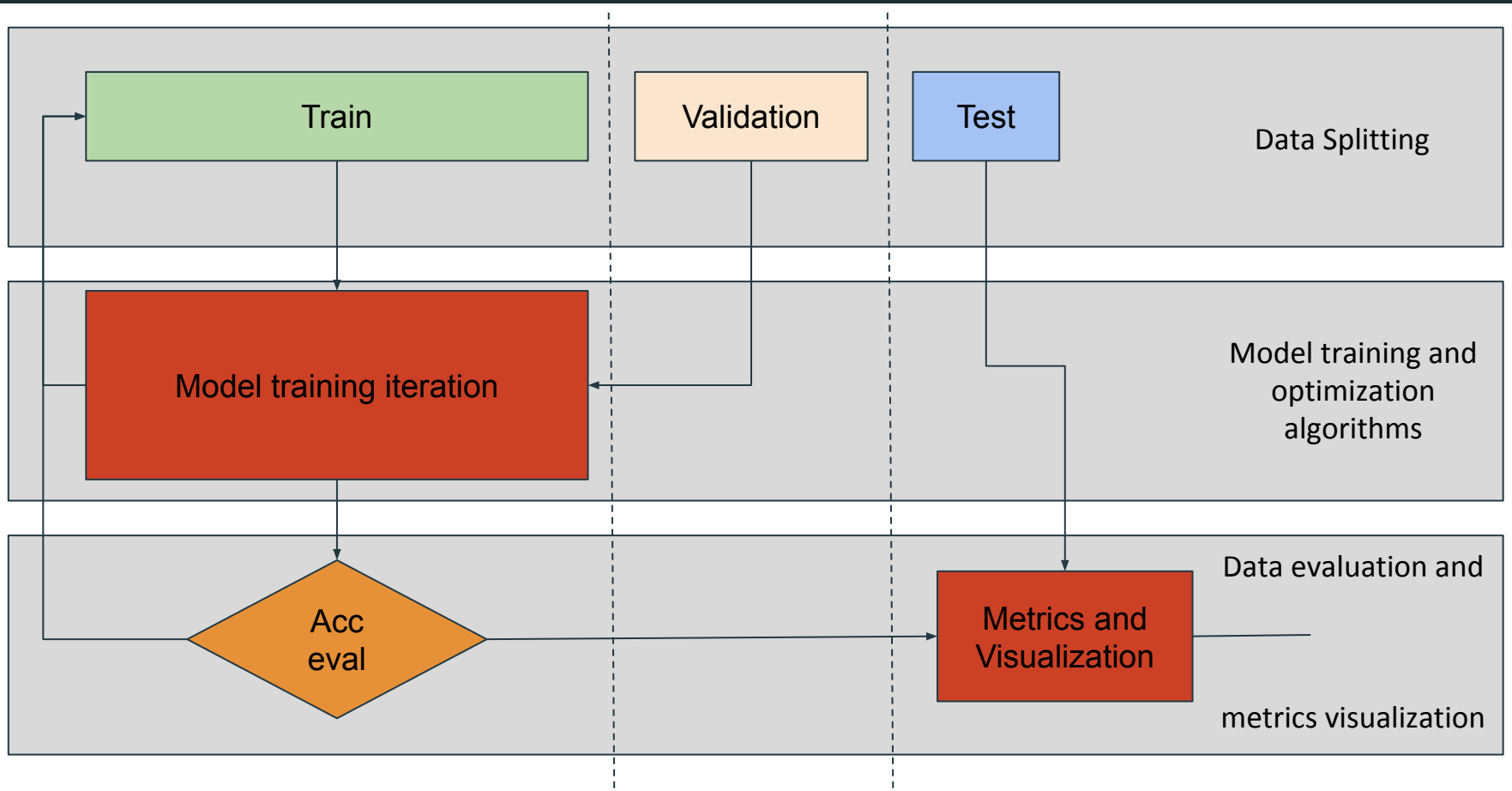

Dataset size (number examples)
10^9
10^8
10^7
10^6
10^5
10^4
10^3
10^2
10^1
10^0
1900
1950
1985
2000
2015
Canadian Hansard
WMT
Sports-1M
ImageNet10k
Public SVHN
Criminals
ImageNet
ILSVRC 2014
MNIST
CIFAR-10
Iris
T vs. G vs. F
Rotated T vs. C


AMD
RADEON
AMD


IMAGENET

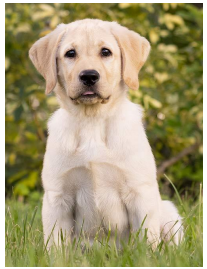

ILSVRC classification error rate
0.30
0.25
0.20
0.15
0.10
0.05
0.00
2010
2011
2012
2013
2014
2015

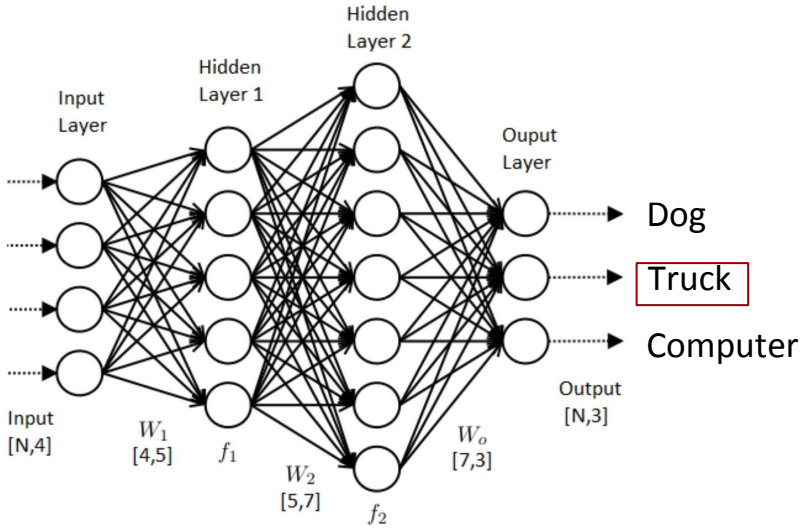
Overview


Train
Validation
Test
Data Splitting
Model training iteration
Model training and optimization algorithms
Acc eval
Metrics and Visualization
Data evaluation and metrics visualization

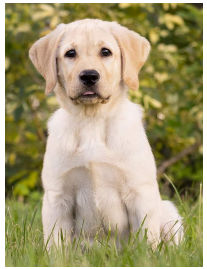
How does it learn?

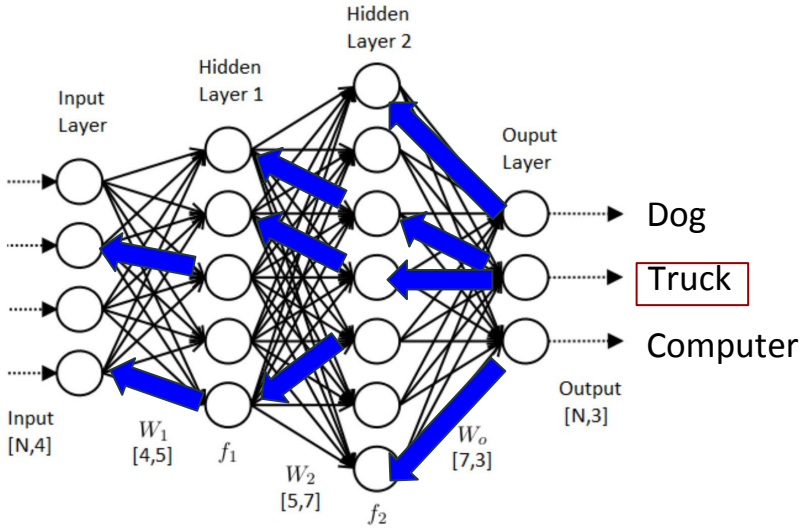
How does it learn?



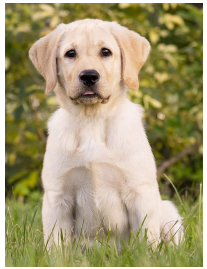


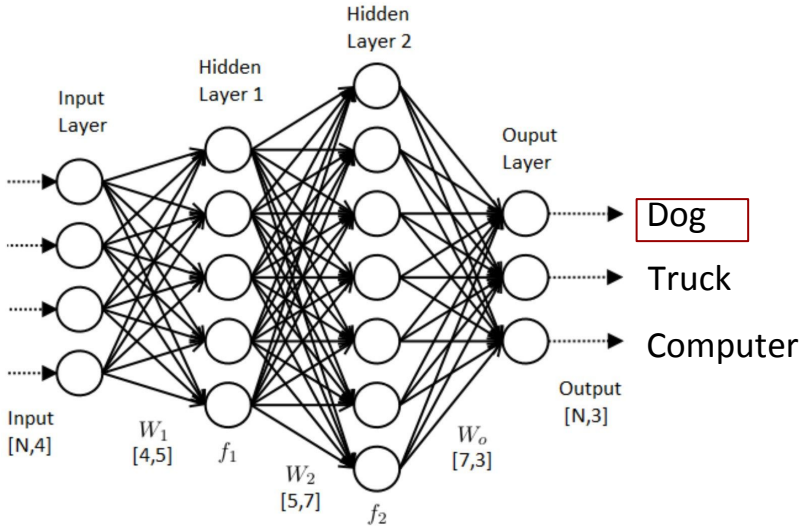
How does it learn?



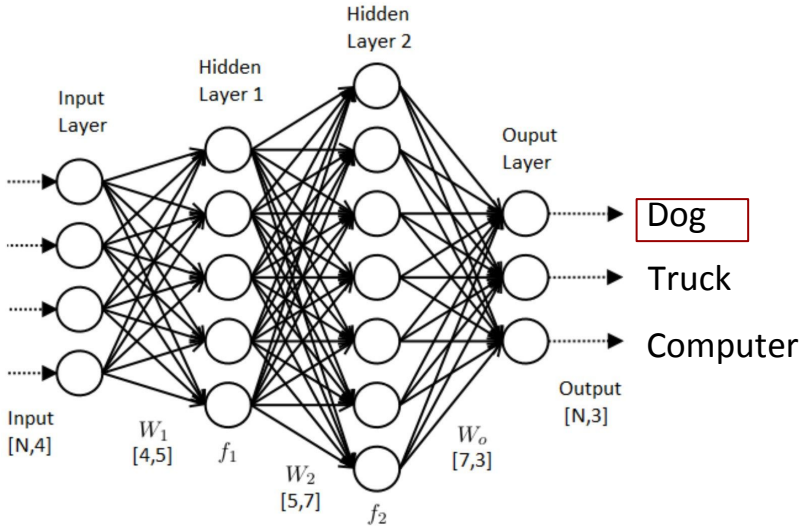


How does it learn?

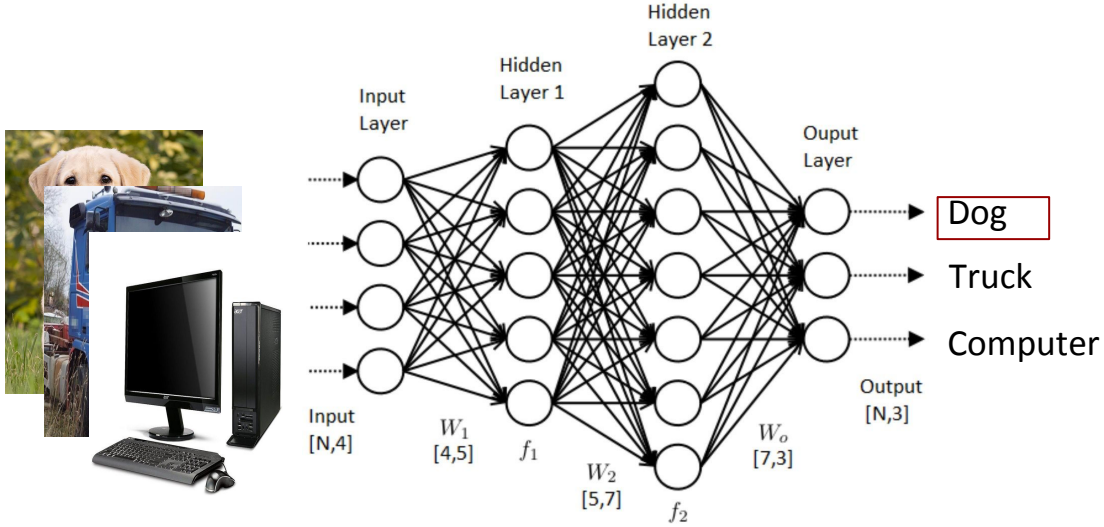




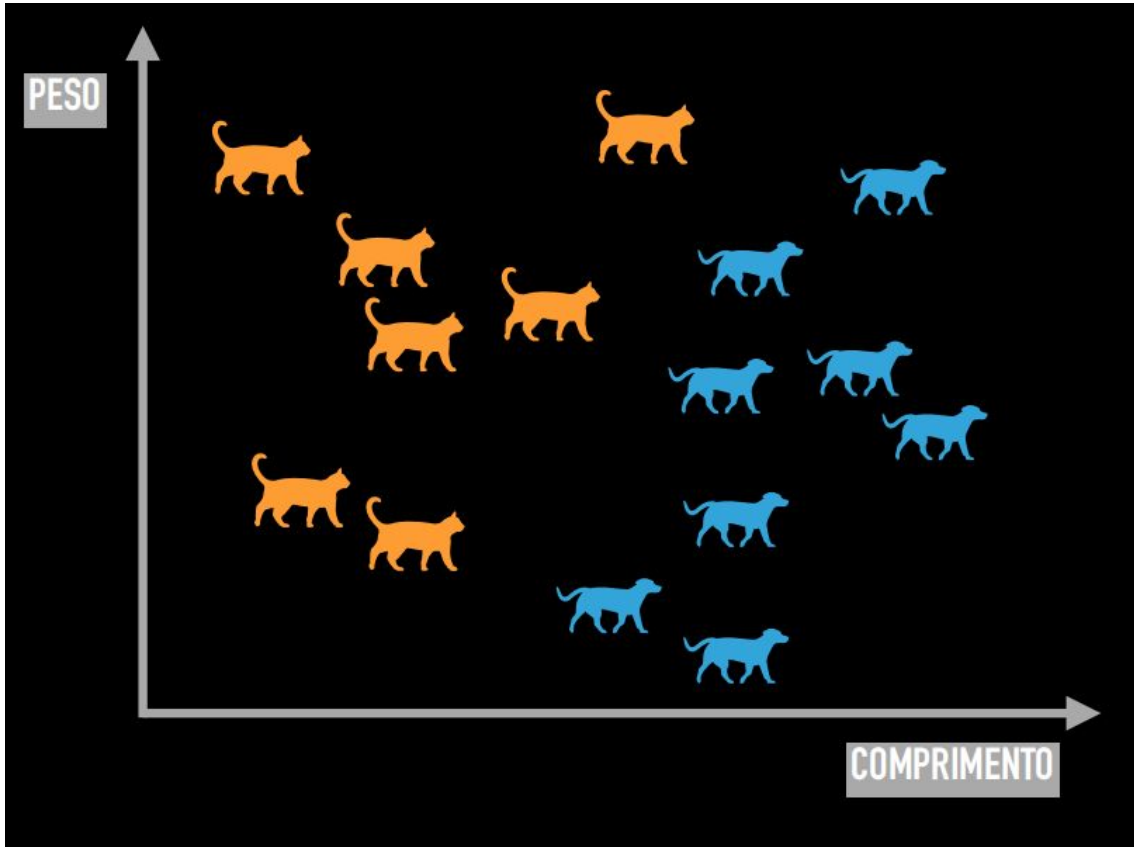
How does it learn?

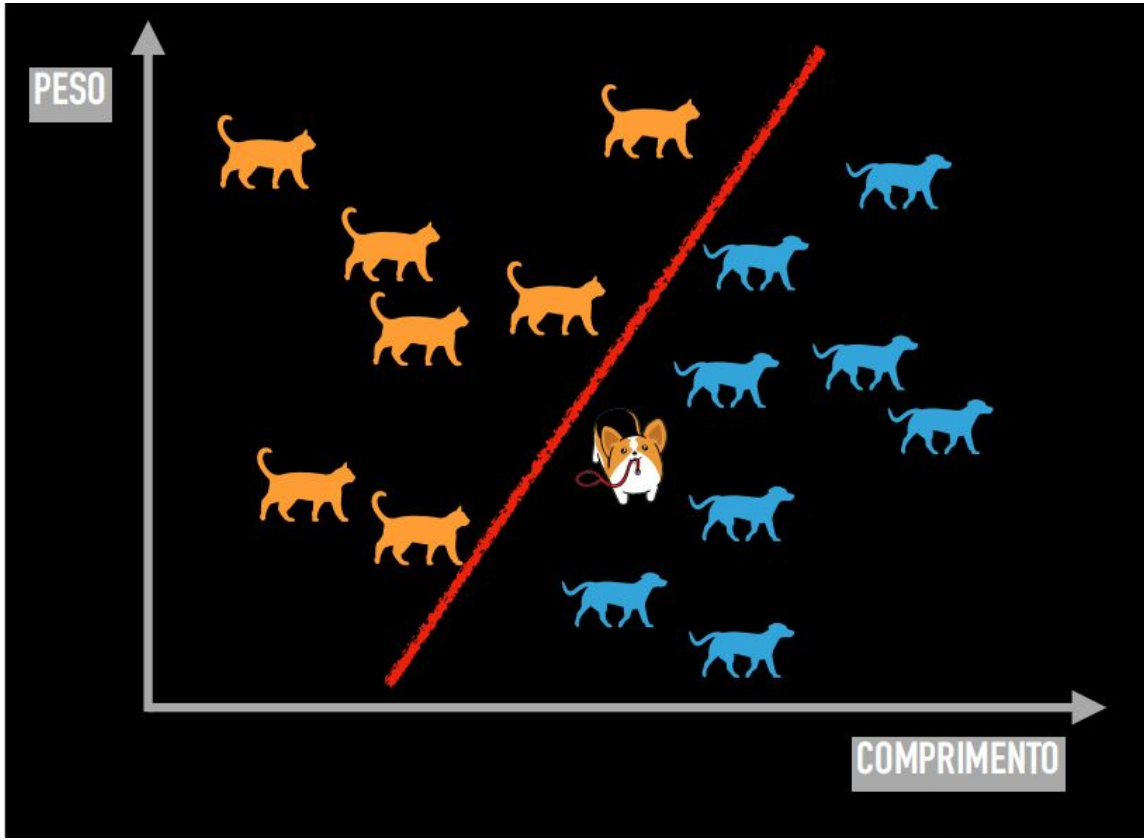


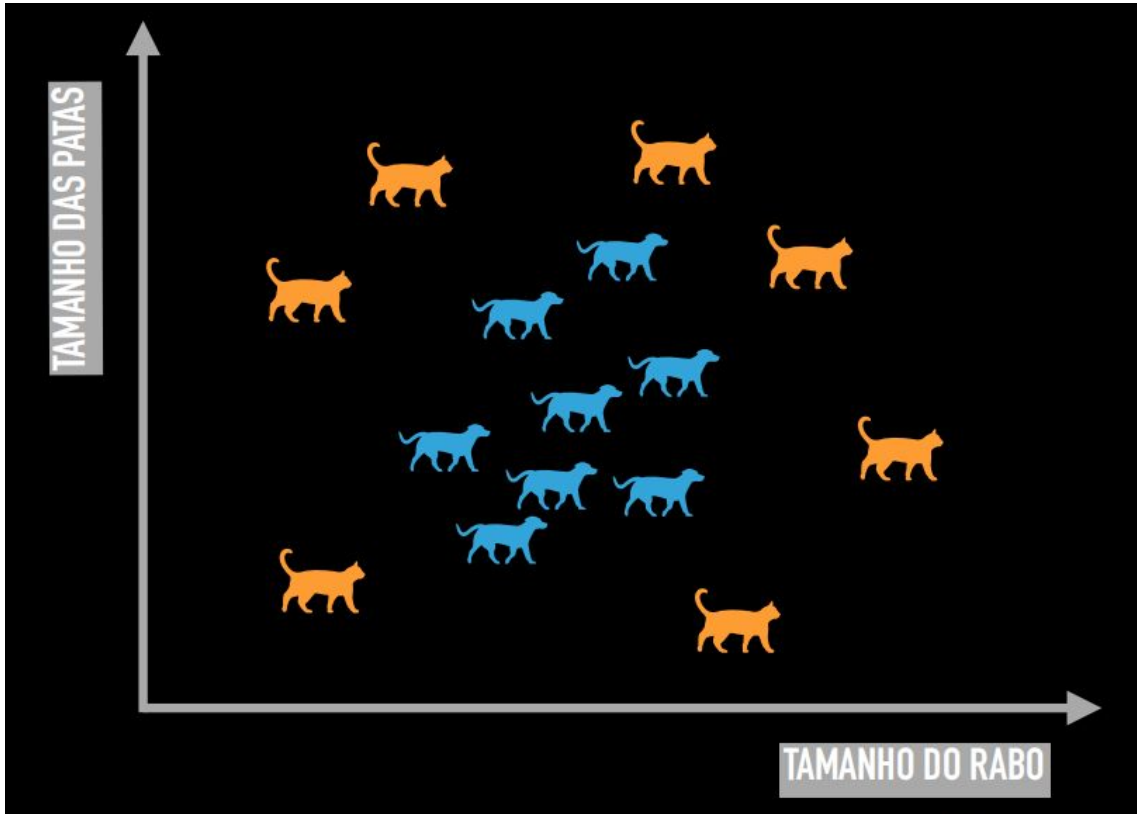
How does it learn?

