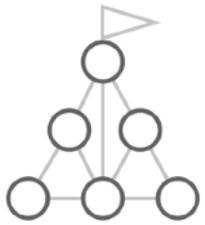


Deep Learning - Introduction

Itthi Chatnuntawech

Artificial Intelligence

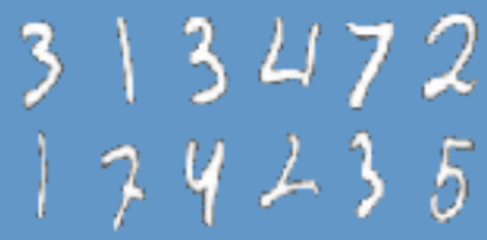
Any technique that enables computers to mimic human behavior

Machine Learning

Ability to learn without explicitly being programmed

Deep Learning

Learn underlying features in data using neural networks



1950's 1960's 1970's 1980's 1990's 2000's 2010's

Fasting blood sugar test → x mg/dL

normal if $x < 100$

prediabetes if $100 \leq x \leq 125$

diabetes if $125 < x$

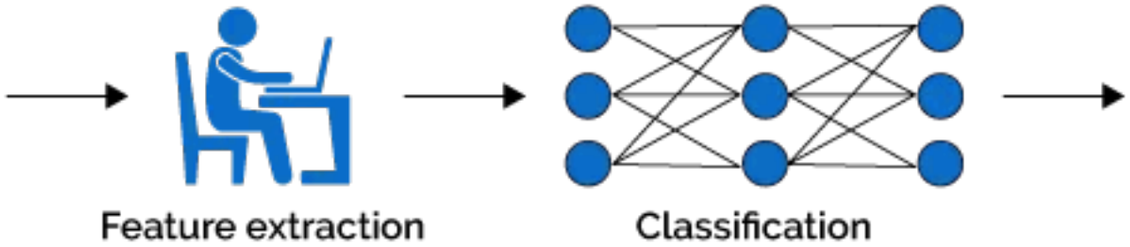
MIT 6.S191 Introduction to Deep Learning

keywords: feature extraction, classification task, classifier

Example: Watermelon or Orange?

Traditional Machine Learning

Input image

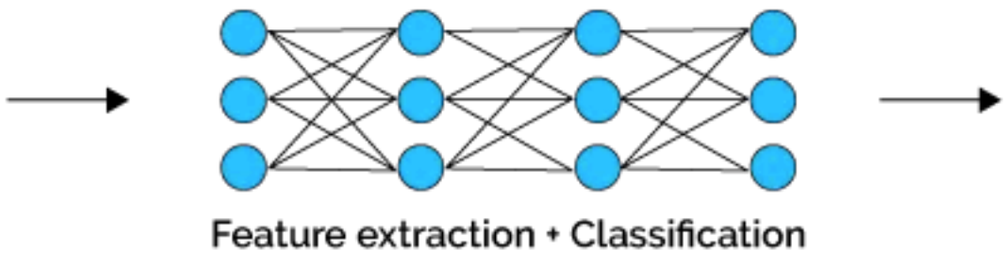


Output

watermelon

Deep Learning

Input image



Output

watermelon

Some parts taken from: Azua Tech

Watermelon or Orange?

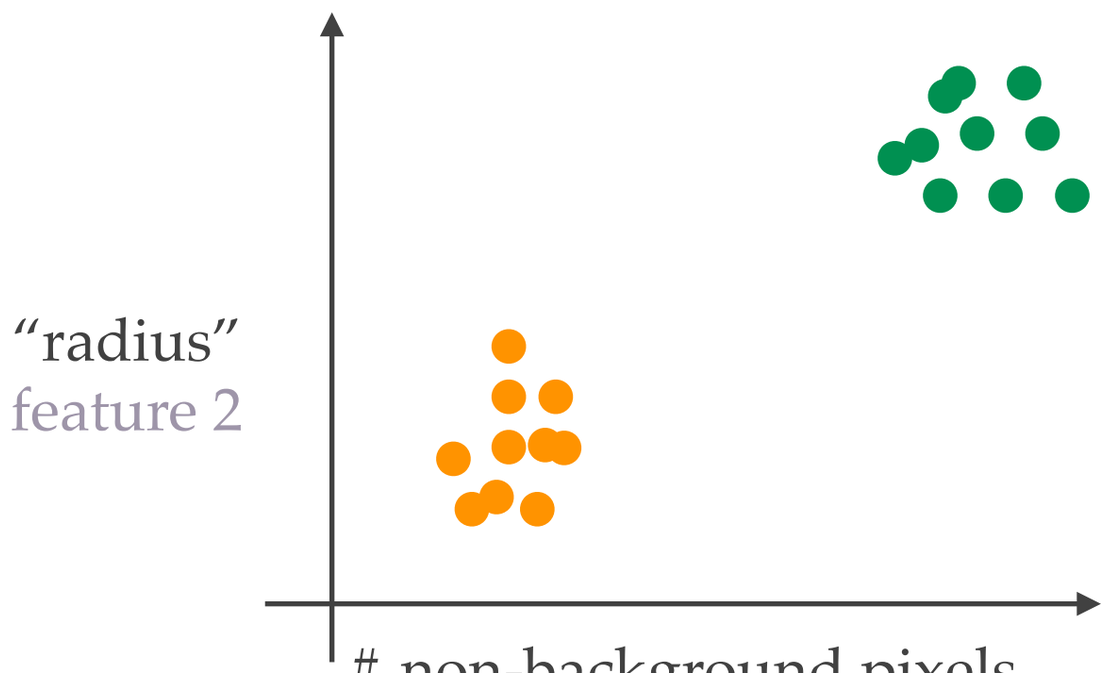
Traditional Machine Learning

Input image

Output

watermelon

Phase: Training



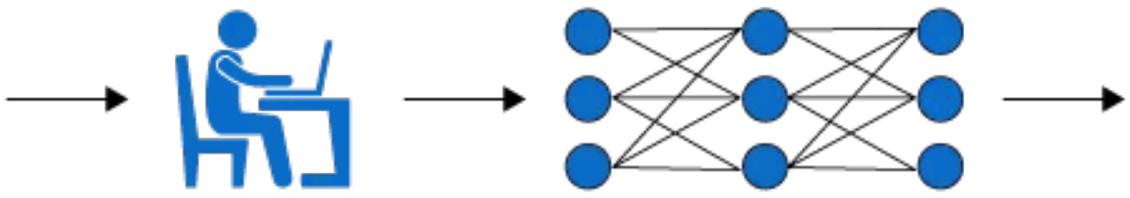
current sample

keywords: feature extraction, feature engineering, handcrafted/hand-designed features, feature space, descriptors

Watermelon or Orange?

