MAI4CAREU | Master programmes in Artificial Intelligence 4 Careers in Europe

University of Cyprus

University of Cyprus

MAI645 - Machine Learning for Graphics and Computer Vision

Andreas Aristidou, PhD

Spring Semester 2023

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU
Master programmes in Artificial Intelligence 4 Careers in Europe

Image Classification & Object Detection

Notes have been prepared in collaboration with the Deep Camera MRG, CYENS CoE

Deep Camera

CYENS CENTRE OF EXCELLENCE

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Classification

Image Classification is a fundamental task in computer vision where a model is trained to assign a specific label or class to an input image.

It involves training a model on a large dataset of images, each labeled with a specific class, and then using that model to make predictions on new, unseen images.

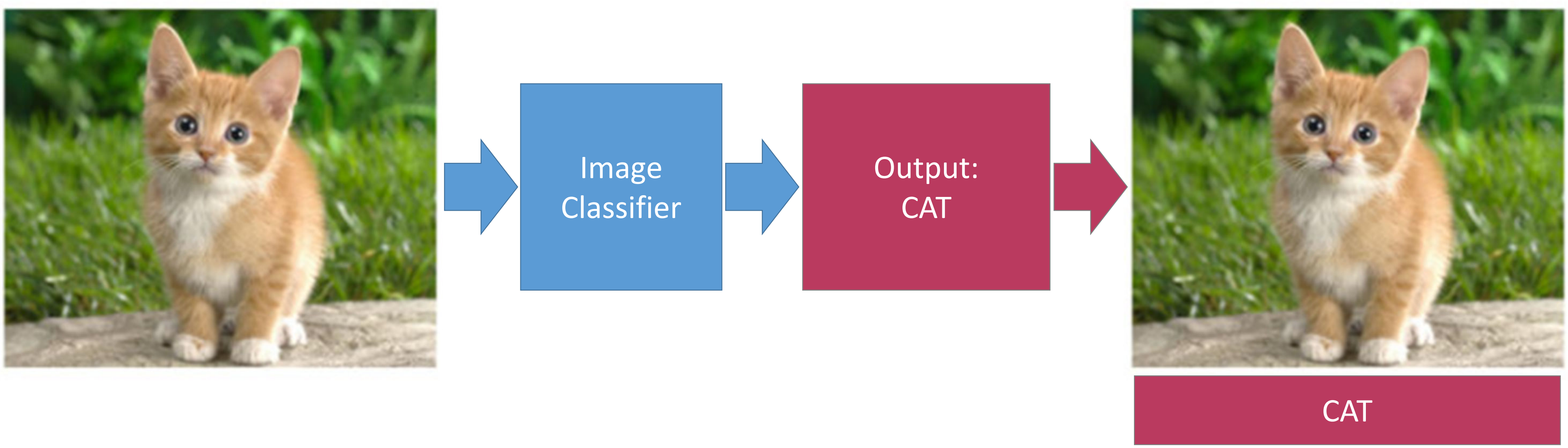
The goal is for the model to accurately categorize the images into their respective classes.

Image classification is used in a variety of applications, such as object recognition, scene understanding, and image retrieval.

Classification

CAT

Image Classification



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Classification

Image Classification: A core task in Computer Vision

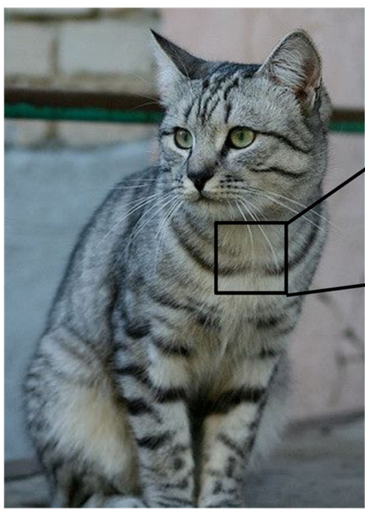
(assume given a set of possible labels)
{dog, cat, truck, plane, ...}