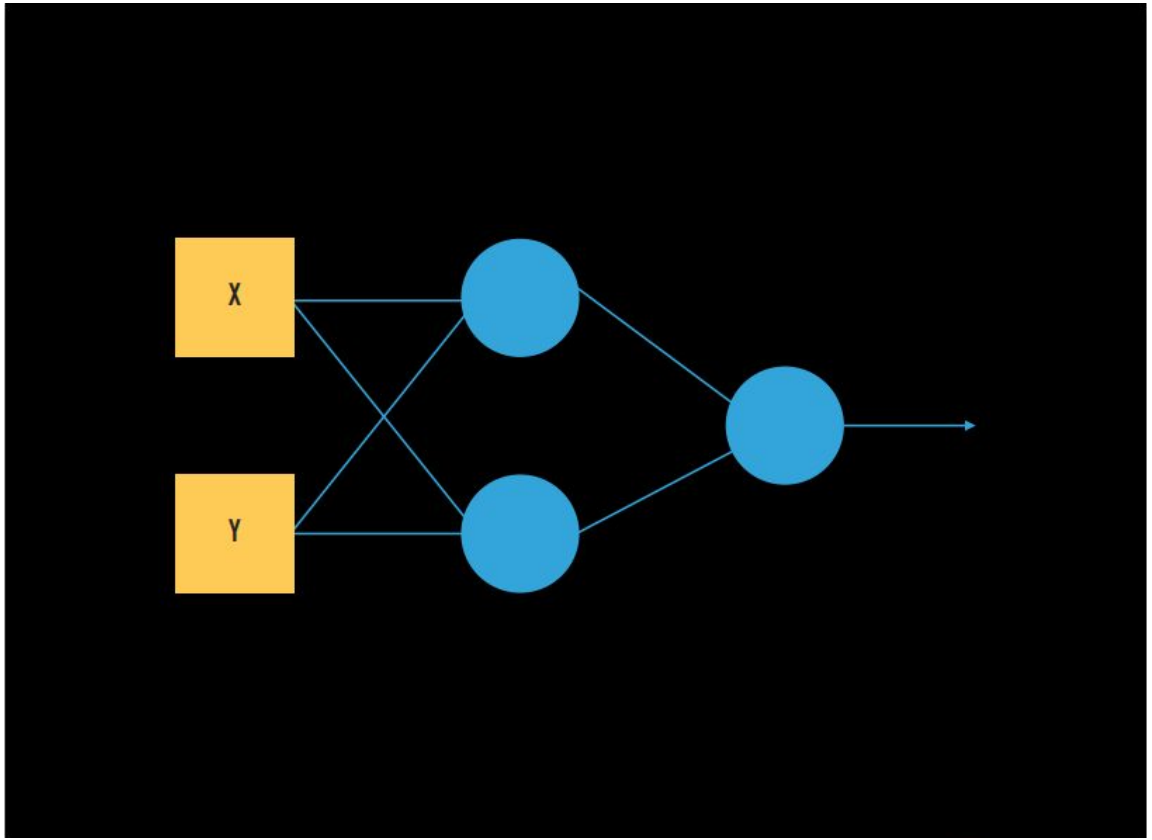


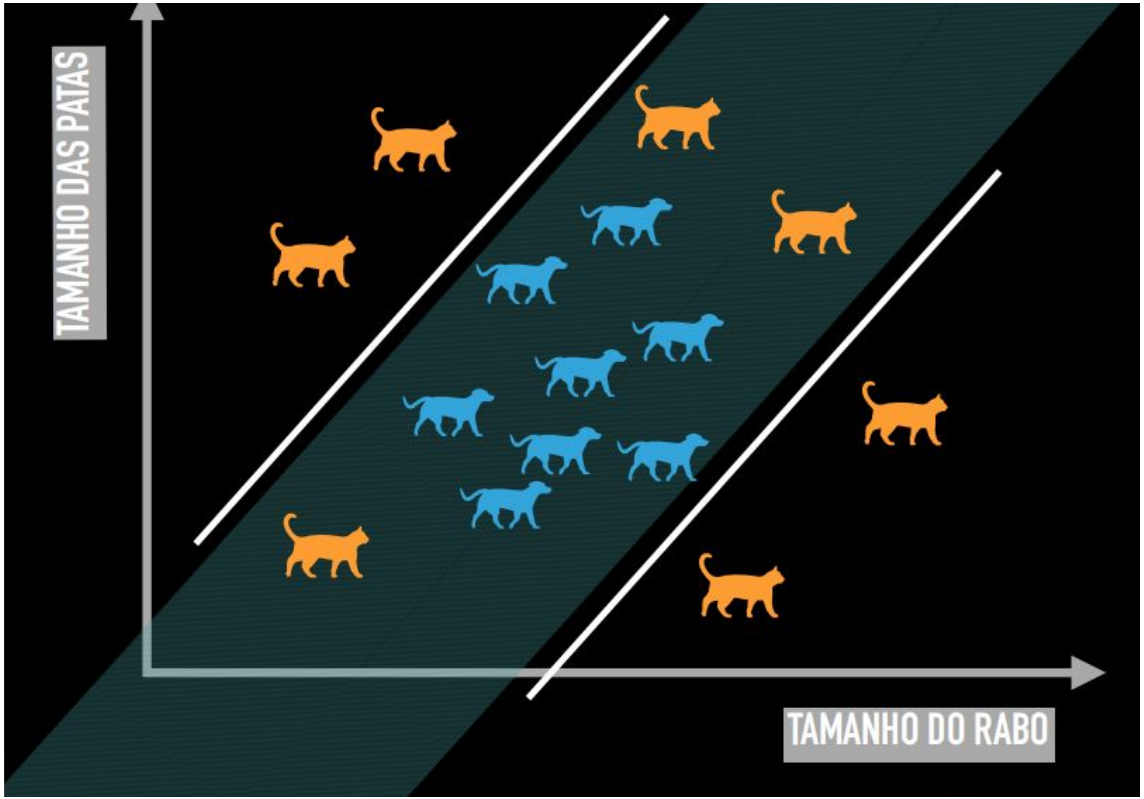
Why is it Deep?