Traditional Machine Learning

Input image

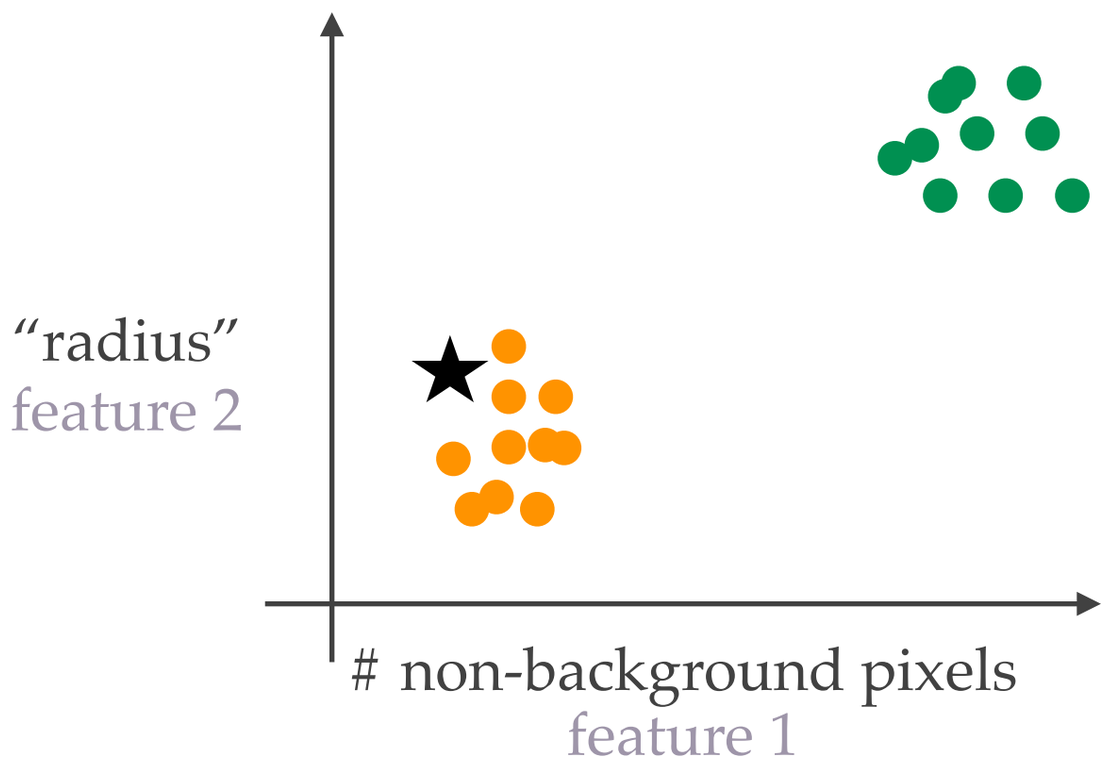


Output

watermelon

Phase: Testing

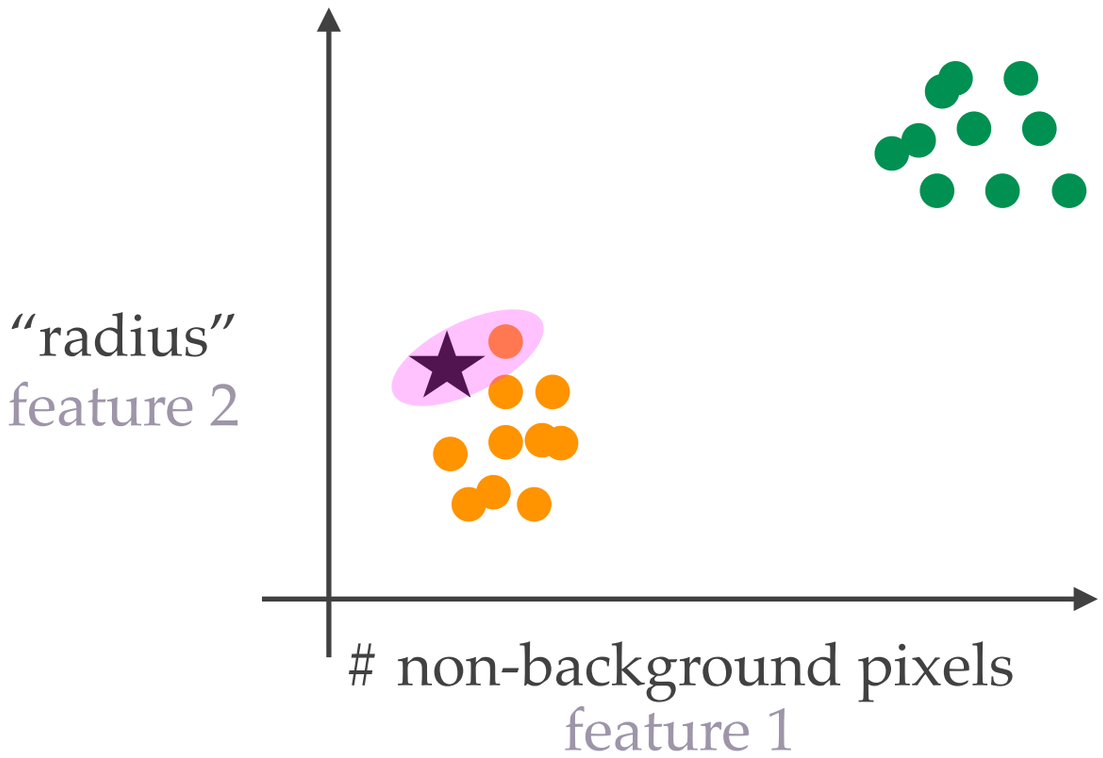
current sample



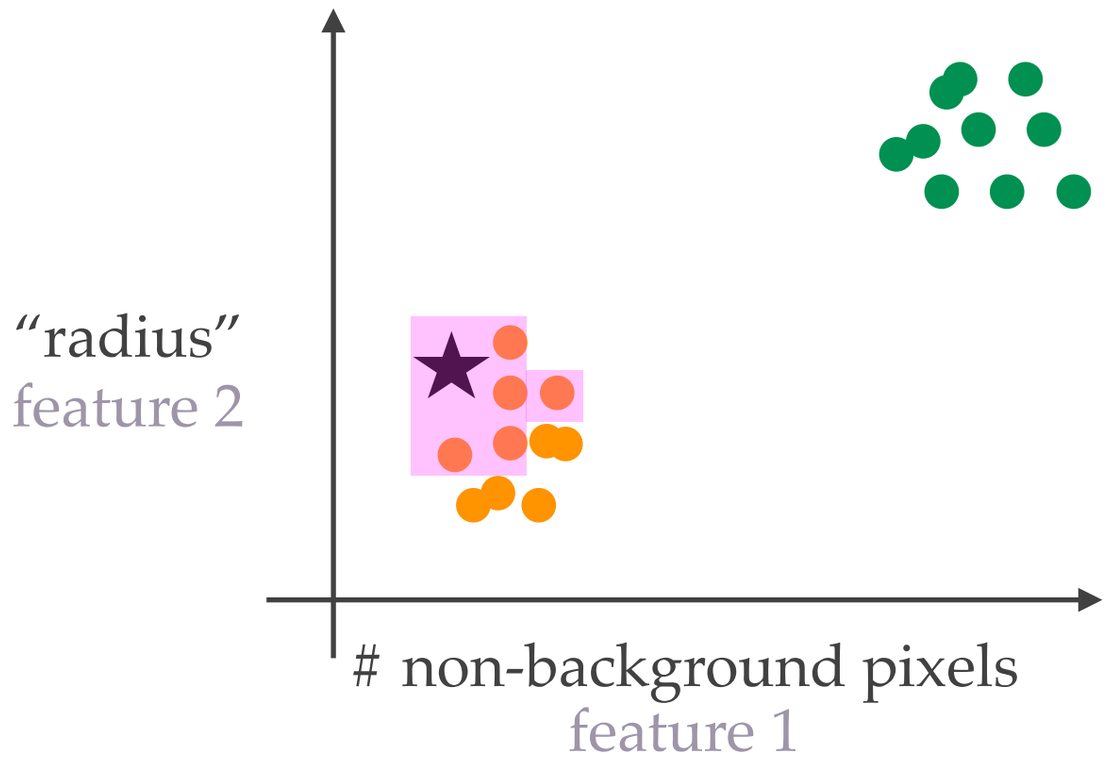
Orange!