→ cat

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Classification

The Problem: Semantic Gap




```
[[105 112 108 111 104  99 106  99  96 103 112 119 104  97  93  87]
 [ 91  98 102 106 104  79  98 103  99 105 123 136 110 105  94  85]
 [ 76  85  90 105 128 105  87  96  95  99 115 112 106 103  99  85]
 [ 99  81  81  93 120 131 127 100  95  98 102  99  96  93 101  94]
 [106  91  61  64  69  91  88  85 101 107 109  98  75  84  96  95]
 [114 108  85  55  55  69  64  54  64  87 112 129  98  74  84  91]
 [133 137 147 103  65  81  80  65  52  54  74  84 102  93  85  82]
 [128 137 144 140 109  95  86  70  62  65  63  63  60  73  86 101]
 [125 133 148 137 119 121 117  94  65  79  80  65  54  64  72  98]
 [127 125 131 147 133 127 126 131 111  96  89  75  61  64  72  84]
 [115 114 109 123 150 148 131 118 113 109 100  92  74  65  72  78]
 [ 89  93  90  97 108 147 131 118 113 114 113 109 106  95  77  80]
 [ 63  77  86  81  77  79 102 123 117 115 117 125 125 130 115  87]
 [ 62  65  82  89  78  71  80 101 124 126 119 101 107 114 131 119]
 [ 63  65  75  88  89  71  62  81 120 138 135 105  81  98 110 118]
 [ 87  65  71  87 106  95  69  45  76 130 126 107  92  94 105 112]
 [118  97  82  86 117 123 116  66  41  51  95  93  89  95 102 107]
 [164 146 112  80  82 120 124 104  76  48  45  66  88 101 102 109]
 [157 170 157 120  93  86 114 132 112  97  69  55  70  82  99  94]
 [130 128 134 161 139 100 109 118 121 134 114  87  65  53  69  86]
 [128 112  96 117 150 144 120 115 104 107 102  93  87  81  72  79]
 [123 107  96  86  83 112 153 149 122 109 104  75  80 107 112  99]
 [122 121 102  80  82  86  94 117 145 148 153 102  58  78  92 107]
 [122 164 148 103  71  56  78  83  93 103 119 139 102  61  69  84]]
```

What the computer sees

An image is a tensor of integers between [0, 255]:

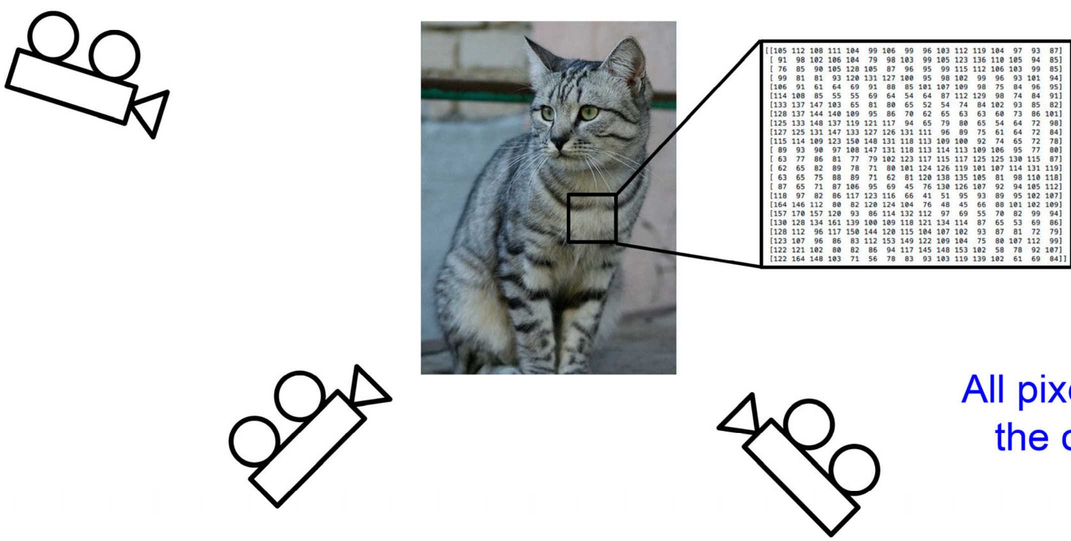
e.g. 800 x 600 x 3
(3 channels RGB)

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Classification

Challenges: Viewpoint variation



All pixels change when
the camera moves!

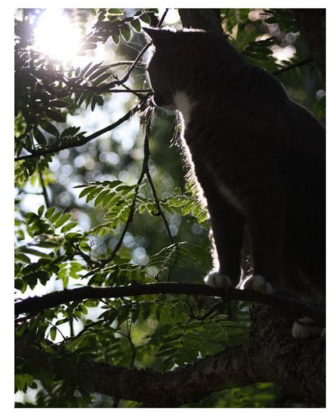
Image Classification

Challenges: Illumination

This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

This image is CC0 1.0 public domain



This image is CC0 1.0 public domain

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Classification

Challenges: Background Clutter

This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

Image Classification

Challenges: Occlusion

This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

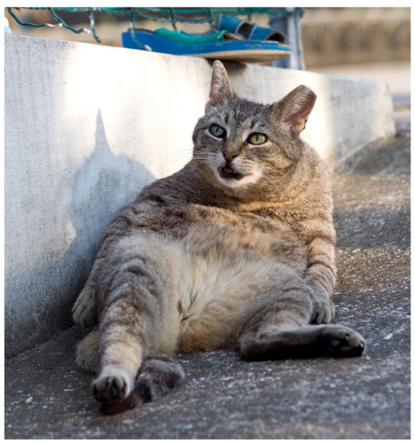
This image by jonsson is licensed under CC-BY 2.0