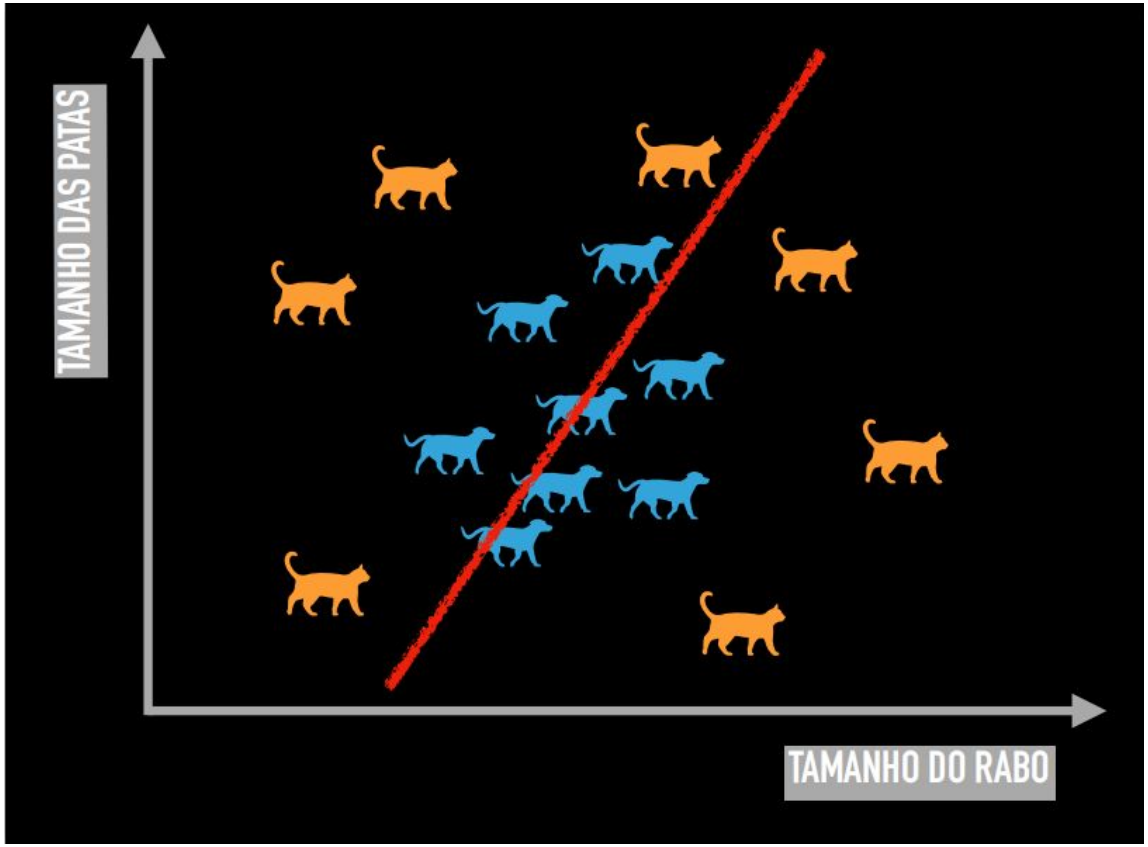

PESO
COMPRIMENTO

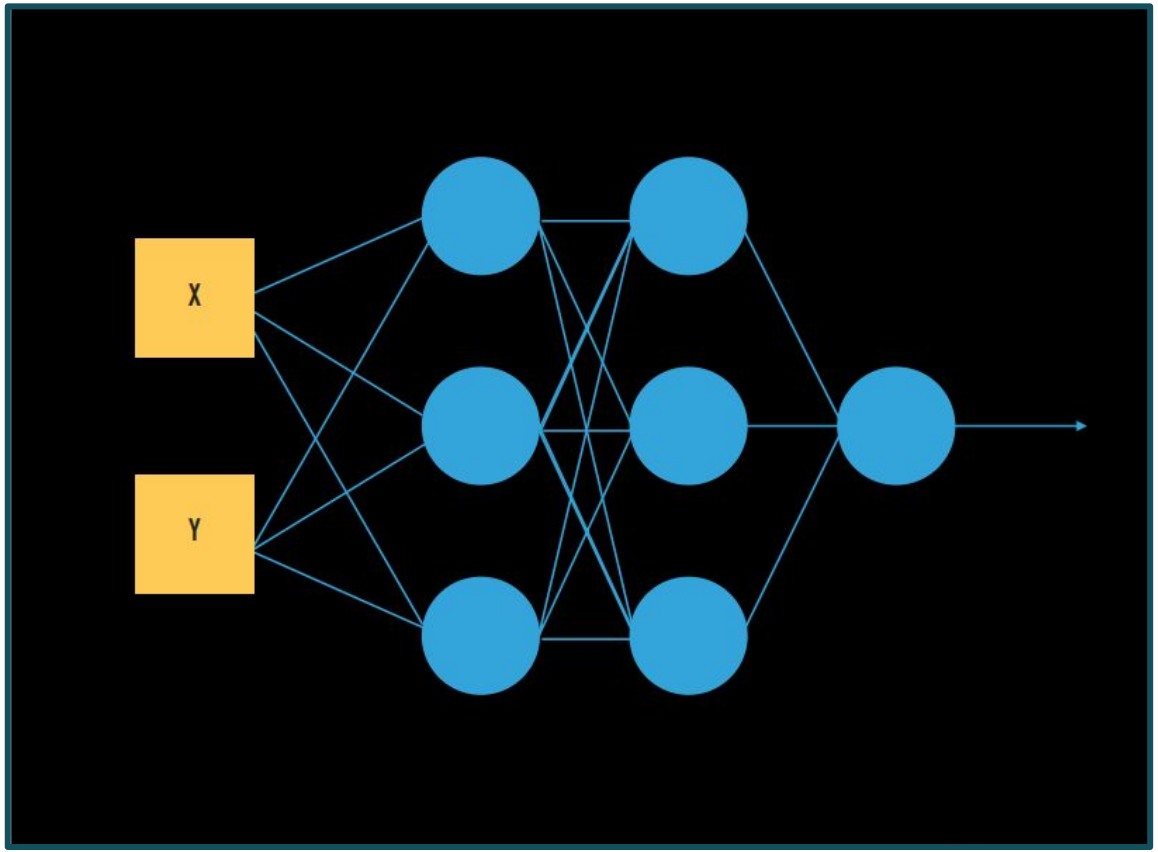

PESO
COMPRIMENTO

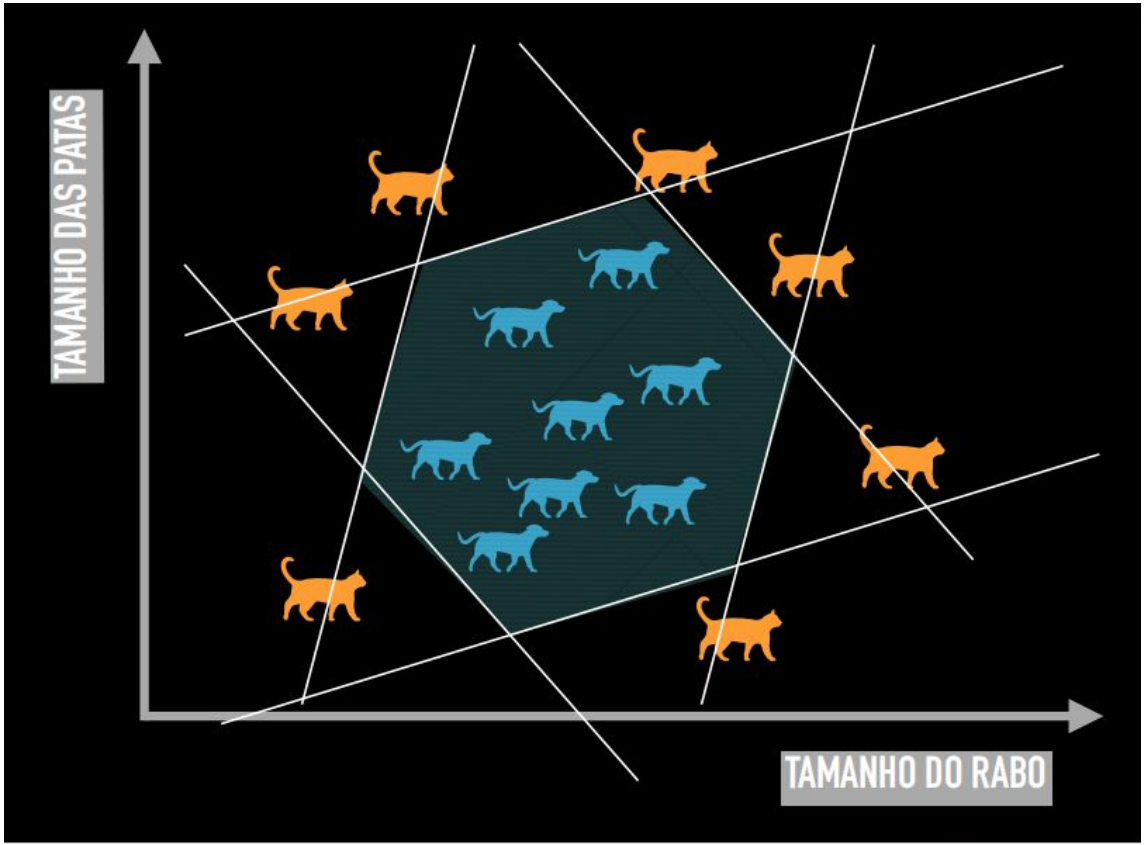

TAMANHO DAS PATAS
TAMANHO DO RABO

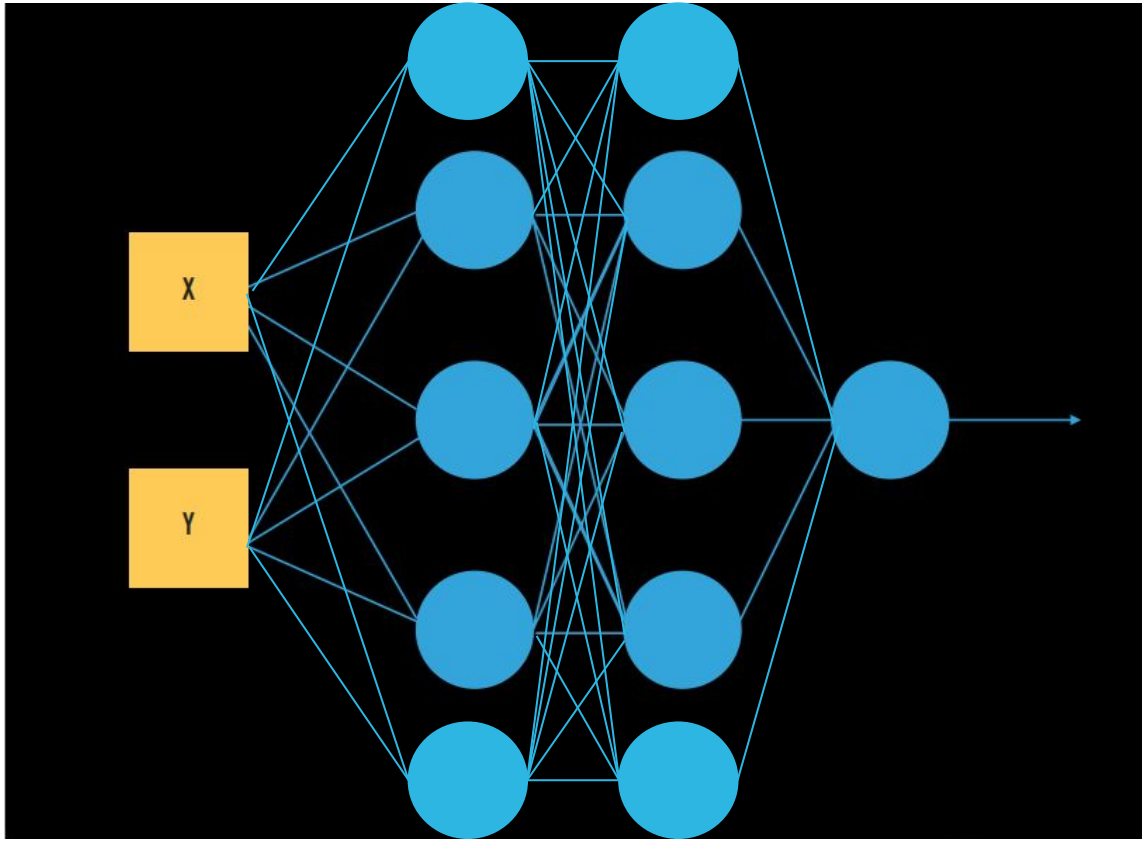

X
Y

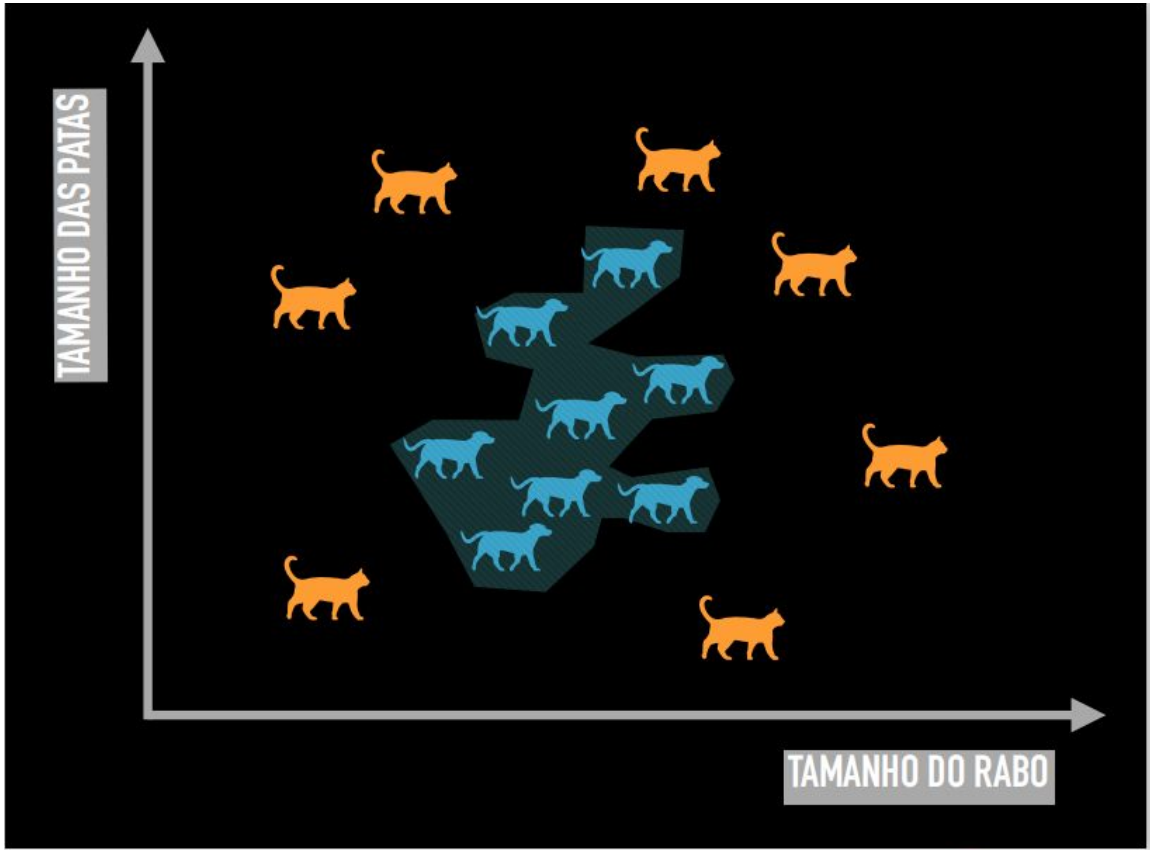

TAMANHO DAS PATAS
TAMANHO DO RABO

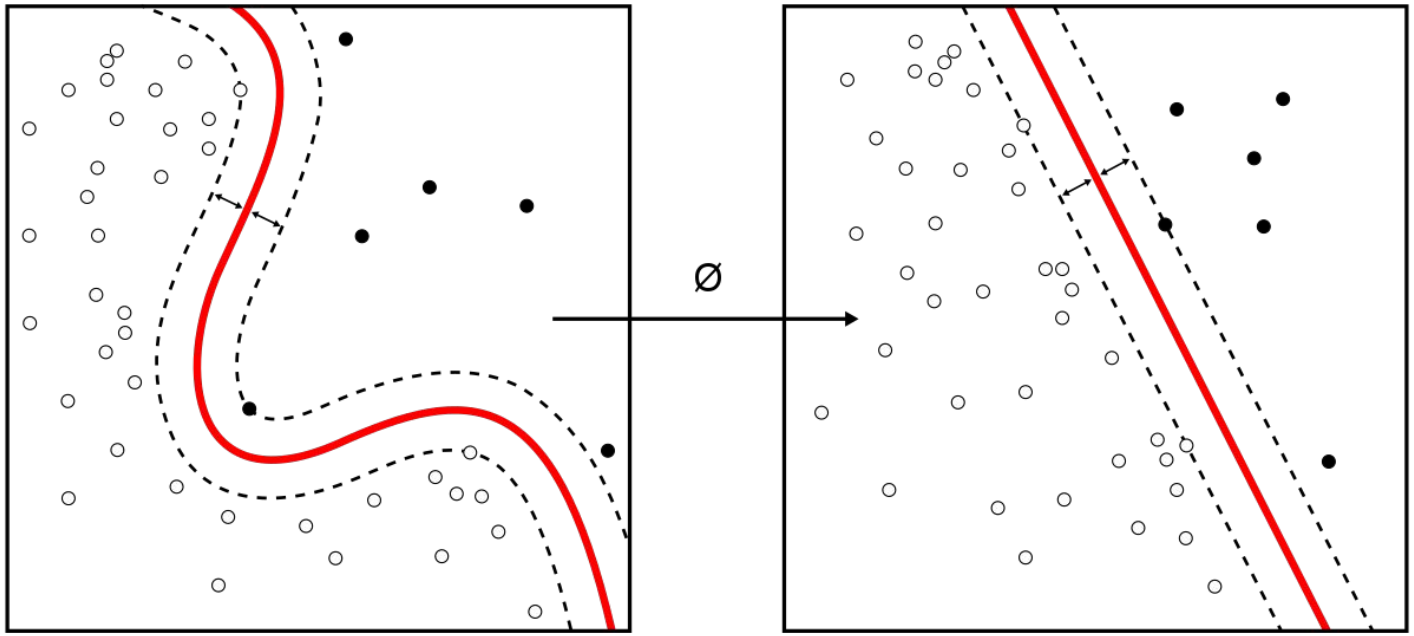

TAMANHO DAS PATAS
TAMANHO DO RABO


X
Y


TAMANHO DAS PATAS
TAMANHO DO RABO

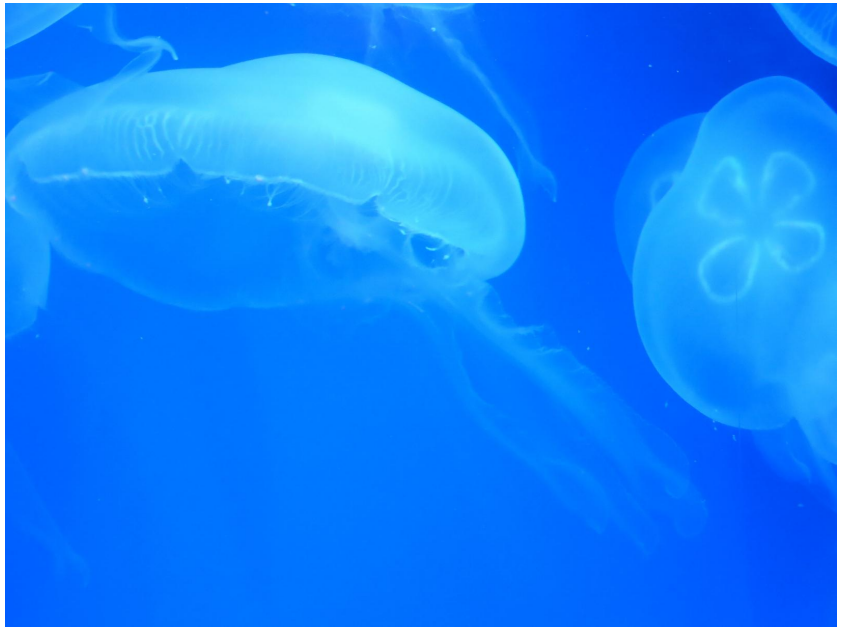

X
Y

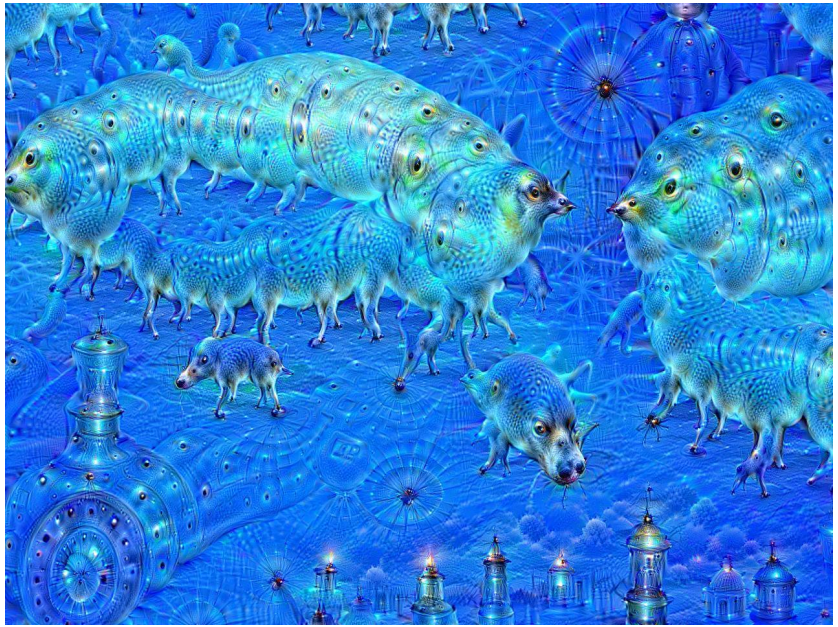

TAMANHO DAS PATAS
TAMANHO DO RABO

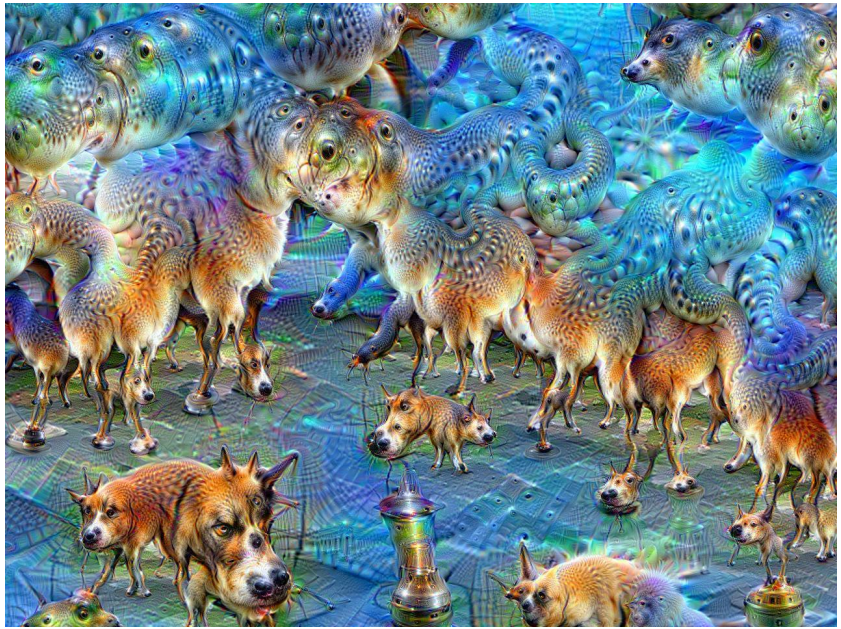

Ø

Interesting stuffs in Deep Learning

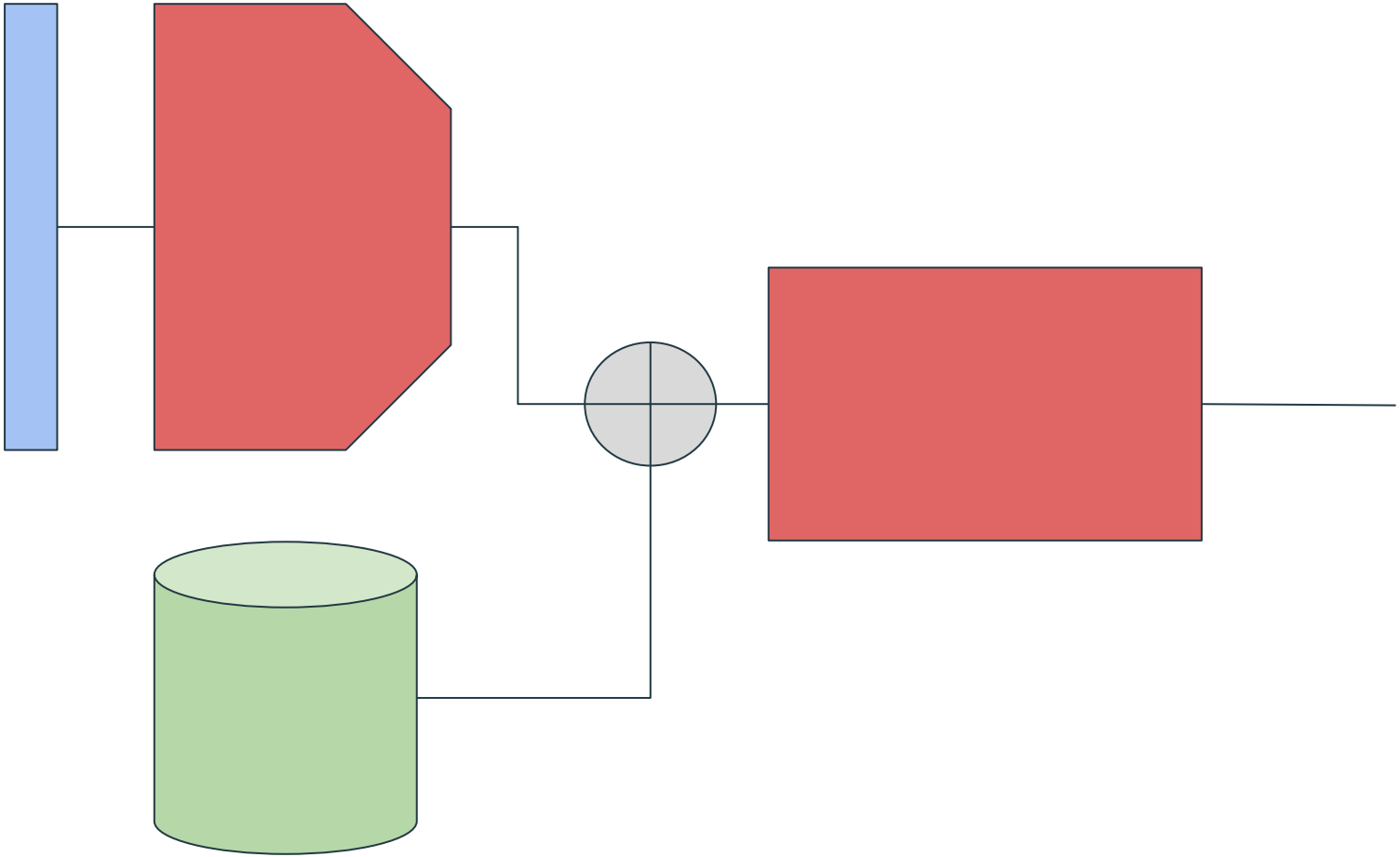
Deep Dream

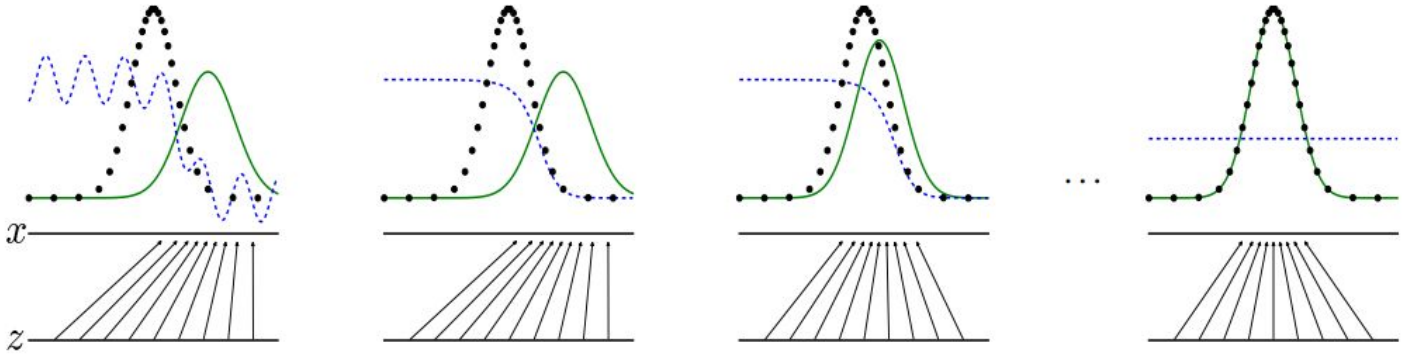




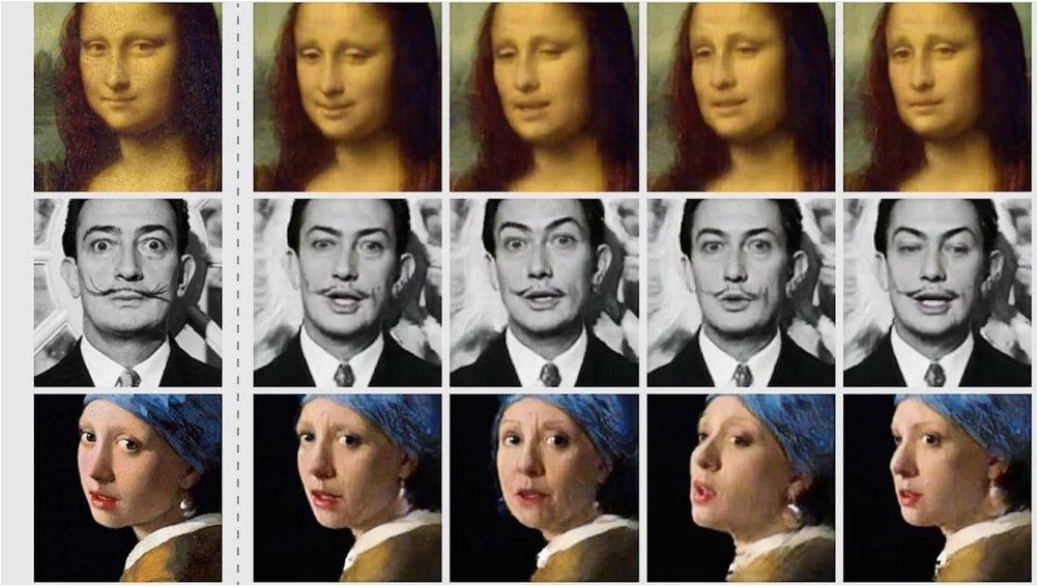


Generative Adversarial Networks

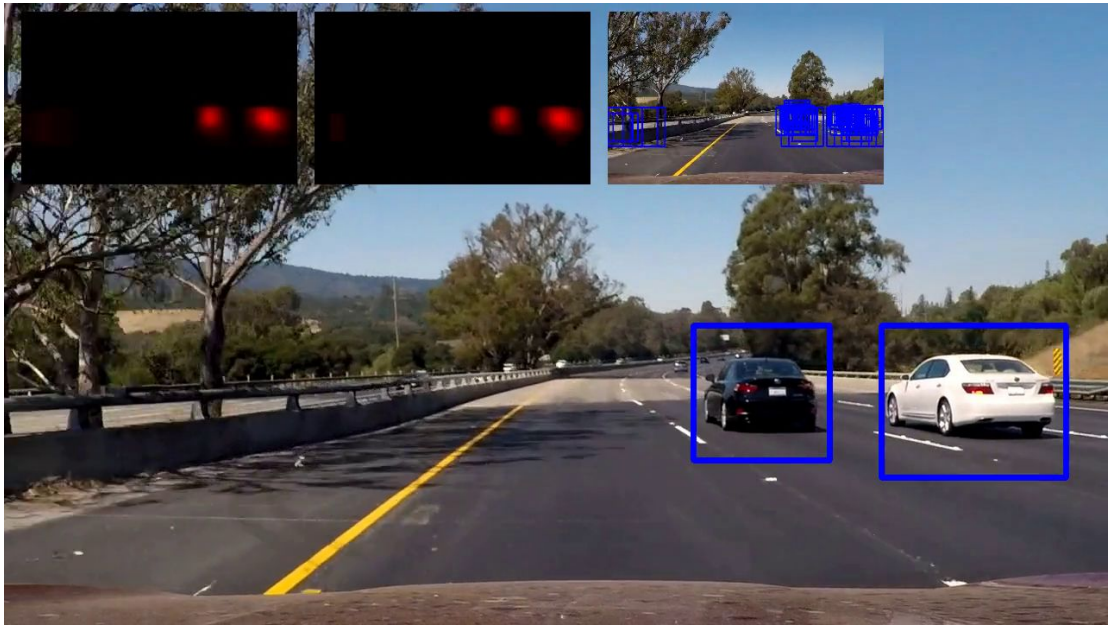


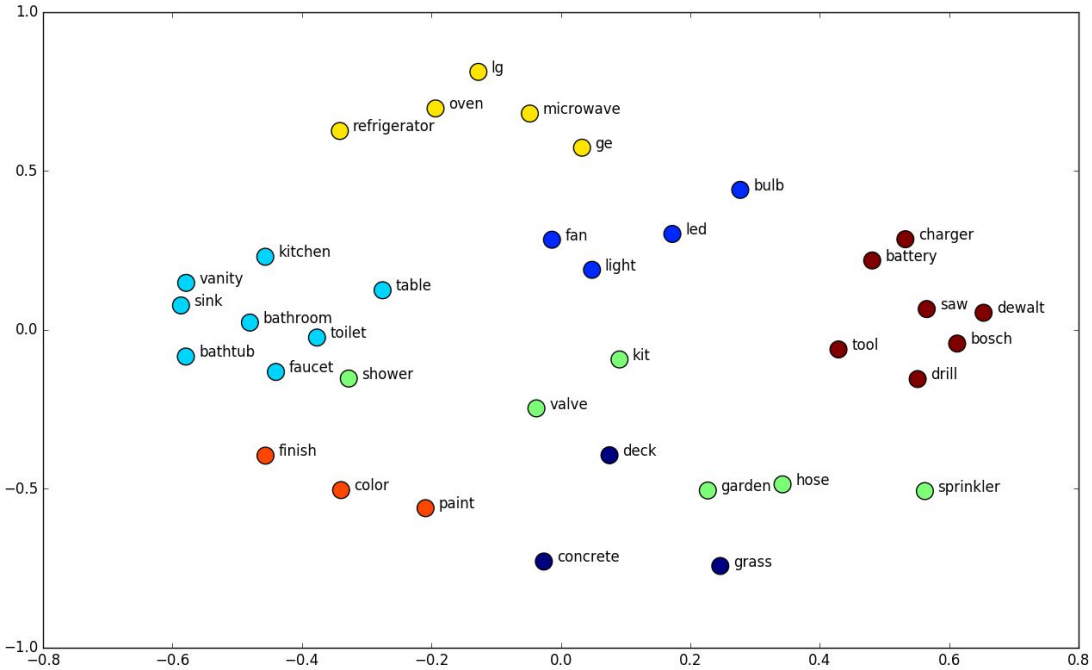

x
z

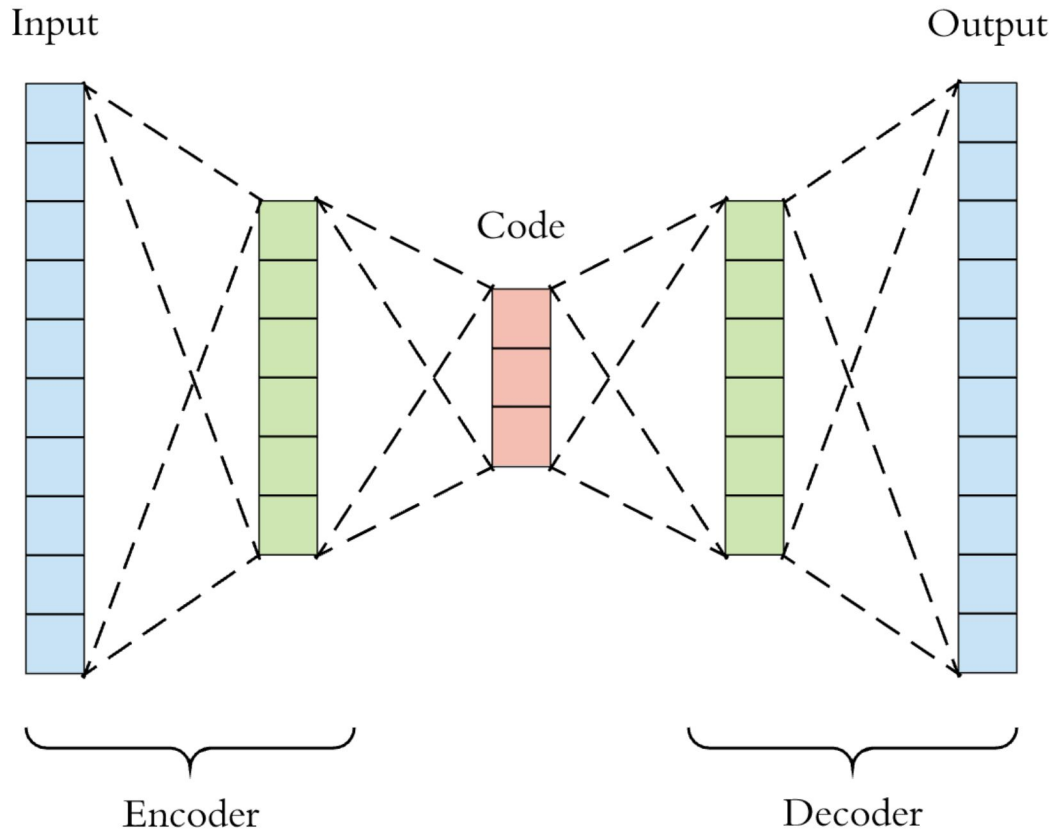
Deep Fake

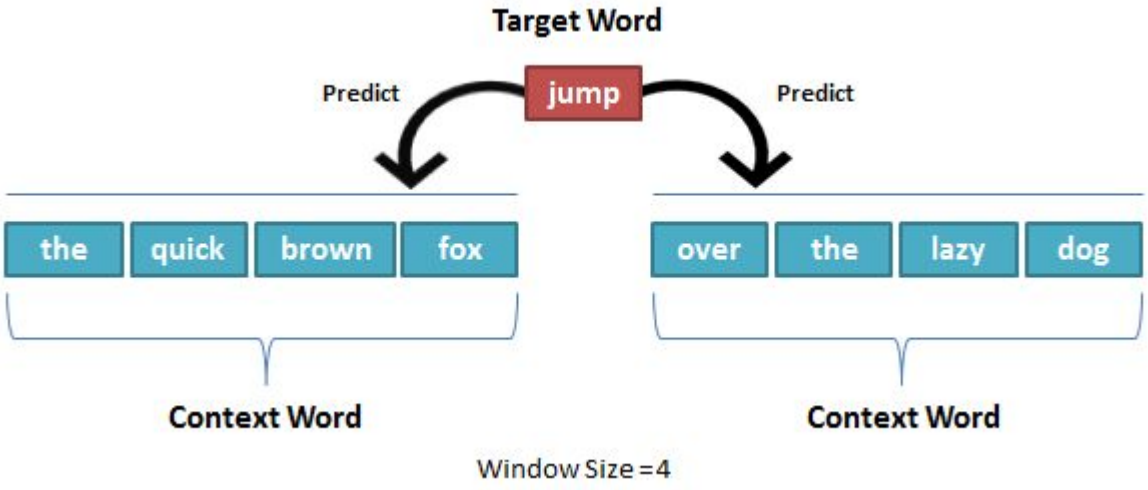


Self-driving Cars

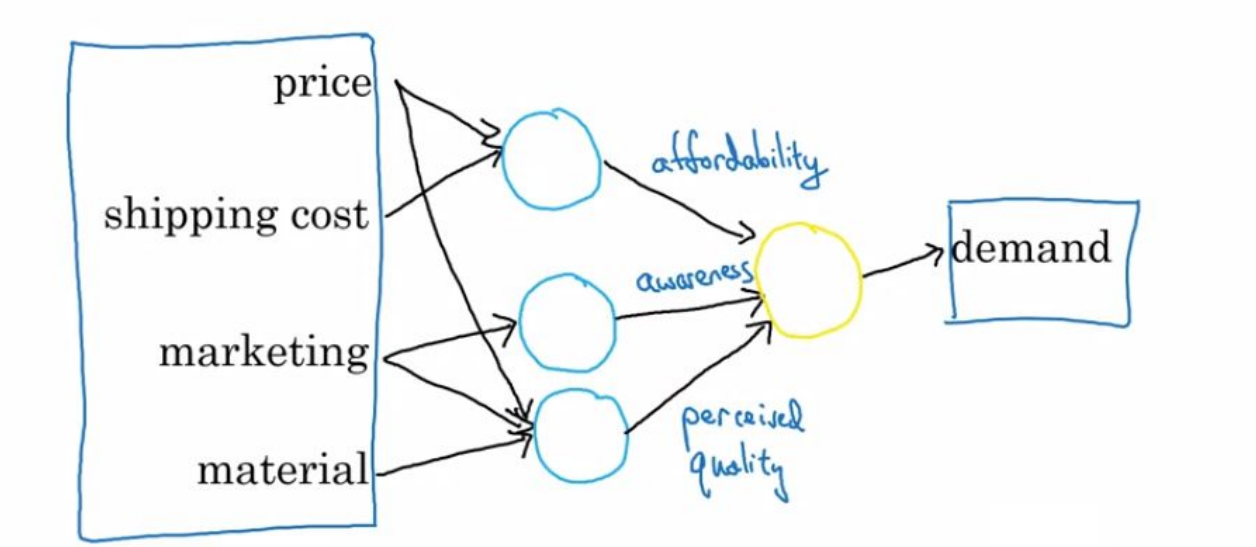



lg
oven
microwave
refrigerator
ge
bulb
fan
led
charger
kitchen
battery
light
vanity
table
saw
dewalt
sink
bathroom
toilet
bosch
tool
bathtub
kit
faucet
shower
drill
valve
finish
deck
color
garden
hose