keyword: decision boundary

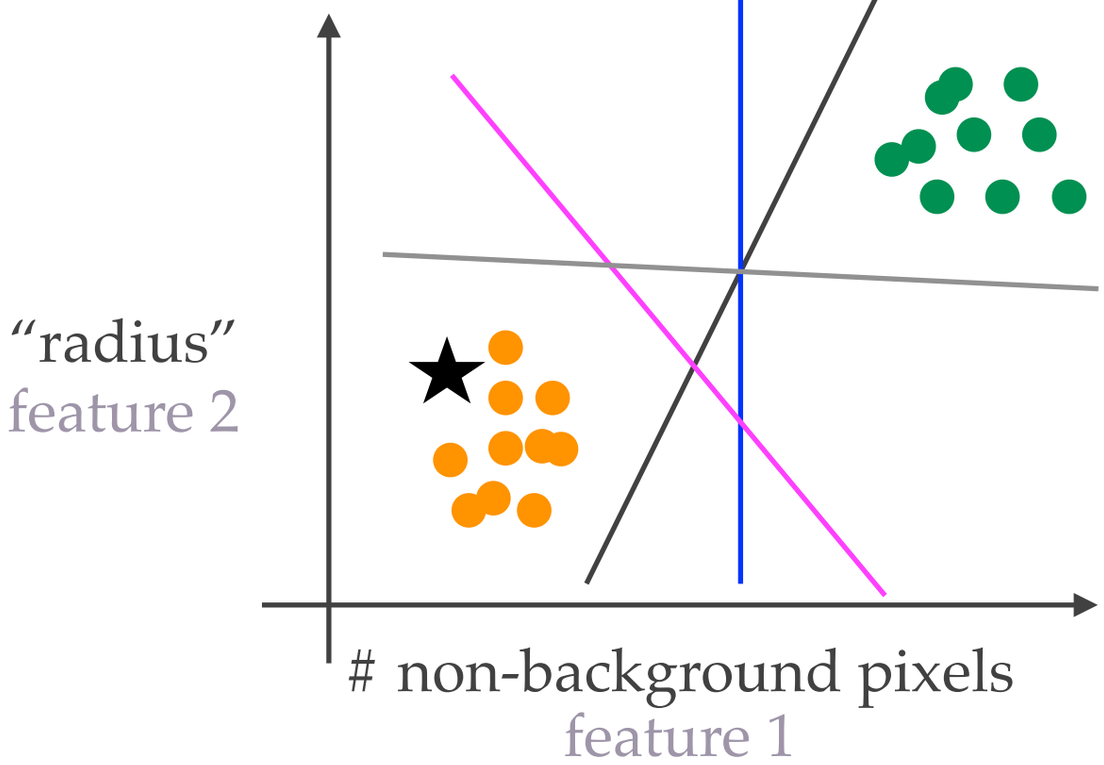
Nearest Neighbor



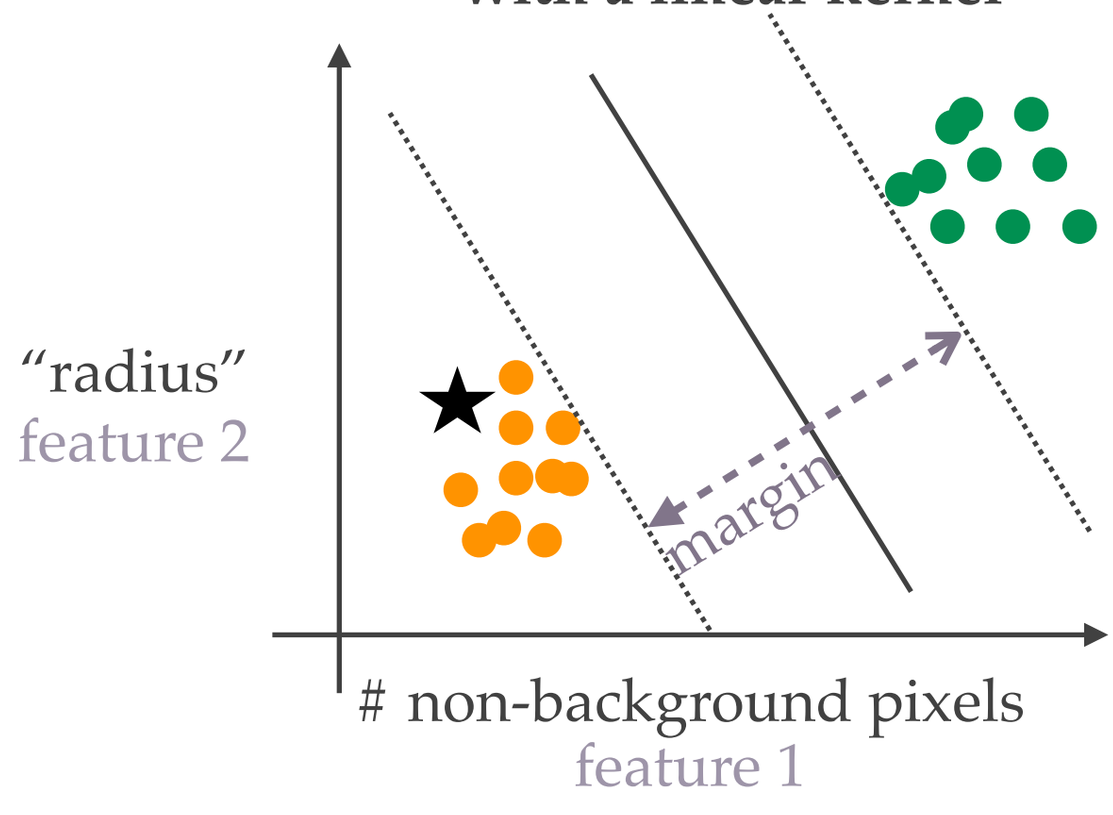
k-Nearest Neighbors (kNN)



Linear Classifier



Support Vector Machine (SVM) with a linear kernel



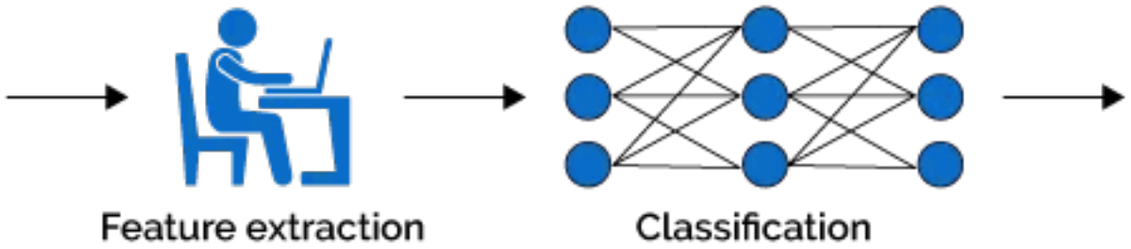
Watermelon or Orange?

Traditional Machine Learning

Input image

hand-engineered / hand-crafted features

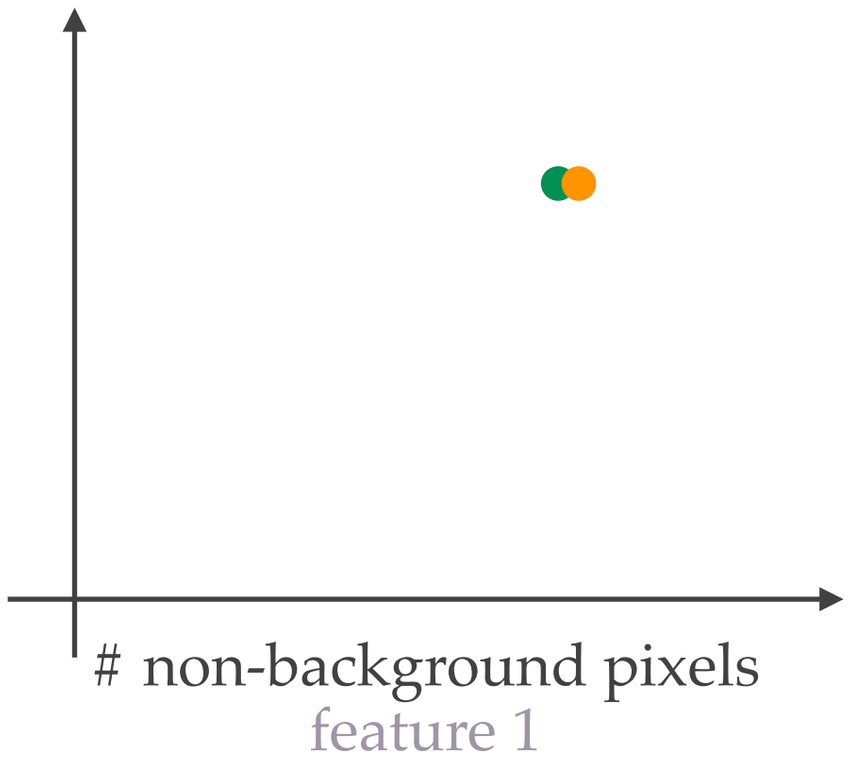
random forest, SVM, kNN, XGBoost, LGBM



Output

watermelon

Problem with feature engineering?

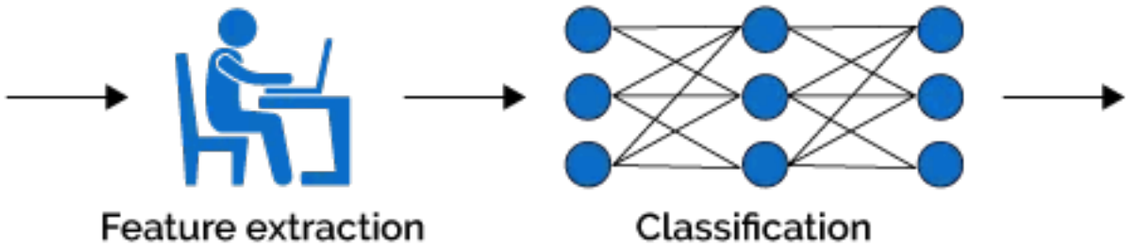


Some parts taken from: Azua Tech

Watermelon or Orange?