sprinkler
paint
concrete
grass
1.0
0.5
0.0
−0.5
−1.0
−0.8
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8

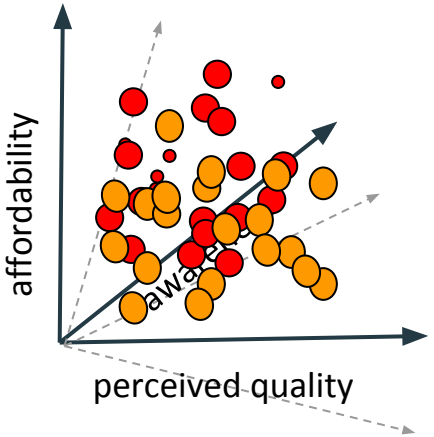

Input
Output
Code
Encoder
Decoder


Target Word
Predict
jump
Predict
the
quick
brown
fox
over
the
lazy
dog
Context Word
Context Word
Window Size = 4

Non-technical explanation of DL



AI for everyone course, by Andrew Ng


affordability
perceived quality

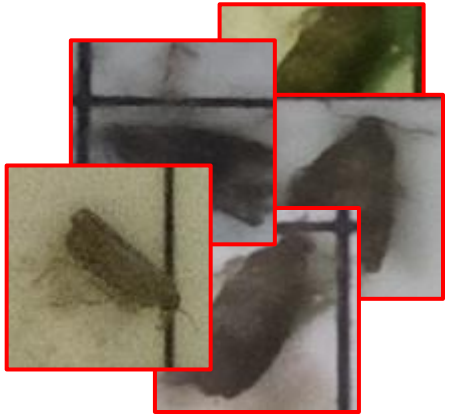
Some problems found in building models

- Unbalanced data (weighted loss, augmentation, etc.)
- Data bias (the ML engineer needs to analyse)
- Small dataset (transfer learning, k-fold cross validation, other ML models rather Deep Learning)
- Constrained resources, i.e., embedded systems (distilling networks, embedded models, network compression)
- Real problem data too distant from the training dataset (augmentation, engineering solutions)
- The need of a massive pre-processing on the data
- Lack of visualization and explanation

Unbalanced Data

~58%

%22%



~20%

Data Bias

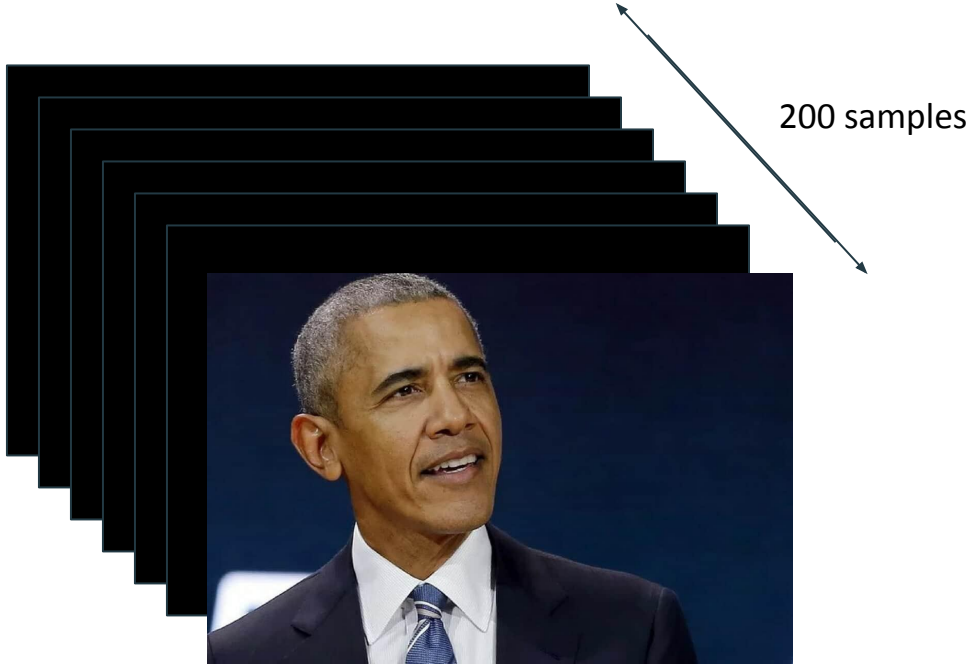

Portuguese English Turkish Detect language
English Portuguese Russian
Translate
O bir hemsire
O bir doktor
26/5000
She is a nurse
He is a doctor

(a) Three samples in criminal ID photo set S_c.

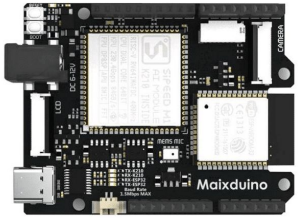
(b) Three samples in non-criminal ID photo set S_n


CRIMINAL
NON-CRIMINAL

Small Dataset

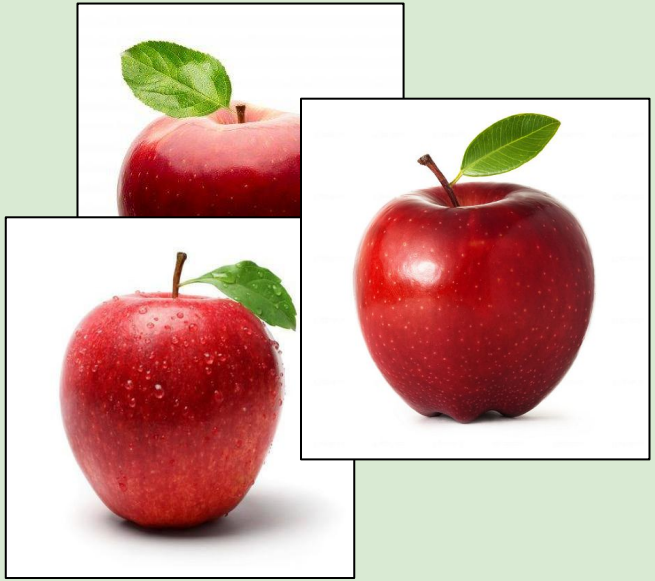

200 samples

Constrained Resources


Maixduino

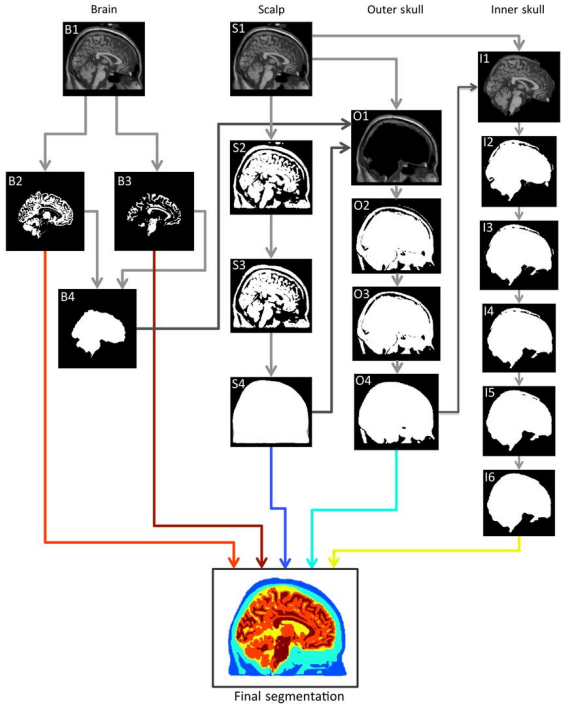
The representation learned from the data diverges from the real data distribution

Training and Test data

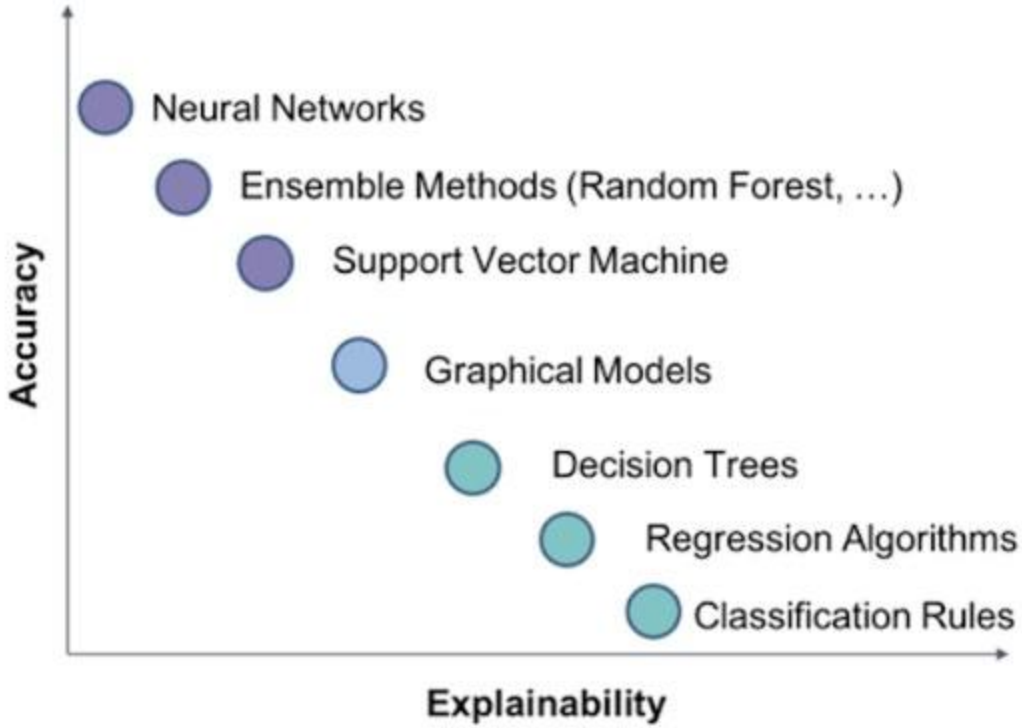


Real data when deployed

Massive Pre-processing


Brain
Scalp
Outer skull
Inner skull
B1
B2
B3
B4
S1
S2
S3
S4
O1
O2
O3
O4
I1
I2
I3
I4
I5
I6
Final segmentation

Lack of explainability


Neural Networks
Ensemble Methods (Random Forest, …)
Support Vector Machine
Graphical Models
Decision Trees
Regression Algorithms
Classification Rules
Accuracy
Explainability

Techniques used to improve generalization and accuracy

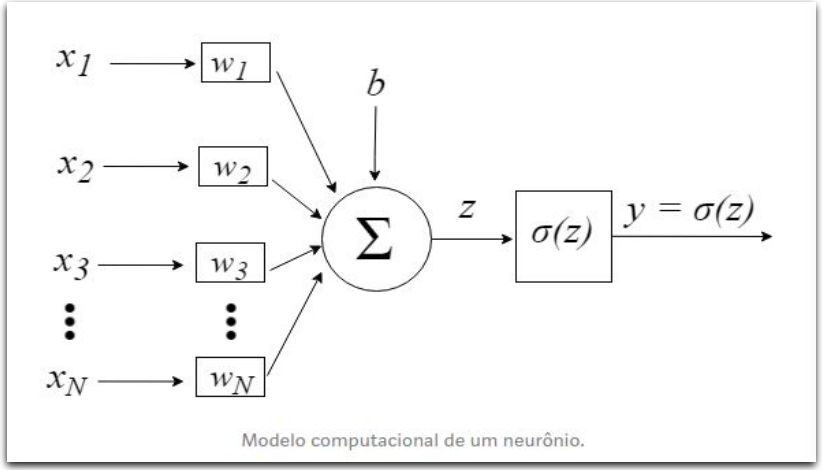
- Normalized inputs (even between layers)
- DropOut (like our brain, sparse networks performs better than dense ones)
- Augmentation (synthetically augmenting samples can help in generalization)
- Transfer learning (small datasets combined to large models improve the risks of overfitting. With transfer learning is needed less epochs and mitigates the risk of overfitting).
- Avoiding temporal correlation between samples

Where and when should you apply deep learning models?

- Large amount of data (big data problem)
- Often in images classification, detection, segmentation Deep Learning outperforms other models
- spacing-correlated, time-series data (images, videos, audio, text, e.g., sequencing)
- non-straight data
- hard-to-describe patterns (how does a cat looks like? Can we describe exactly?)

Deeper into Deep Learning

Mathematical model of a Neuron



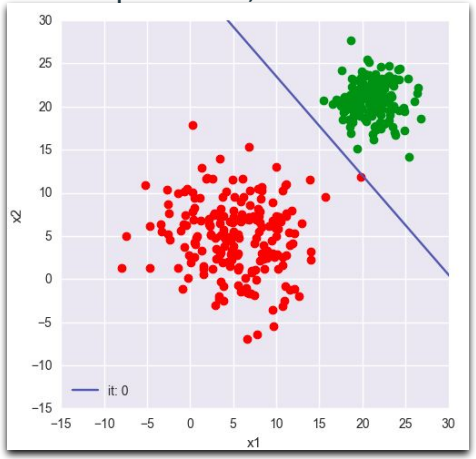
Modelo computacional de um neurônio.

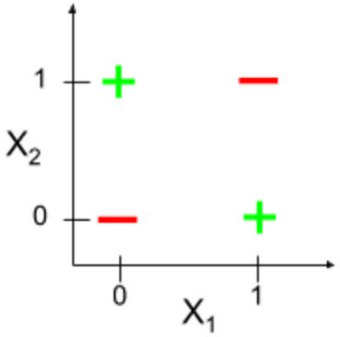
$$z = \sum_{i=1}^{N} x_i w_i + b$$

$$w_{t+1} = w_t + f(x, y) - f(x, \hat{y}).$$

Regressor

- Classification Problem: it performs well in linearly separable problems .
- Regression: it performs well for linear data.
- In real problems, it doesn't work so well where there are multidimentional data (many covariants)





Loss Functions

- mean square error
- log loss (cross-entropy)
- L1 (absolute difference)
- NLLLoss

Mean Square Error

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

Cross-Entropy

$$H(p, q) = -\sum_{x} p(x) \log q(x)$$

L1 Loss

$$L = \sum_{i=1}^{n} |y_i - f(x_i)|$$

NLLLoss

$$\text{loss}(x, class) = -\log\left(\frac{\exp(x[class])}{\sum_j \exp(x[j])}\right) = -x[class] + \log\left(\sum_j \exp(x[j])\right)$$

Evaluation

How to evaluate a machine learning model?

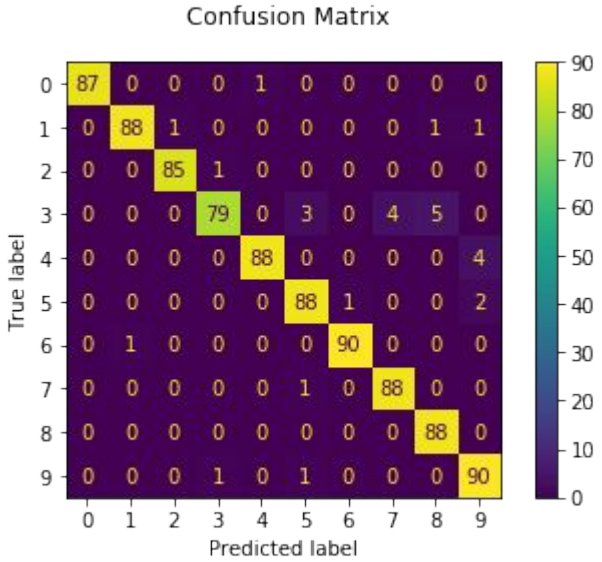
What each metric means?

There’s an overall metric?

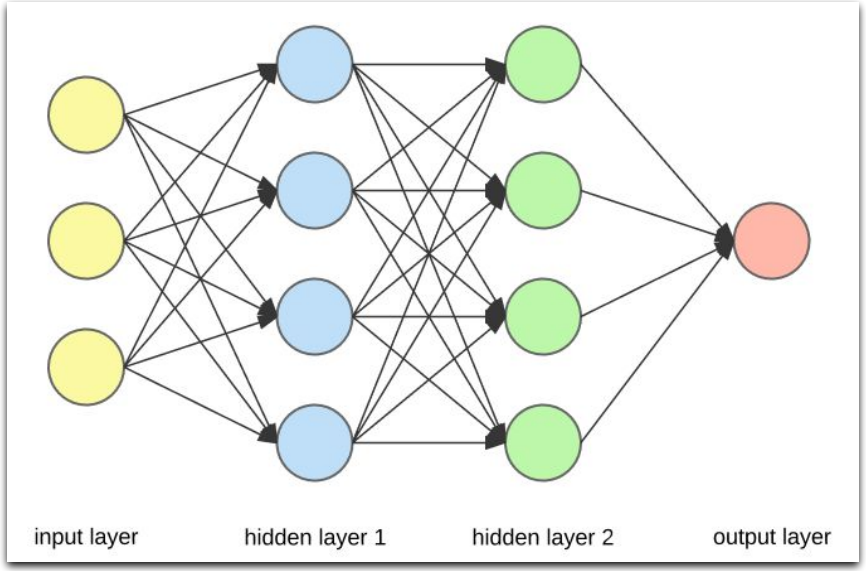
Metric Visualization

Evaluation Metrics

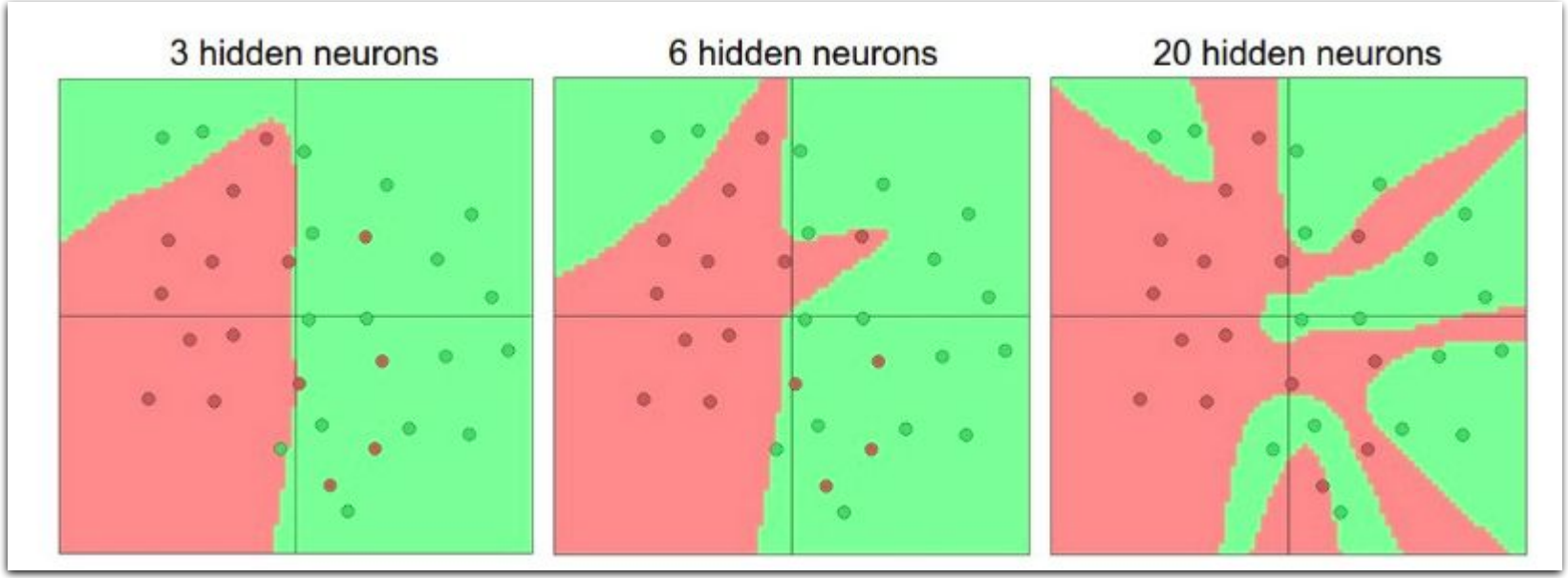
- Loss ??
- accuracy
- true positive rate
- true negative rate
- precision
- recall
- sensitivity
- specificity
- F1-score