Traditional Machine Learning

Input image

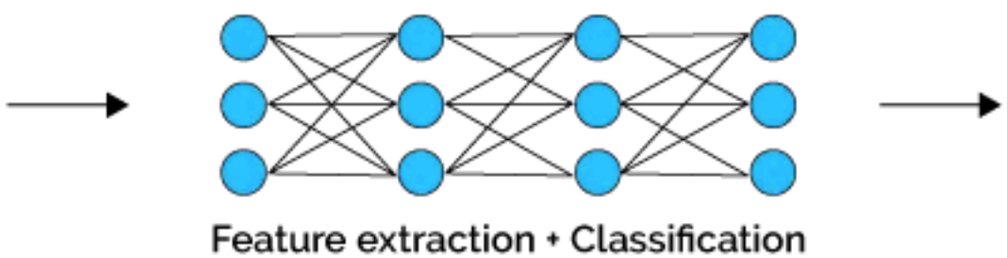


Output

watermelon

Deep Learning

Input image



Output

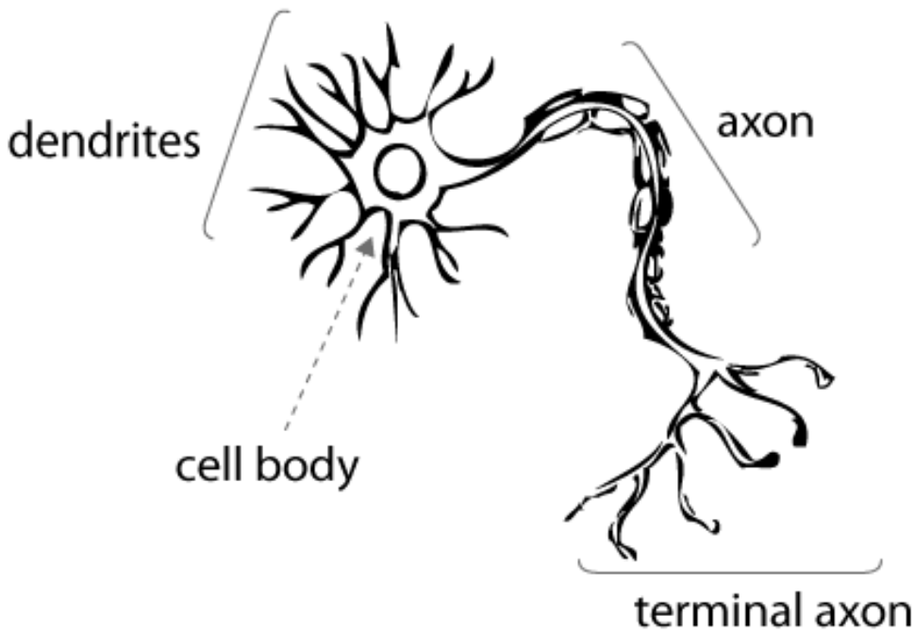
watermelon

Some parts taken from: Azua Tech

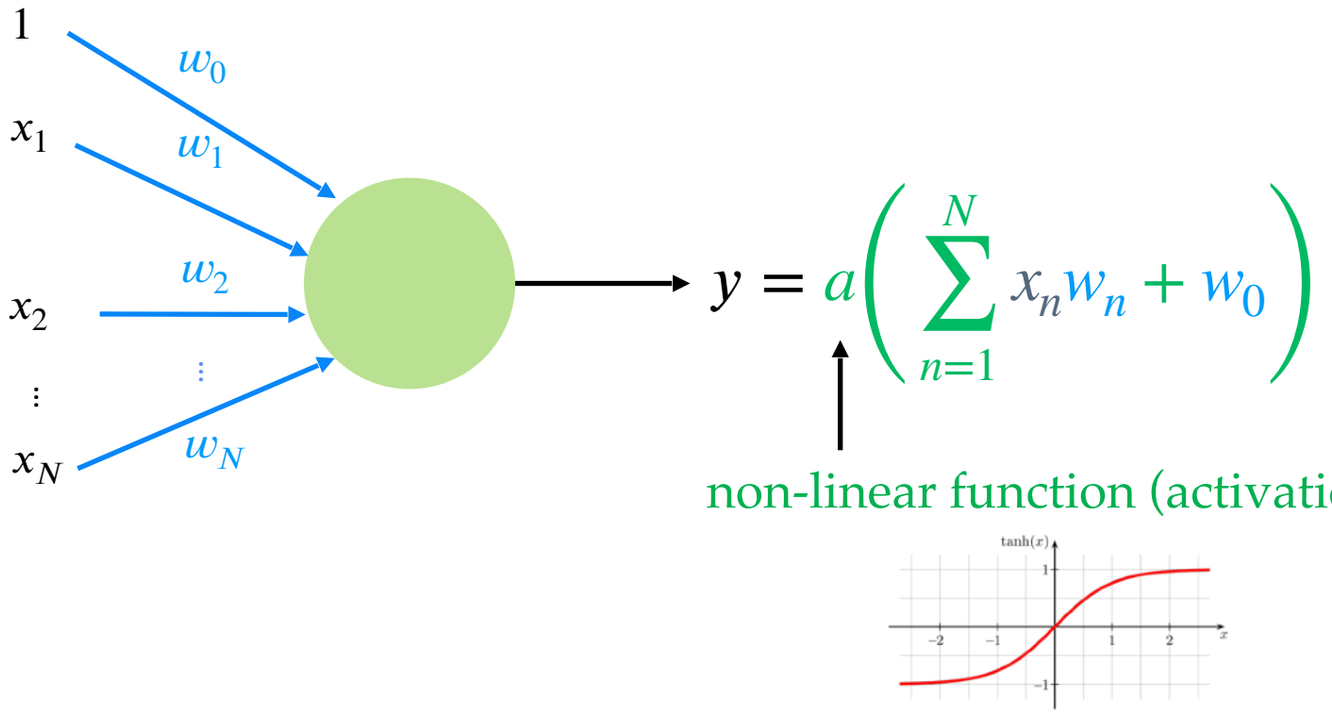
Deep Learning

- Deep learning is a subfield of machine learning that stems from artificial neural networks (ANN)

Single neuron



Artificial neuron



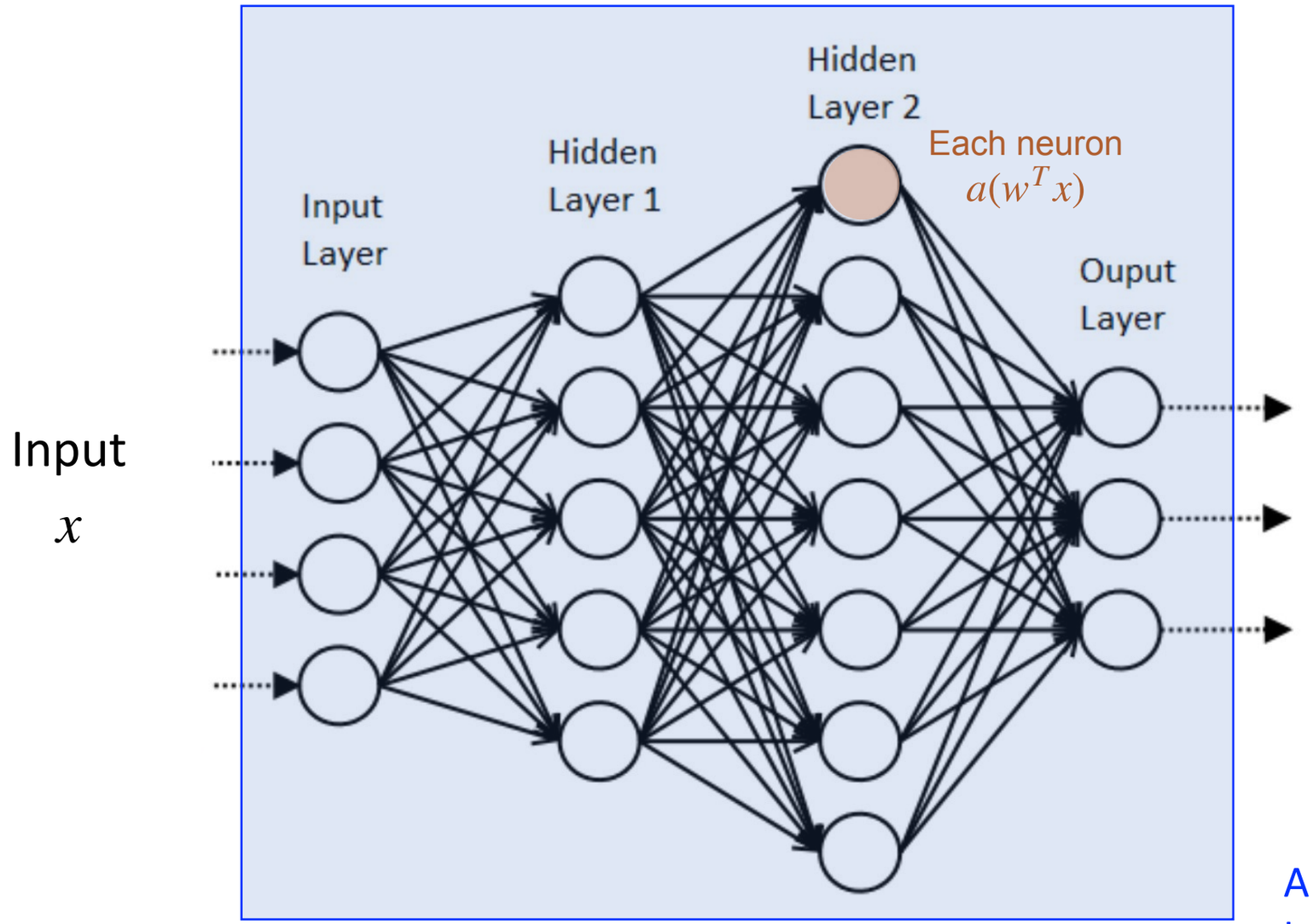
If we set all the weights to 0, then $y = 0$.
If we set $w_1 = 1$ and the rest to 0, then $y = a(x_1)$.
If we set $w_0 = 0$ and the rest to 1, then $y = a(x_1 + x_2 + \ldots + x_N)$.