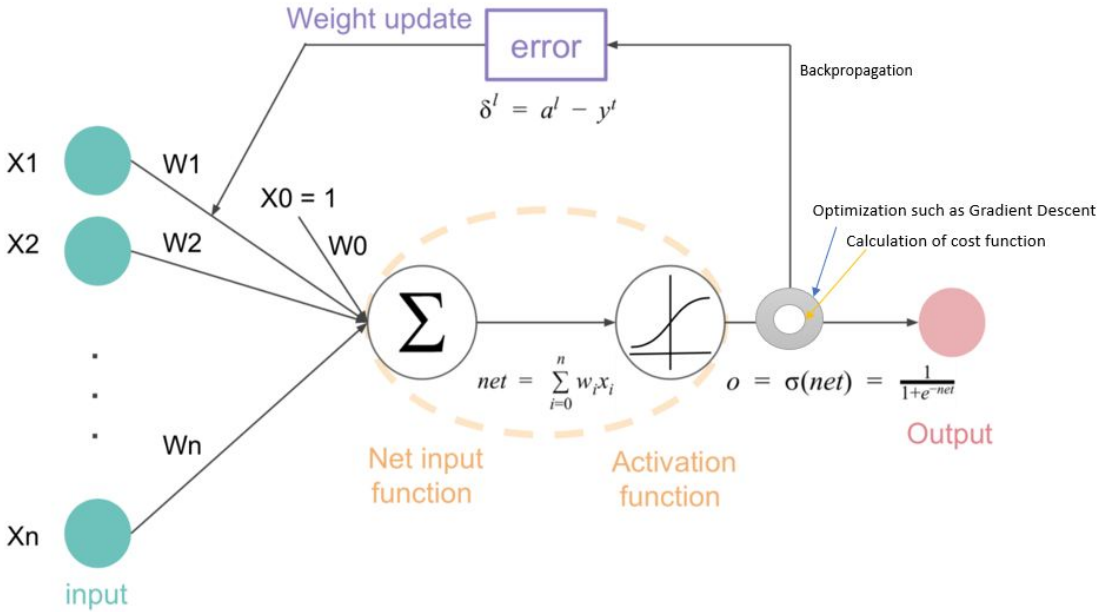
Multi Layer Perceptron



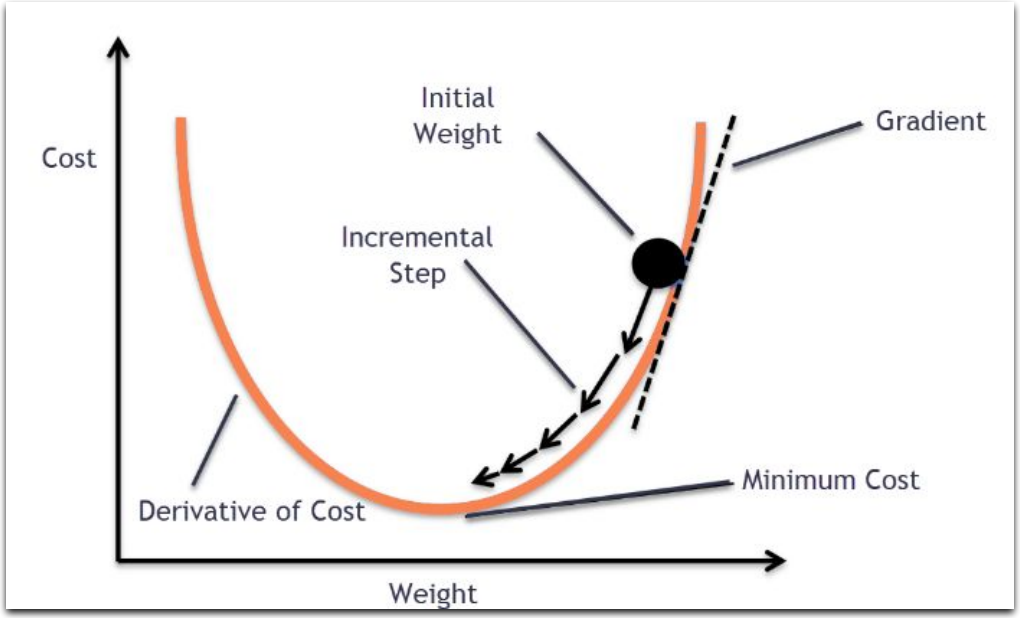
Multi Layer Perceptron



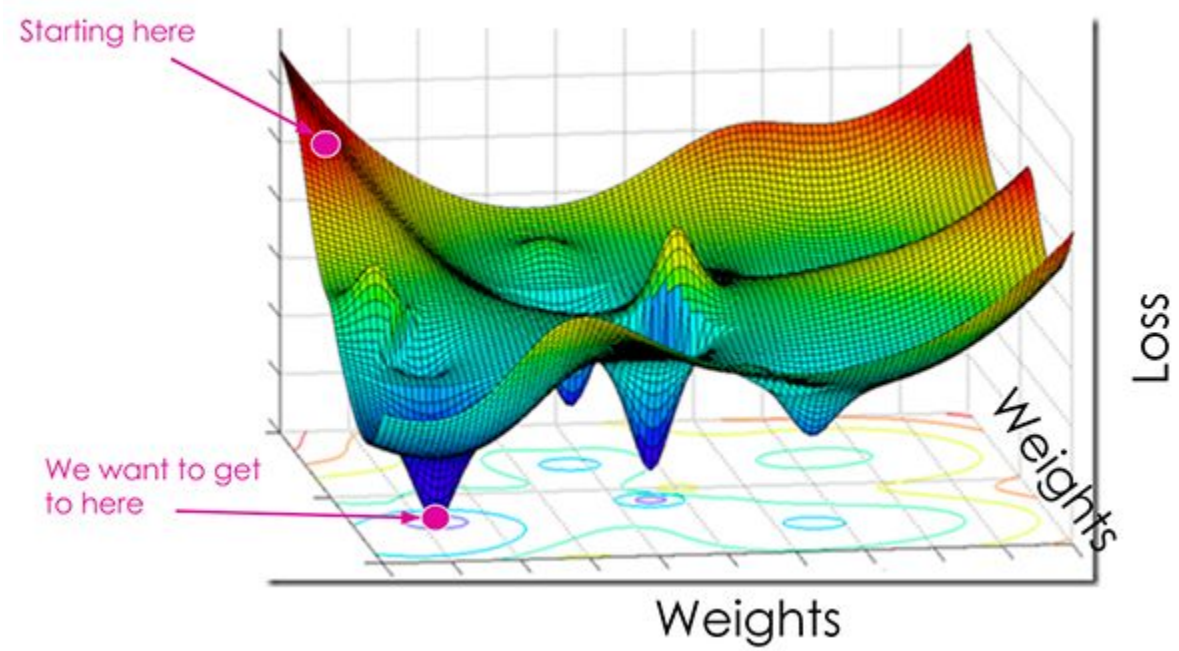
Multi Layer Perceptron

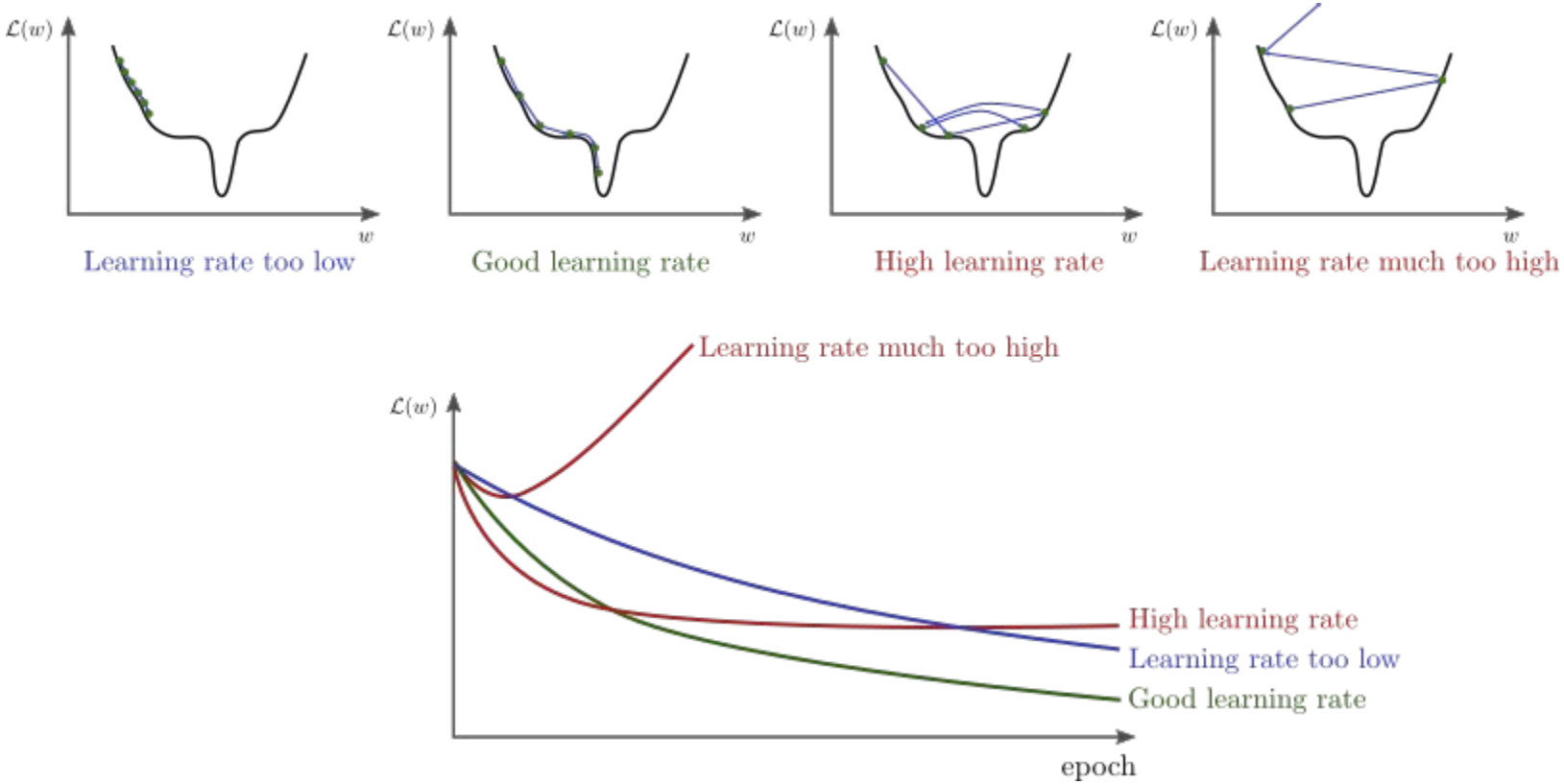


Gradient Descent

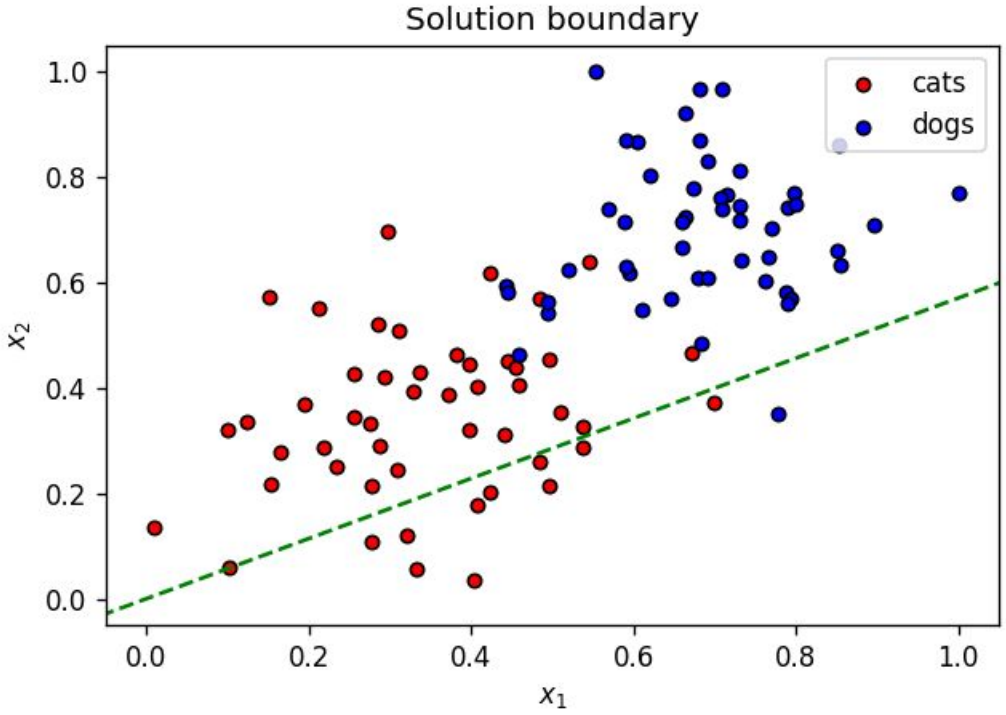


Weight Initialization



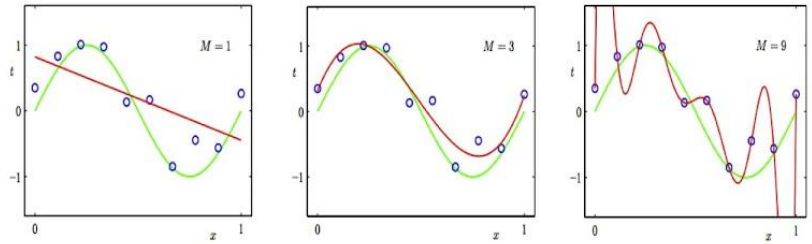

$\mathcal{L}(w)$
w
Learning rate too low
$\mathcal{L}(w)$
w
Good learning rate
$\mathcal{L}(w)$
w
High learning rate
$\mathcal{L}(w)$
w
Learning rate much too high
Learning rate much too high
$\mathcal{L}(w)$
High learning rate
Learning rate too low
Good learning rate
epoch

Vanishing Gradient


Solution boundary
cats
dogs
1.0
0.8
0.6
0.4
0.2
0.0
x_2
0.0
0.2
0.4
0.6
0.8
1.0
x_1

Overfitting and Underfitting

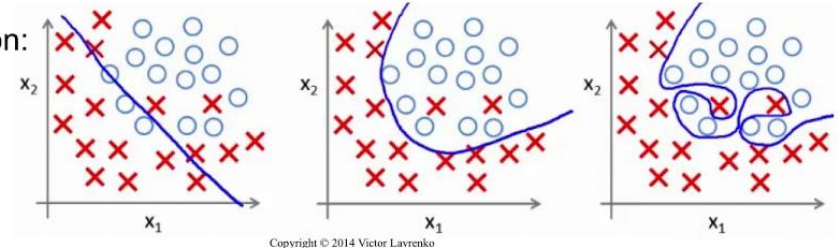
Regression:



predictor too inflexible:
cannot capture pattern

predictor too flexible:
fits noise in the data

Classification:



Regularizers

- DropOut: sparse nets often generalizes better

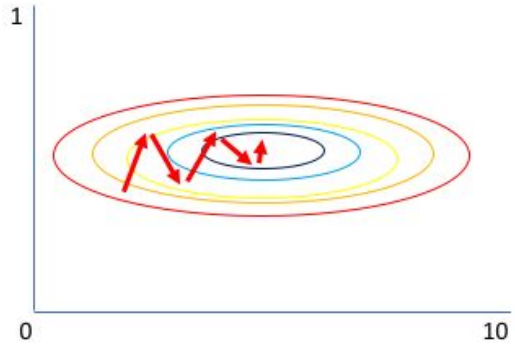
Regularizers

- DropOut
- L1 e L2: large weights are penalized, restricting them to close to zero numbers.

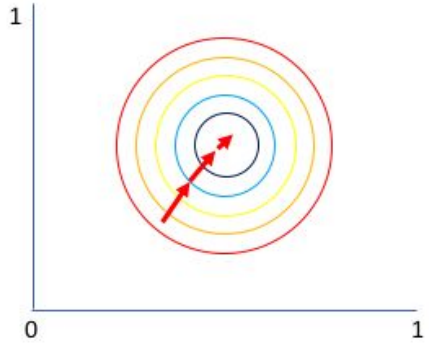
Regularizers

- DropOut
- L1 e L2
- Augmentation: the dataset is virtually enlarged with digital imaging processing

Normalization and Pre-training



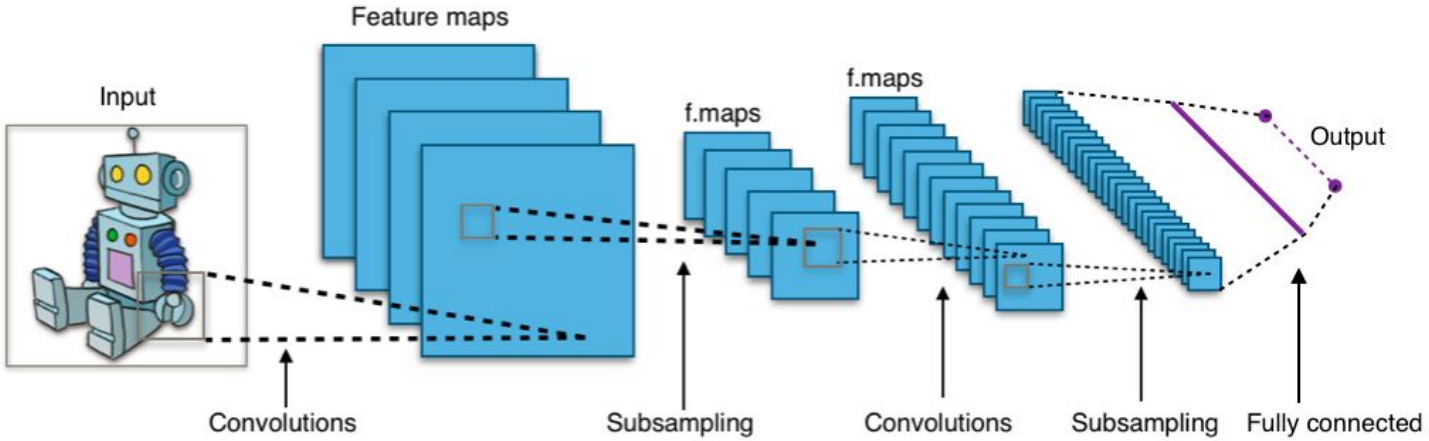
Gradient of larger parameter dominates the update

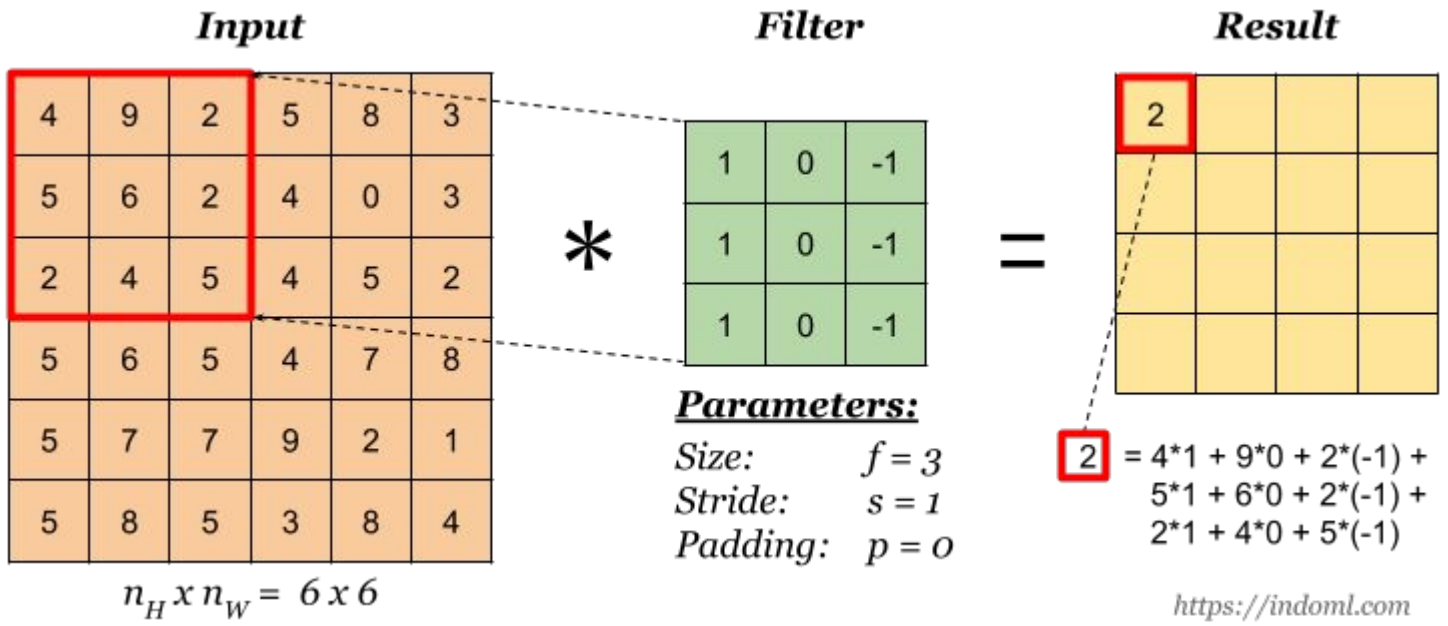


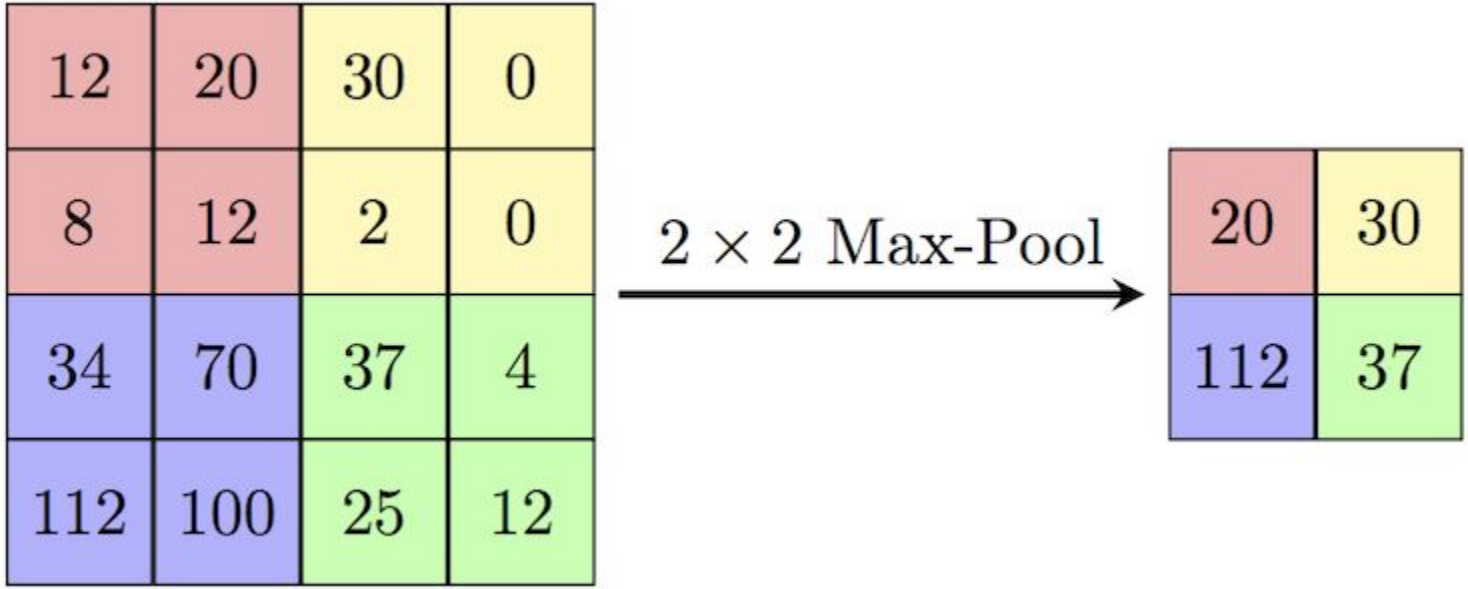
Both parameters can be updated in equal proportions

Convolutional Neural Networks

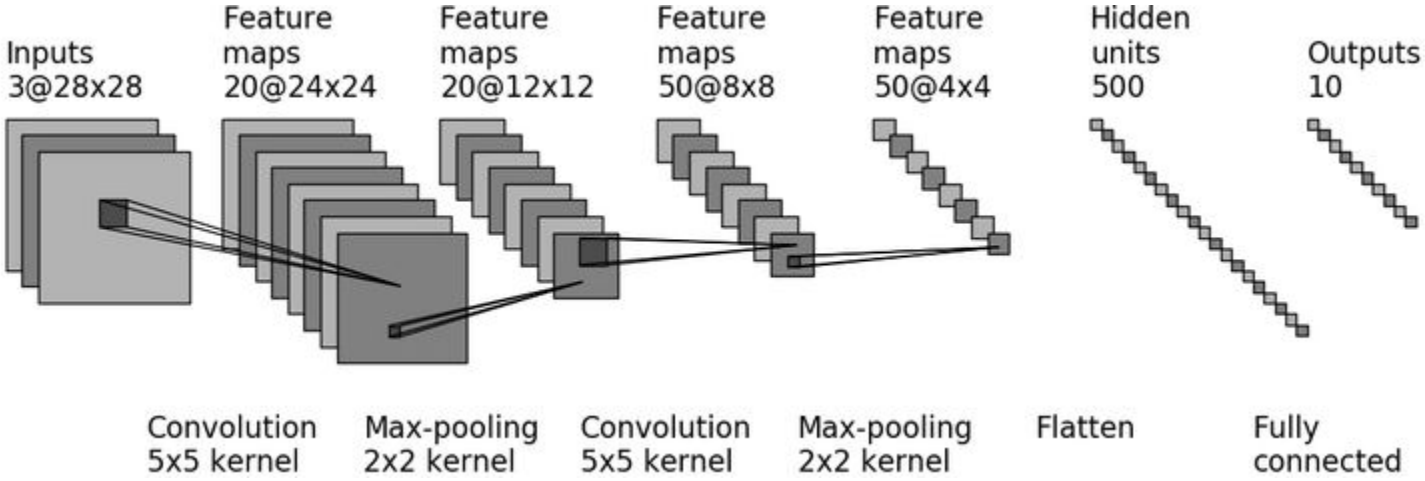
- Constrained application domain
- Embedded knowledge (constrained domain) helps to outperforms generic networks such as MLP
- kernels and feature maps
- Automatic feature extractor (it can handle any classifier on top of layers stack)


Input
Feature maps
f.maps
f.maps
Output
Convolutions
Subsampling
Convolutions
Subsampling
Fully connected

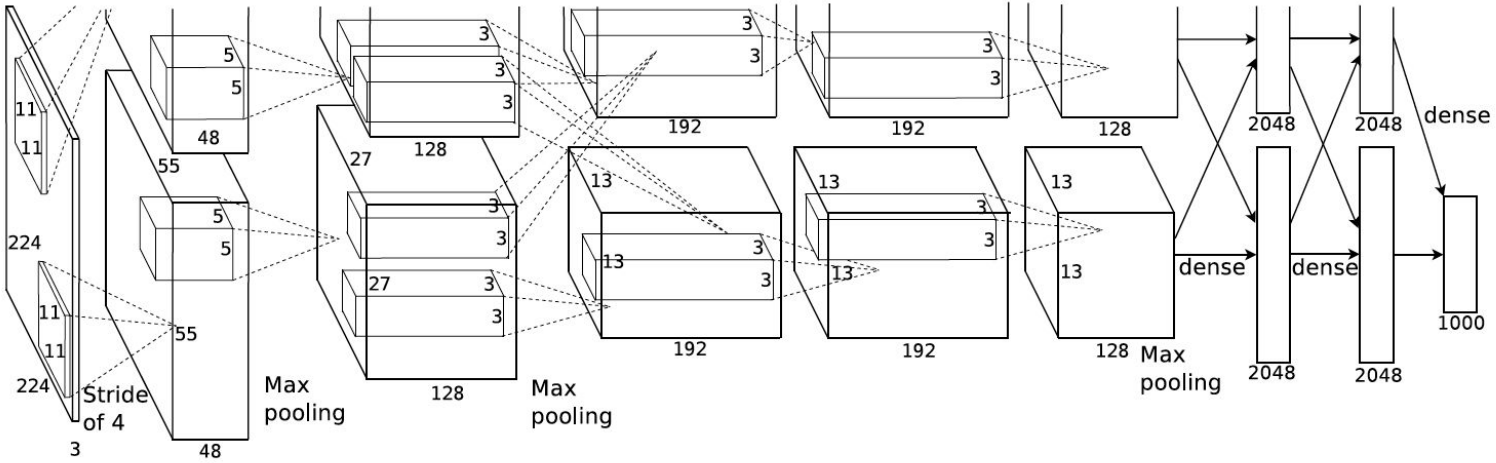

Input
Filter
Result
4 9 2 5 8 3
5 6 2 4 0 3
2 4 5 4 5 2
5 6 5 4 7 8
5 7 7 9 2 1
5 8 5 3 8 4
*
1 0 -1
1 0 -1
1 0 -1
=
2
Parameters:
Size: f = 3
Stride: s = 1
Padding: p = 0
2 = 4*1 + 9*0 + 2*(-1) +
5*1 + 6*0 + 2*(-1) +
2*1 + 4*0 + 5*(-1)
$n_H \times n_W = 6 \times 6$
https://indoml.com

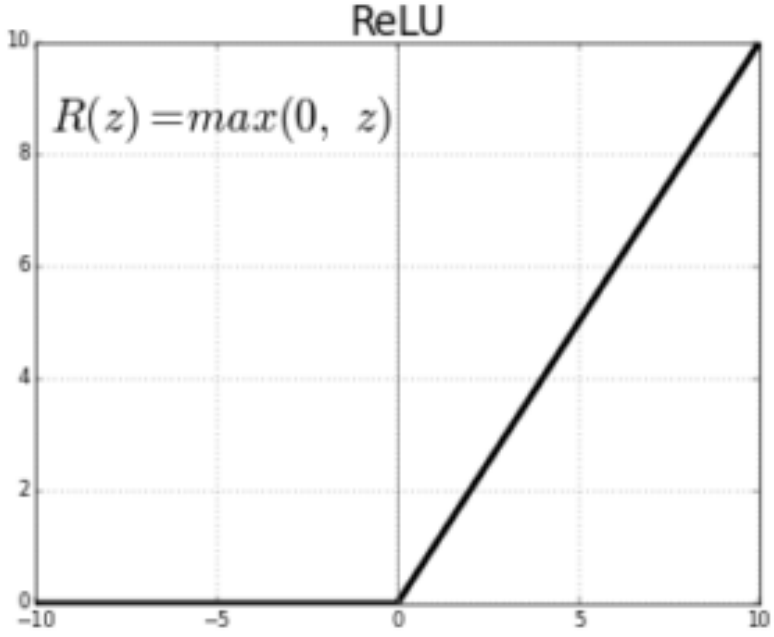

12 20 30 0
8 12 2 0
34 70 37 4
112 100 25 12
2 × 2 Max-Pool
20 30
112 37

LeNet


Inputs
3@28x28
Feature
maps
20@24x24
Feature
maps
20@12x12
Feature
maps
50@8x8
Feature
maps
50@4x4
Hidden
units
500
Outputs
10
Convolution
5x5 kernel
Max-pooling
2x2 kernel
Convolution
5x5 kernel
Max-pooling
2x2 kernel
Flatten
Fully
connected

AlexNet

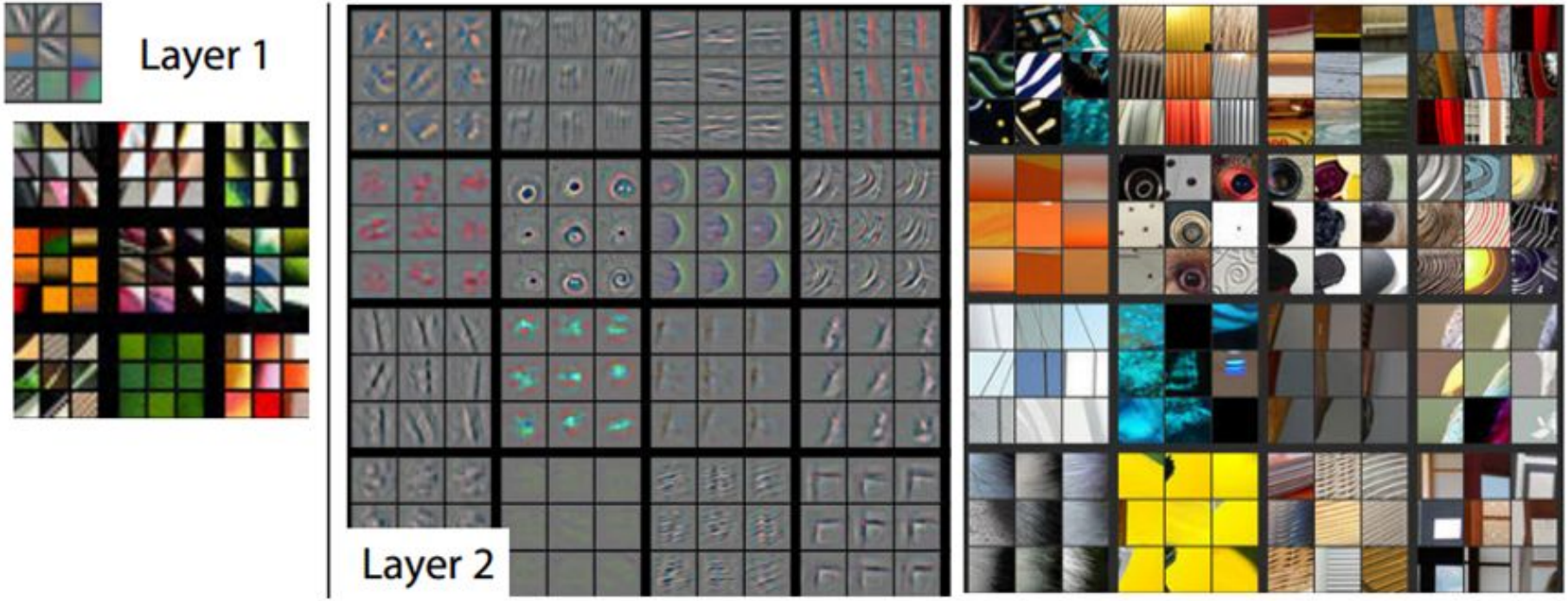

11
11
224
224
3
Stride
of 4
55
5
5
48
48
55
Max
pooling
27
5
5
128
3
3
3
27
3
3
128
Max
pooling
13
3
3
13
192
3
3
192
13
3
3
13
192
3
3
13
128
Max
pooling
dense
dense
2048
2048
2048
2048
dense
1000


ReLU
$R(z) = max(0,\ z)$
10
8
6
4
2
0
−10
−5
0
5
10

ResNet

Inception

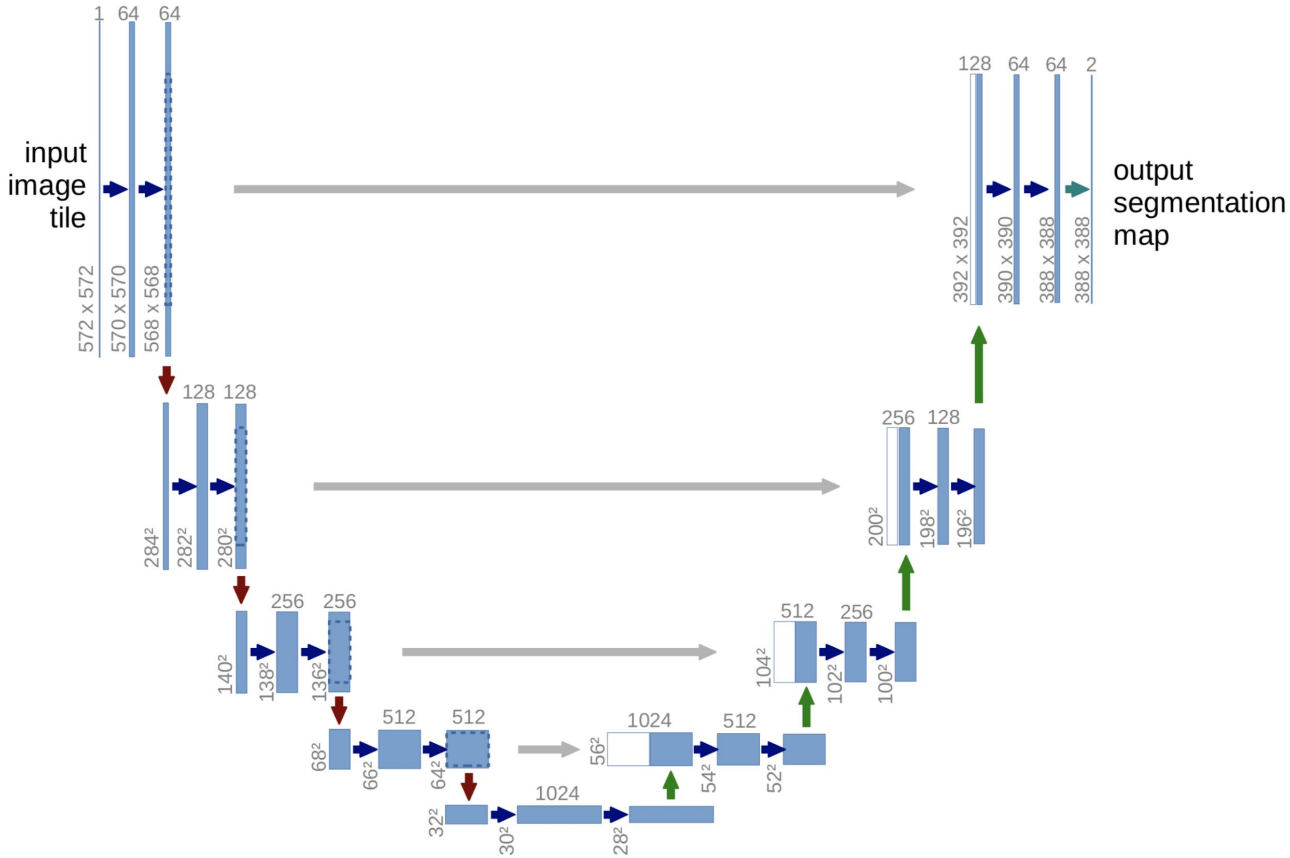
ResNext

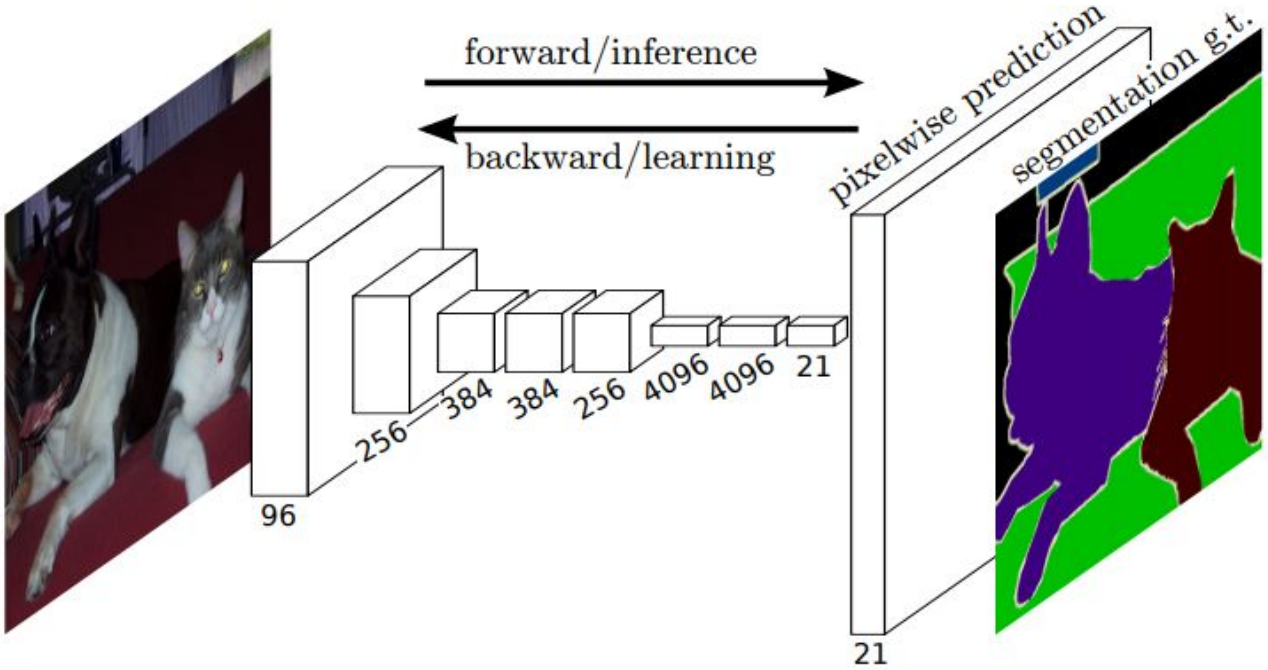


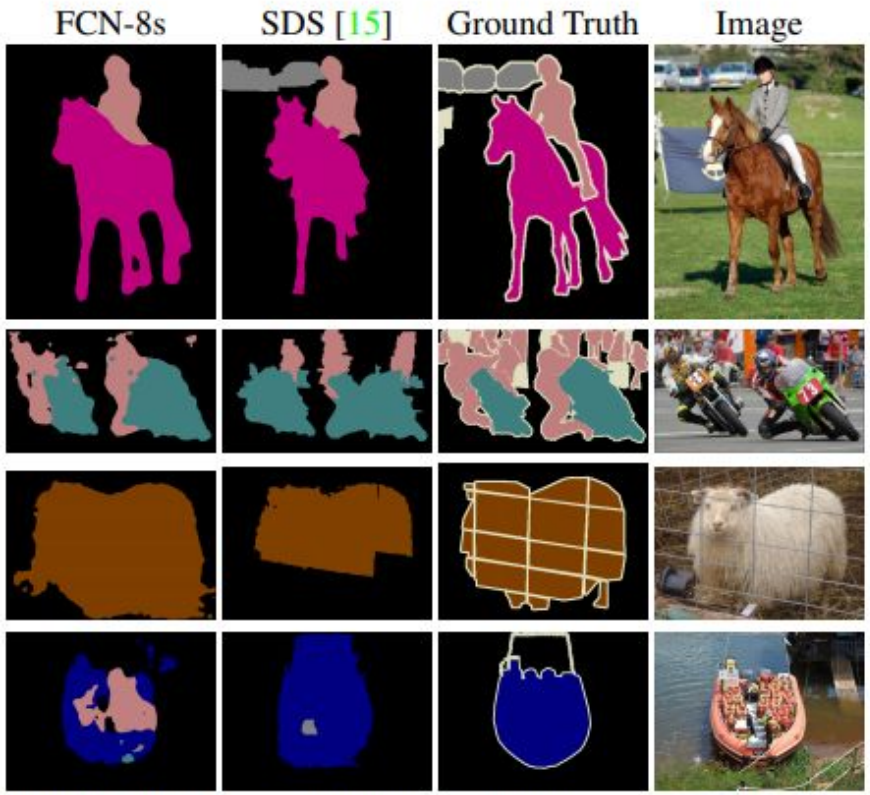
Visualizations of Layer 1 and 2. Each layer illustrates 2 pictures, one which shows the filters themselves and one that shows what part of the image are most strongly activated by the given filter. For example, in the space labled Layer 2, we have representations of the 16 different filters (on the left)