Different weights give rise to different behavior

Source: The Essense of Artificial Neural Networks by Ivan Liljeqvist (Medium)

Deep Learning

- An artificial neural network with many hidden layers is called a deep artificial neural network (ANN)



$$y = f_W(x)$$

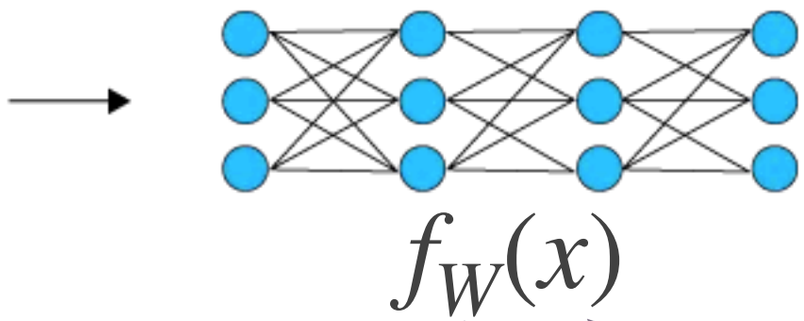
A function that can change behavior depending on the current value of W

Deep Learning

- Think of a deep NN as a universal function approximator

Image classification

Input image

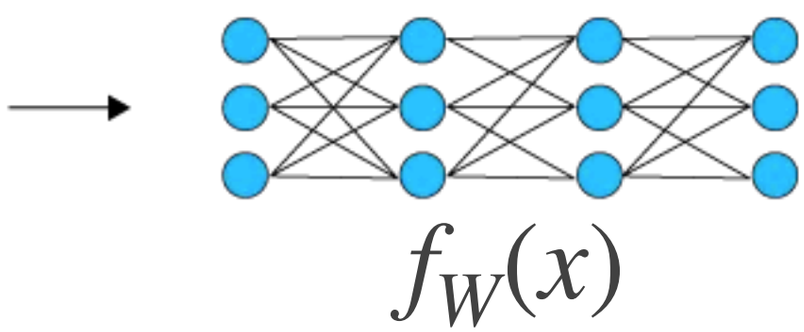


Output: Label

1 (watermelon)

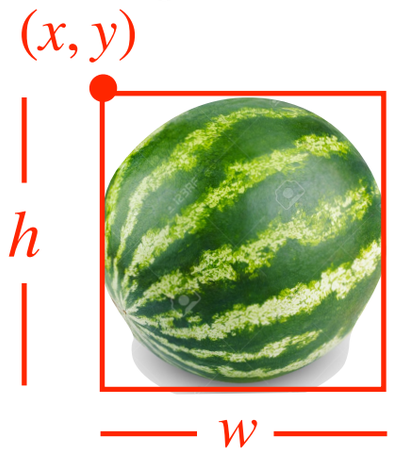
Object detection

Input image



Output: rectangular box

$$\begin{bmatrix} x \\ y \\ w \\ h \end{bmatrix}$$



Fully connected layer
(Dense)

Convolutional layer

Conv1D, 2D, 3D, …
separable Conv

Optimizer

SGD
Adam
RMSprop

Deep Learning

Pooling layer

max-pooling
average-pooling

Evaluation metric

accuracy
F1-score
AUC
confusion matrix

Activation function

sigmoid
softmax
ESP (swish)
ReLU

Loss function

categorical crossentropy
binary crossentropy
mean squared error
mean absolute error

Regularization

Dropout
Data augmentation
l_1, l_2 regularizations

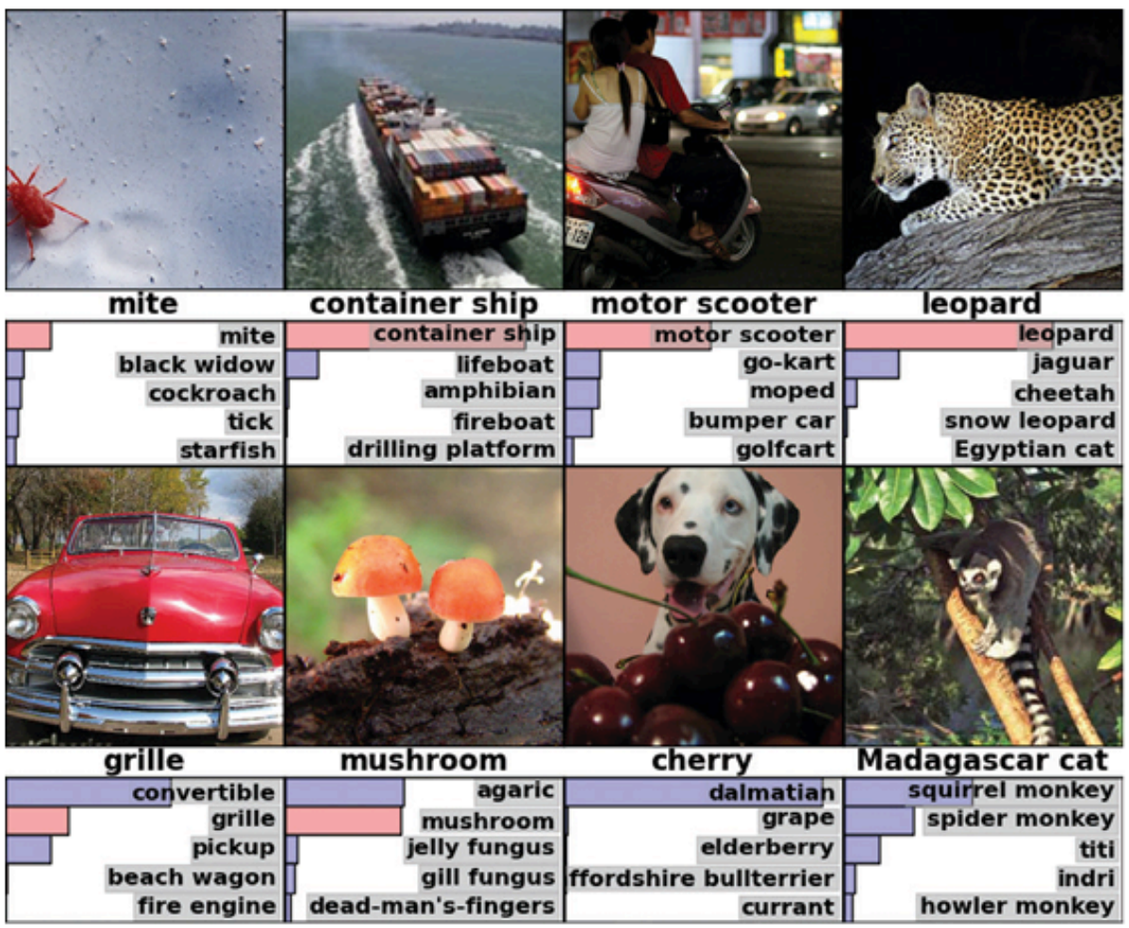
- Combine basic components to build a neural network
  - More components → "More" representative power

ImageNet

IMAGENET

1000 classes, 14,197,122 images

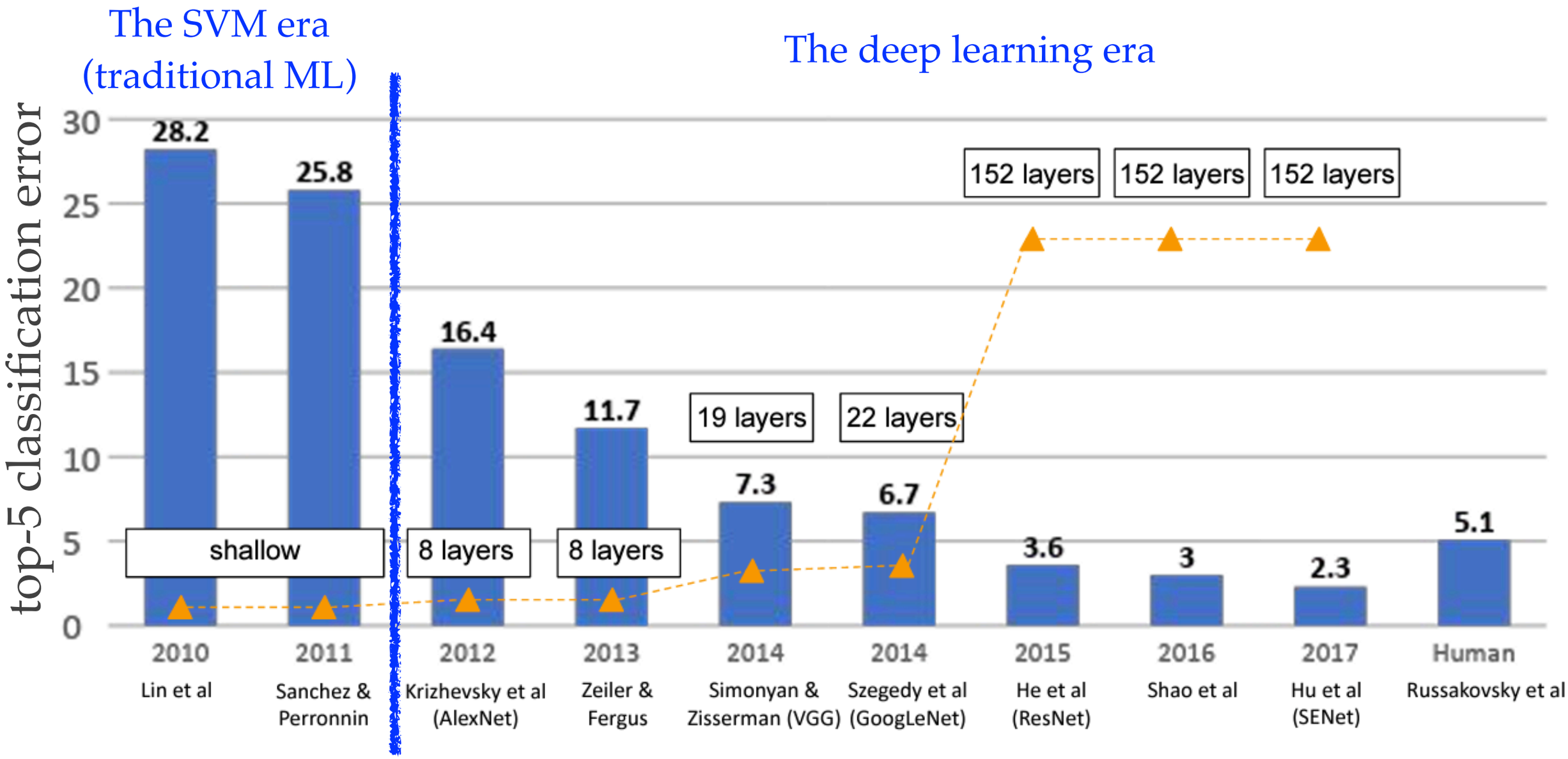
Figure 4. (Left) Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). (Right) Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.



Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems 25 (2012).

ImageNet

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



Supervised Model Training

Step 1: Create a neural network with some initial weight f_{W_0}

Neural network

Training iteration 0

f_{W_0}

Supervised Model Training

Step 2: Prepare a dataset which is a collection of input-output pairs

Prepared inputs
x

Prepared outputs
(true labels)
y

1

0

0

1

Neural network

Training iteration 0
f_{W_0}

orange: 0
watermelon: 1

Supervised Model Training

Step 3: Pass the prepared inputs to the network

Prepared inputs
x

Neural network

Training iteration 0
f_{W_0}

Estimated outputs
$\hat{y} = f_{W_0}(x)$

Prepared outputs (true labels)
y

Estimated outputs $\hat{y} = f_{W_0}(x)$	Prepared outputs (true labels) y
0.6	1
0.6	0
0.5	0
0.5	1

orange: 0
watermelon: 1

Supervised Model Training

Step 4: Compare $\hat{y}$ to y and modify the weights of the neural network to make $\hat{y}$ approach y using the backpropagation algorithm

Prepared inputs
x

A simple loss function: $L(y, \hat{y}) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$

$L(y, \hat{y}) = (0.6 - 1)^2 + (0.6 - 0)^2 + (0.5 - 0)^2 + (0.5 - 1)^2 = 1.02$

Estimated outputs
$\hat{y} = f_{W_0}(x)$

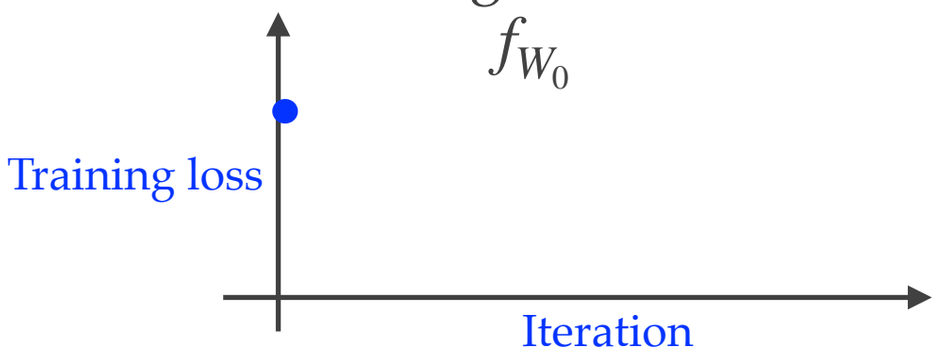
Prepared outputs
(true labels)
y

0.6

1

Neural network

Training iteration 0
f_{W_0}



0.6 0

0.5 0

0.5 1

orange: 0
watermelon: 1

Supervised Model Training

Repeat steps 3 and 4 to continuously improve the weights of the neural network

Prepared inputs
x

A simple loss function: $L(y, \hat{y}) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$

Estimated outputs
$\hat{y} = f_{W_1}(x)$

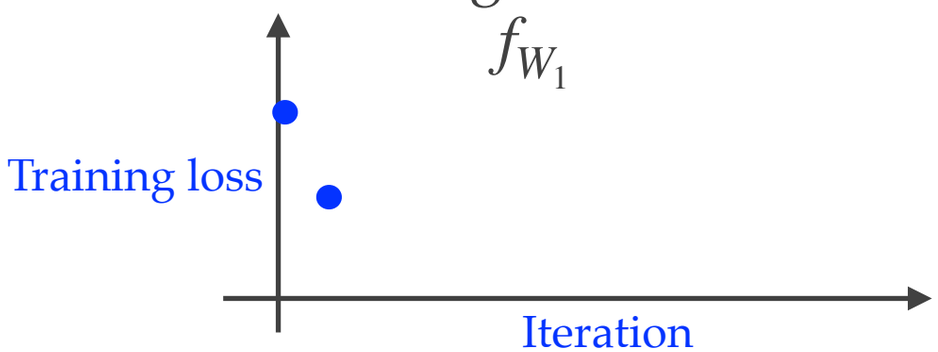
Prepared outputs (true labels)
y

$L(y, \hat{y}) = (0.6 - 1)^2 + (0.4 - 0)^2 + (0.4 - 0)^2 + (0.6 - 1)^2 = 0.64$

0.6 1

Neural network

Training iteration 1



0.4 0

0.4 0

0.6 1

orange: 0

watermelon: 1

Supervised Model Training

Repeat steps 3 and 4 to continuously improve the weights of the neural network

Prepared inputs
x

A simple loss function: $L(y, \hat{y}) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$

Estimated outputs
$\hat{y} = f_{W_2}(x)$

Prepared outputs (true labels)
y

$L(y, \hat{y}) = (0.7 - 1)^2 + (0.35 - 0)^2 + (0.4 - 0)^2 + (0.6 - 1)^2 = 0.5325$

0.7 — 1

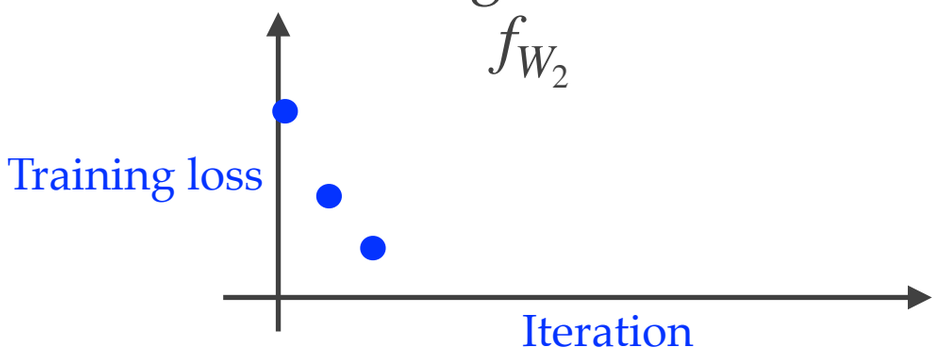
Neural network

0.35 — 0

0.4 — 0

Training iteration 2

f_{W_2}



0.6 — 1

orange: 0
watermelon: 1

Supervised Model Training

Repeat steps 3 and 4 to continuously improve the weights of the neural network

Prepared inputs
x

A simple loss function: $L(y, \hat{y}) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$

Estimated outputs
$\hat{y} = f_{W_{50}}(x)$

Prepared outputs (true labels)
y

$L(y, \hat{y}) = (0.8 - 1)^2 + (0.15 - 0)^2 + (0.1 - 0)^2 + (0.9 - 1)^2 = 0.0825$

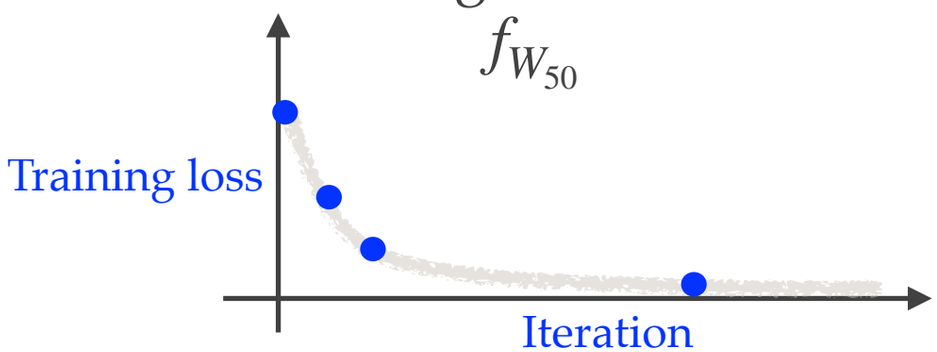
0.8 — 1

Neural network

0.15 — 0

0.1 — 0

Training iteration 50
$f_{W_{50}}$



0.9 — 1

orange: 0
watermelon: 1

Supervised Model Training

Repeat steps 3 and 4 to continuously improve the weights of the neural network

Prepared inputs

x

A simple loss function: $L(y, \hat{y}) = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$