Segmentadores usando Fully convolutional NN

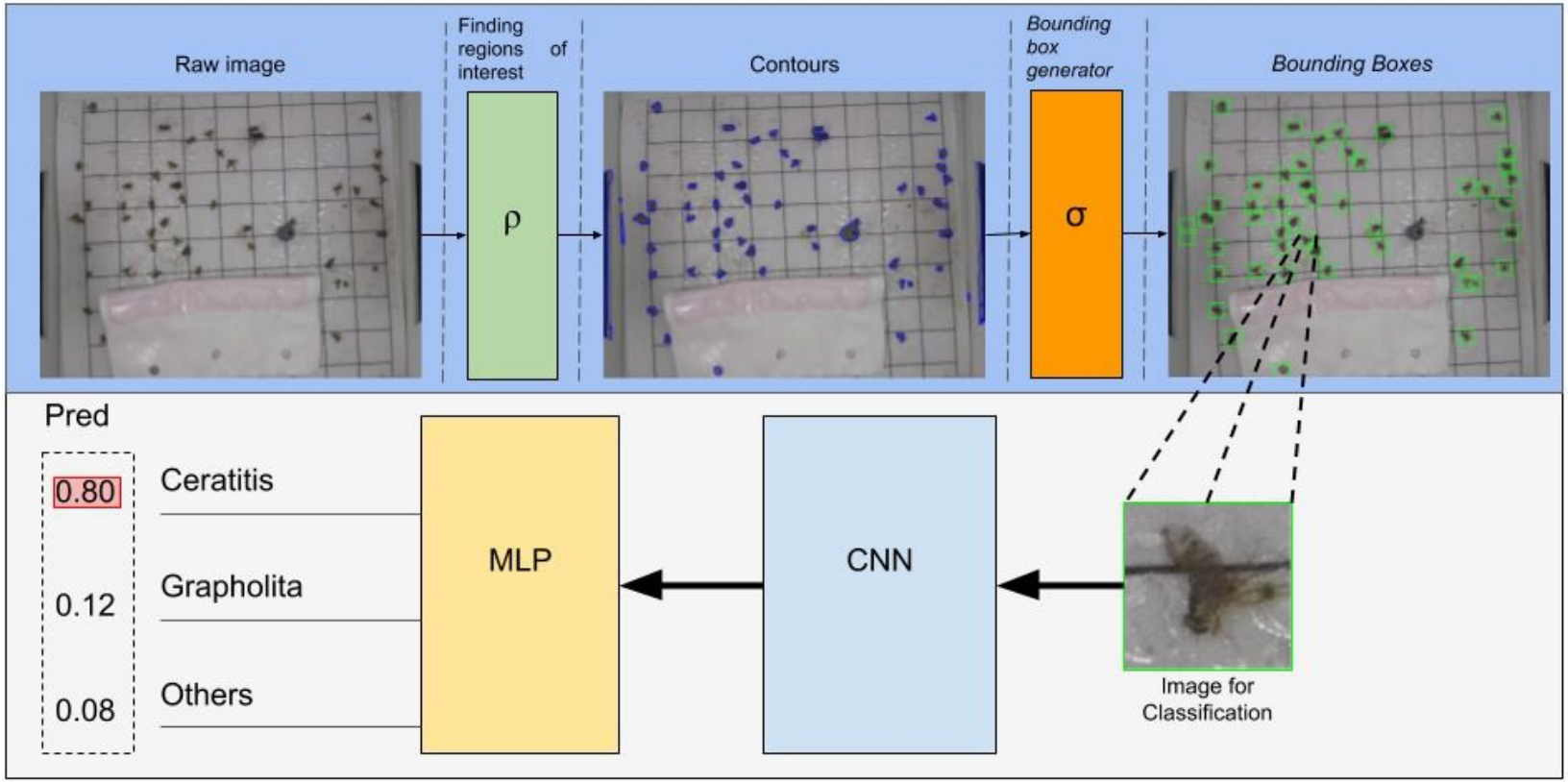
- Semantic segmentation (multiclass classification in a single image)
- It is a labor work to build a dataset for this task


input
image
tile
output
segmentation
map
1
64
64
572 x 572
570 x 570
568 x 568
128
128
284²
282²
280²
256
256
140²
138²
136²
512
512
68²
66²
64²
1024
32²
30²
28²
1024
512
56²
54²
52²
512
256
104²
102²
100²
256
128
200²
198²
196²
128
64
64
2
392 x 392
390 x 390
388 x 388
388 x 388


forward/inference
backward/learning
pixelwise prediction
segmentation g.t.
96
256
384
384
256
4096
4096
21
21


FCN-8s
SDS [15]
Ground Truth
Image

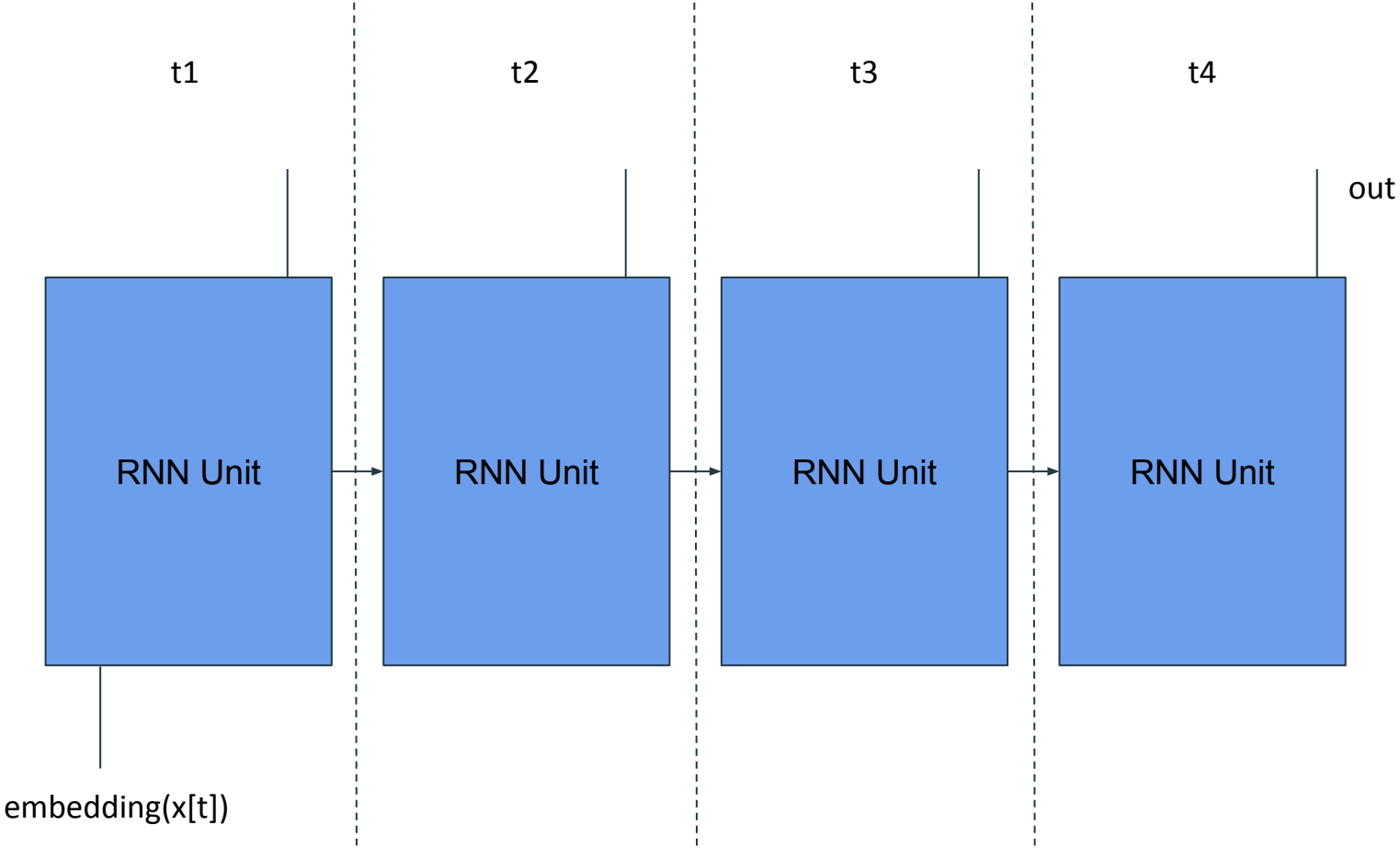
R-based CNN



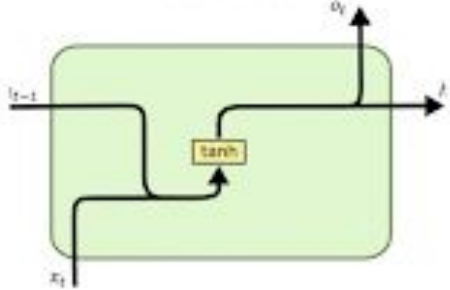
Martins, Valter A. M.; Freitas, Lucas C.; *et al*

Recurrent Neural Networks

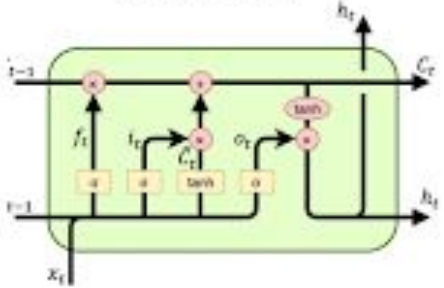
- applied to time-series data


t1
t2
t3
t4
out
RNN Unit
RNN Unit
RNN Unit
RNN Unit
embedding(x[t])

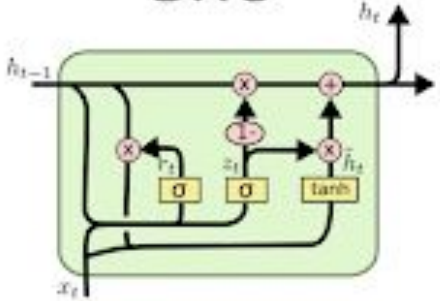
RNN



LSTM

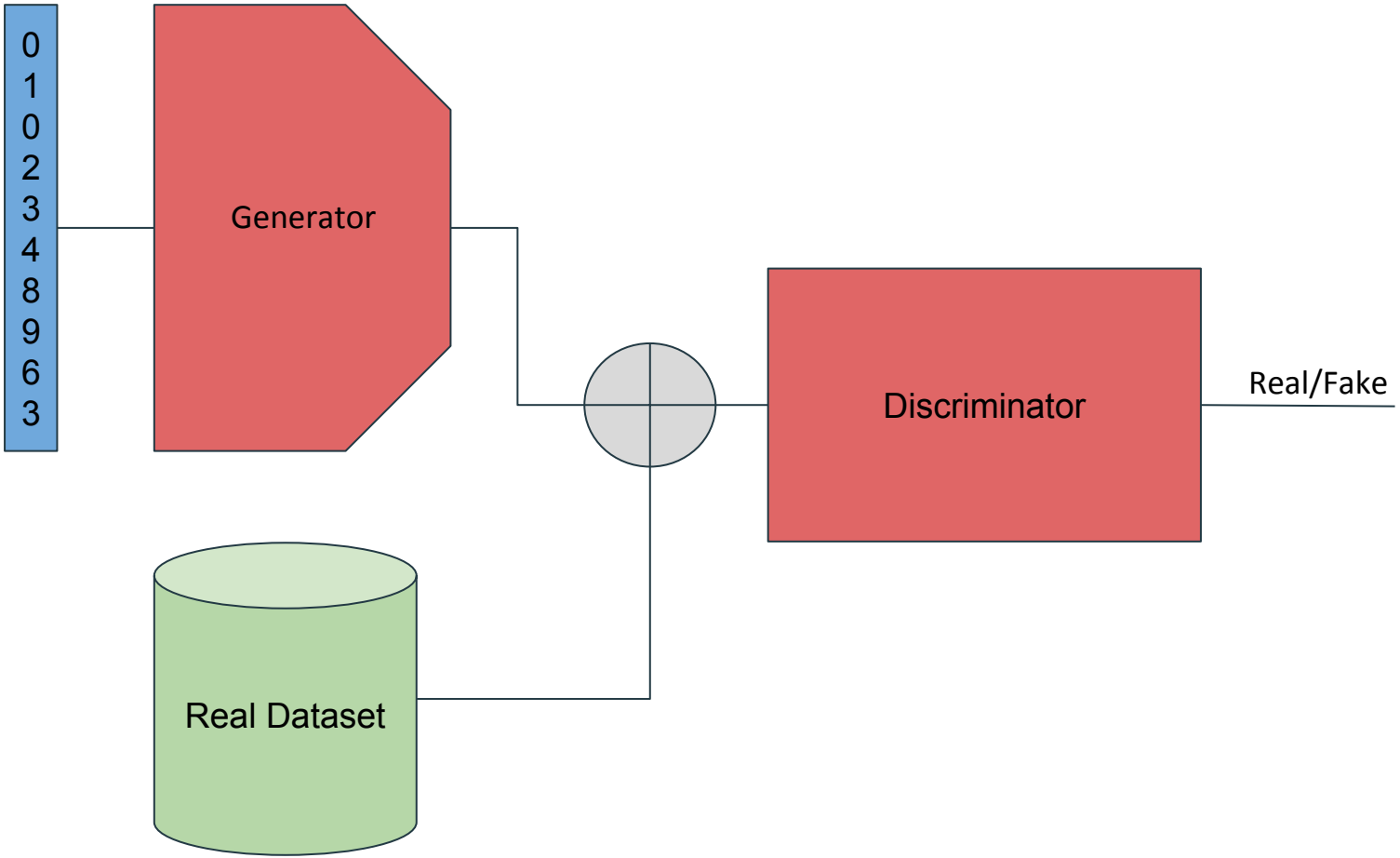


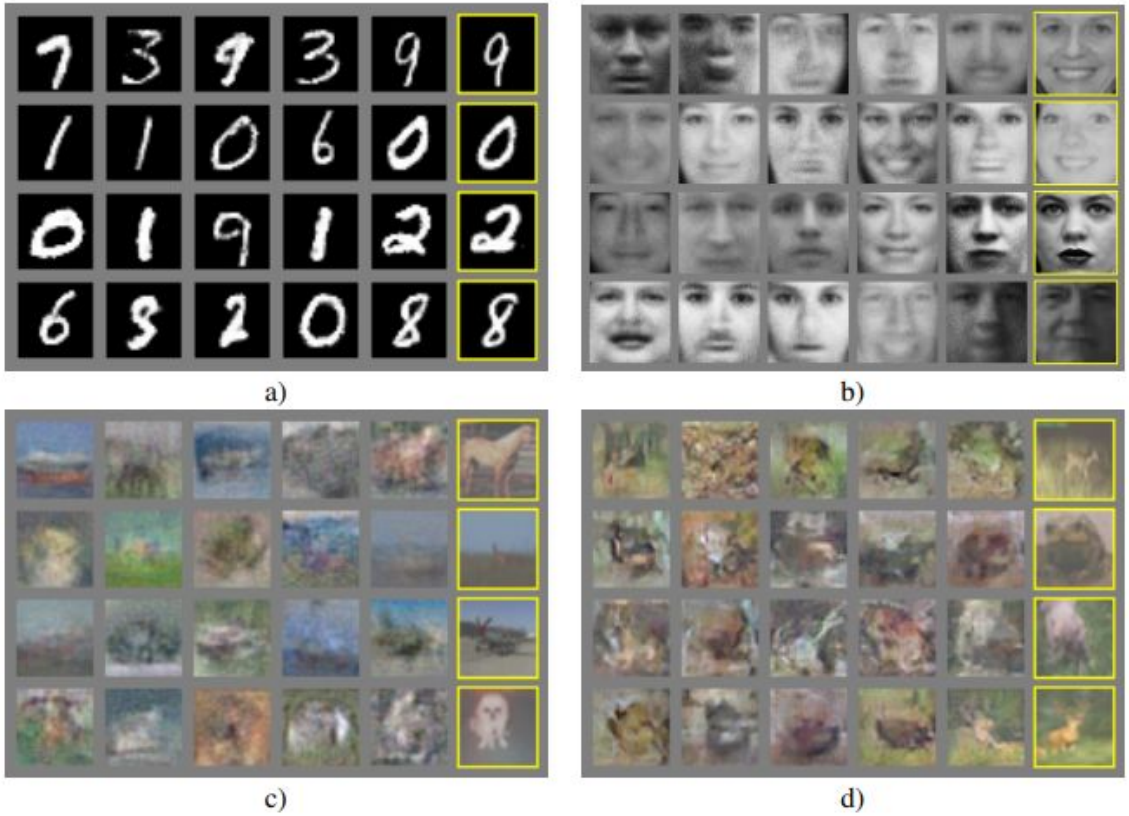
GRU



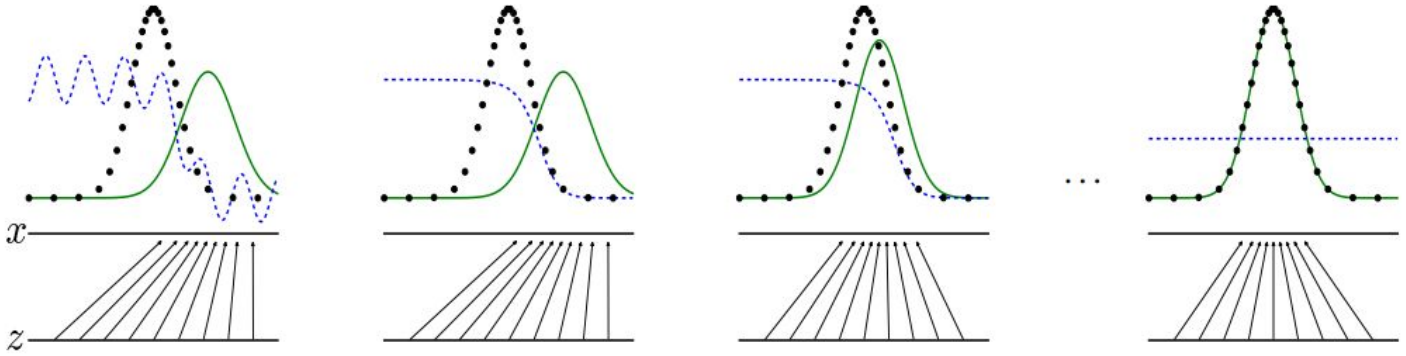
Generative Adversarial Networks

- Applications
- Hard convergence
- Generates artificial datasets


0
1
0
2
3
4
8
9
6
3
Generator
Discriminator
Real/Fake
Real Dataset



Generative Adversarial Networks; Goodfellow, Ian.



Generative Adversarial Networks; Goodfellow, Ian.

thispersondoesnotexit.com

Interesting Links

https://affinelayer.com/pixsrv/ : adding fancy to your sketch

https://colab.research.google.com/github/tensorflow/hub/blob/master/examples/colab/biggan_generation_with_tf_hub.ipynb#scrollTo=HuCO9tv3IKT2 : interpolation of a dog and a bath

https://www.thispersondoesnotexist.com/ : sort of people who don't exist

https://www.youtube.com/watch?v=MwtVkPKx3RA : Death Metal generated by a RNN

https://www.youtube.com/watch?v=cQ54GDm1eL0 : Fake Obama

Interesting Links

https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

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

https://www.nature.com/articles/s41467-019-10168-2.pdf

https://arxiv.org/pdf/1811.12808.pdf

https://jalammar.github.io/illustrated-word2vec/

References

RUDER, Sebastian. An overview of gradient descent optimization algorithms. **arXiv preprint arXiv:1609.04747**, 2016.

LONG, Jonathan; SHELHAMER, Evan; DARRELL, Trevor. Fully convolutional networks for semantic segmentation. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2015. p. 3431-3440.LONG, Jonathan; SHELHAMER, Evan; DARRELL, Trevor. Fully convolutional networks for semantic segmentation. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. 2015. p. 3431-3440.

Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.

LECUN, Yann et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278-2324, 1998.

LECUN, Yann et al. Backpropagation applied to handwritten zip code recognition. **Neural computation**, v. 1, n. 4, p. 541-551, 1989.

References

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep learning**. MIT press, 2016.