Estimated outputs

$\hat{y} = f_{W_{1000}}(x)$

Prepared outputs (true labels)

y

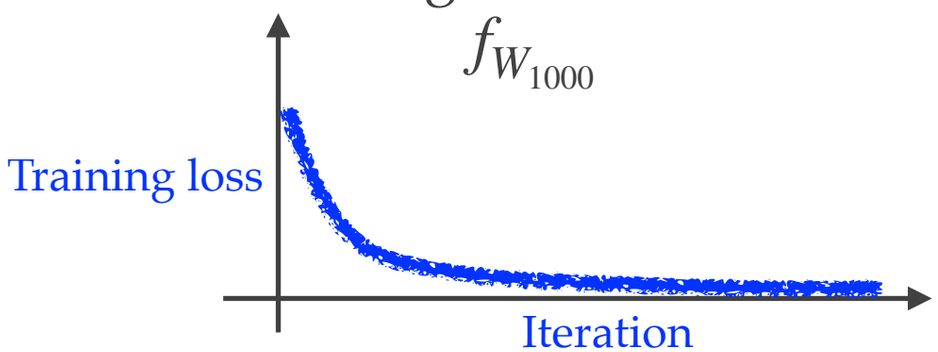
$L(y, \hat{y}) = (0.99 - 1)^2 + (0.05 - 0)^2 + (0.01 - 0)^2 + (0.97 - 1)^2 = 0.0036$

0.99 1

Neural network

Training iteration 1000

$f_{W_{1000}}$



0.05 0

0.01 0

0.97 1

orange: 0

watermelon: 1

keywords: overfitting, underfitting, regularization, early stopping, data augmentation, training/validation/test data

Test the Trained Model

Test image (unseen)

Trained neural network

Predicted Label

0.01

less than 0.5, so it is likely orange

orange: 0
watermelon: 1

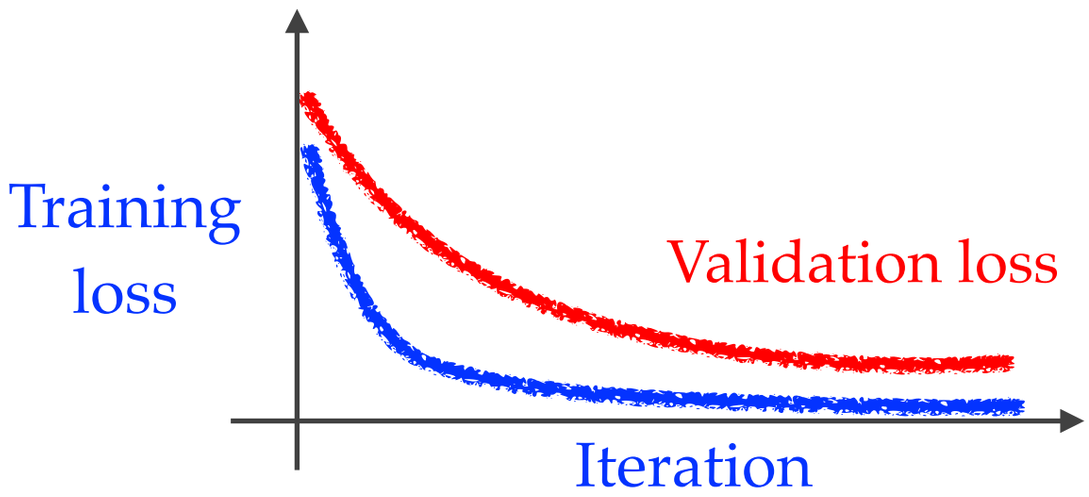
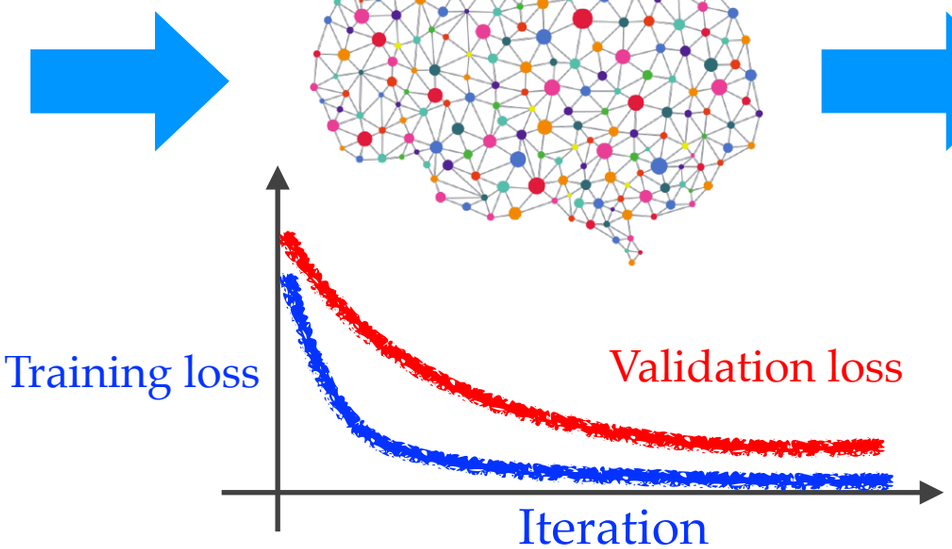


Image Classification Using Deep Learning

- Training phase: Optimize the weights of a deep neural network

Prepared inputs



Estimated outputs	Prepared outputs
0.7	1
0.2	0
0.1	0
0.9	1

- Test phase: Perform prediction using the trained neural network

0.01

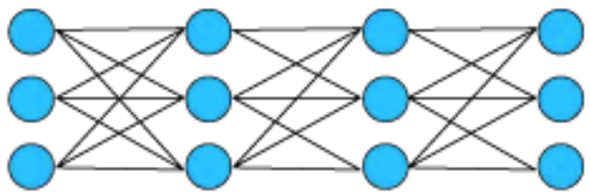
orange!

Deep Learning

- Train a deep neural network (NN) to perform some task from data
- Think of a deep NN as a universal function approximator

Image classification

Input: image



$f_W(x)$

Adjustable weight

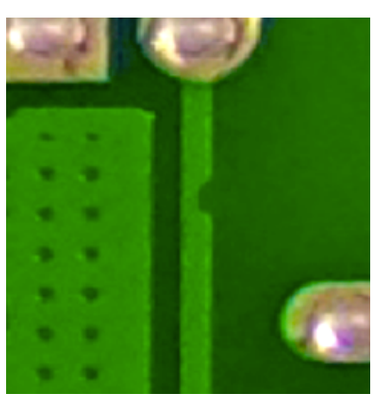
Input

Output: Label

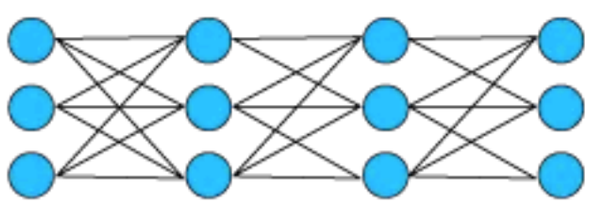
1: good condition

Defect detection

Input: image



Huang, Weibo, and Peng Wei. "A PCB dataset for defects detection and classification." arXiv preprint arXiv:1901.08204 (2019).

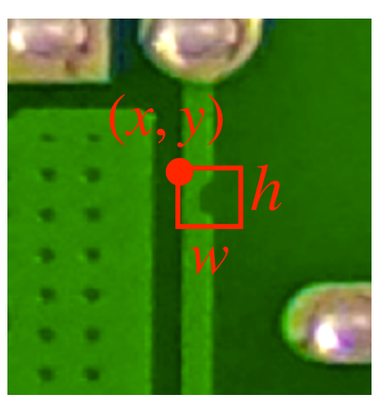


$f_W(x)$

Output: rectangular box(es) and the corresponding class(es)

$$\begin{bmatrix} x \\ y \\ w \\ h \end{bmatrix}$$

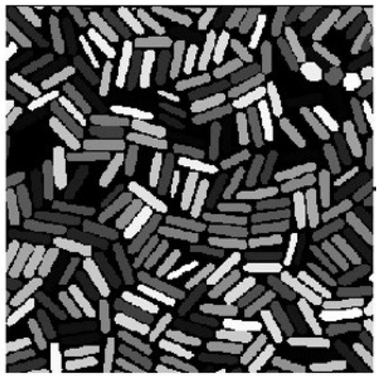
corresponding class



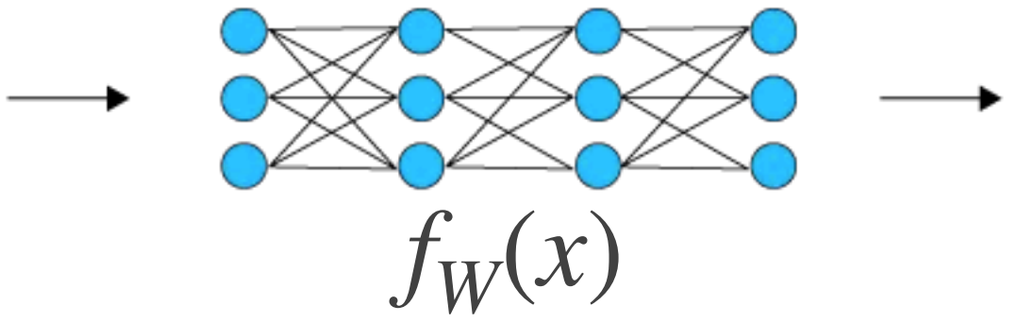
Electron Microscopy Particle Segmentation

Input

EM image



Yildirim, Batuhan, and Jacqueline M. Cole. "Bayesian particle instance segmentation for electron microscopy image quantification." Journal of Chemical Information and Modeling 61.3 (2021): 1136-1149.



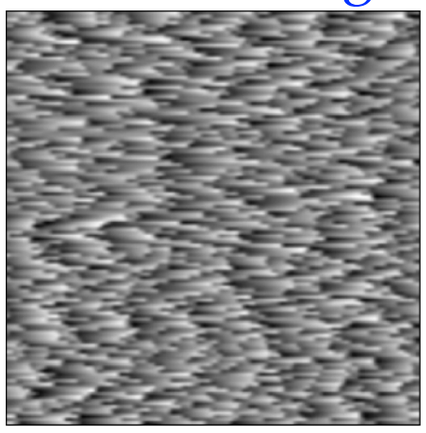
Output

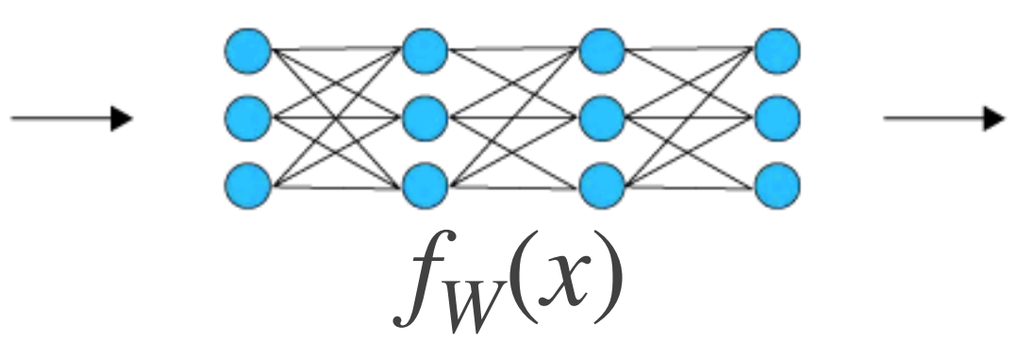
segmentation map

Super-resolution

Input

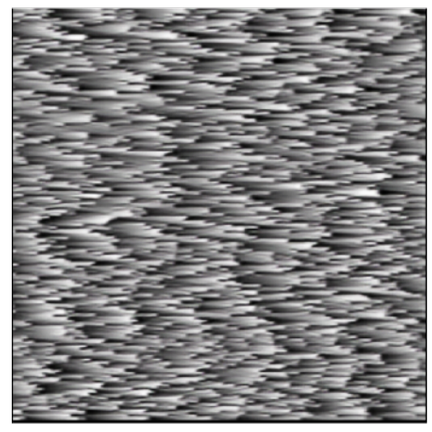
low-res image





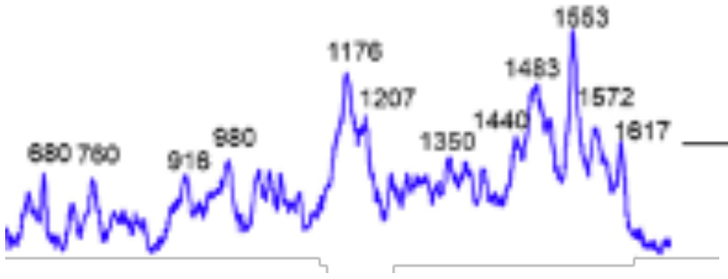
Output

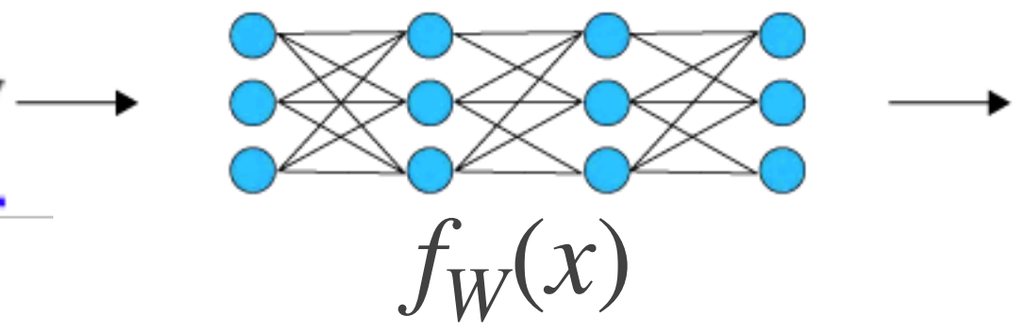
high-res image



Spectrum classification

Input





Output

Category

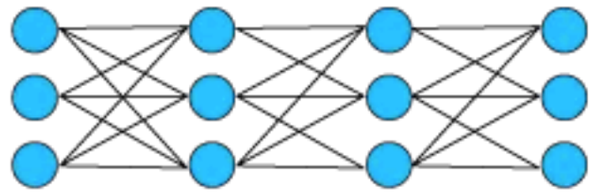
Synthesis parameter optimization

Min, Kyoungmin, et al. "Machine learning assisted optimization of electrochemical properties for Ni-rich cathode materials." Scientific reports 8.1 (2018): 1-7.

Input

Experimental parameters

- Composition
- Temperature
- Dopant
- Washing
- Coating
- ICP
- XRD



$f_W(x)$

Output

Electrochemical properties

- The 1st discharge capacity
- Capacity retention rate
- Amount of residual Li after initial synthesis

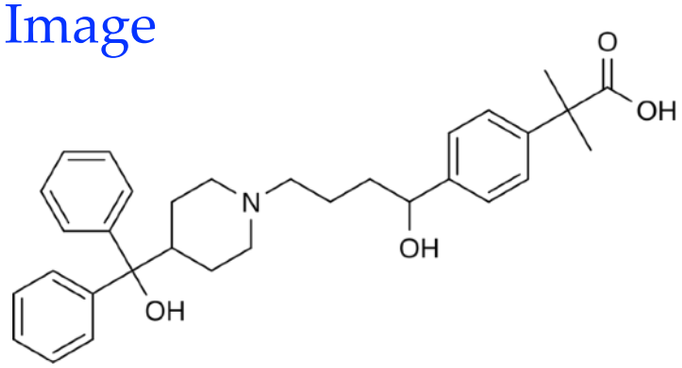
Molecule classification

Input

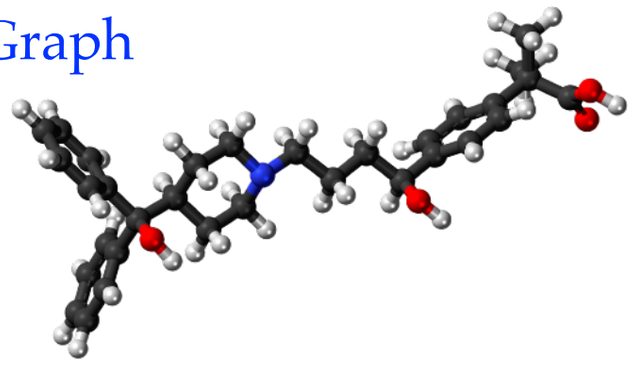
String of characters

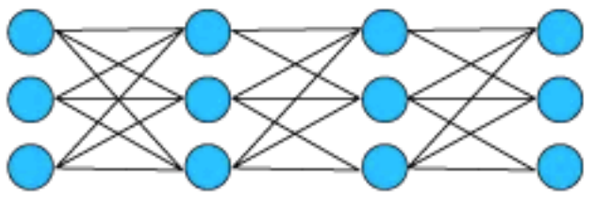
$C_{32}H_{39}NO_4$

Image



Graph





$f_W(x)$

Output

2nd-generation antihistamine drug