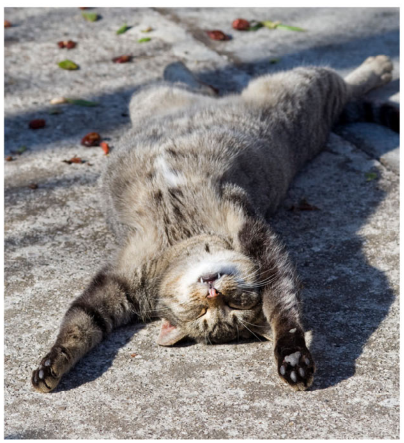
Co-financed by the European Union
Connecting Europe Facility

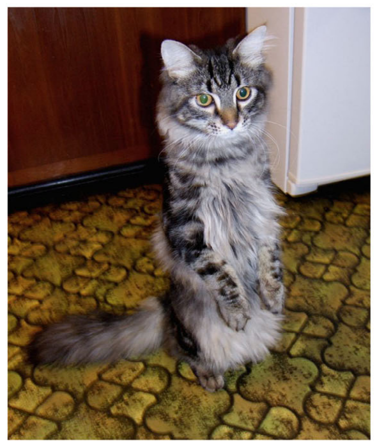
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Classification

Challenges: Deformation







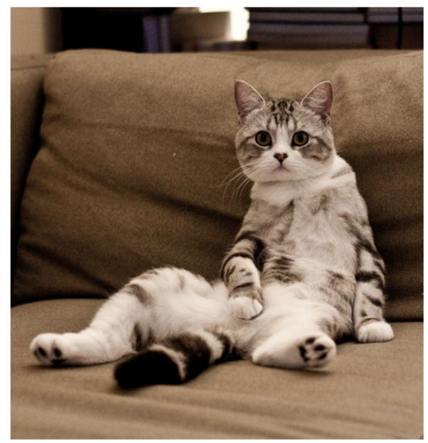
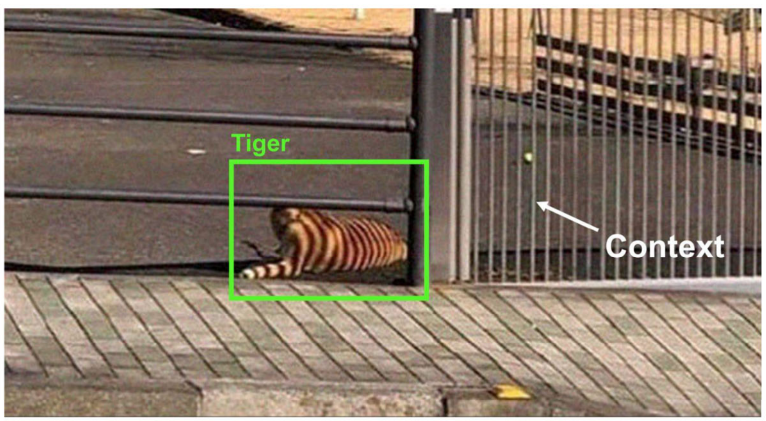


Image Classification

Challenges: Intraclass variation

Image Classification

Challenges: Context

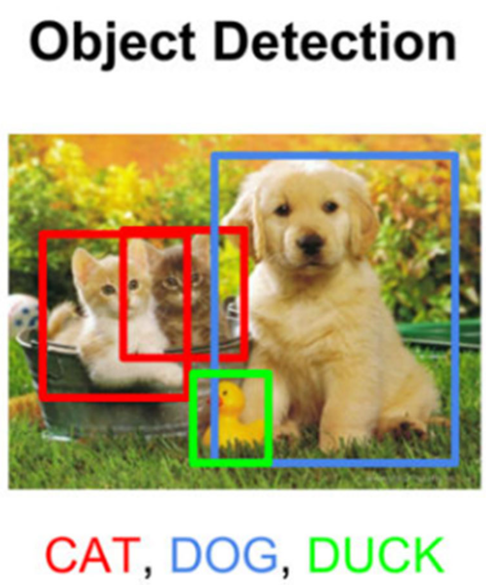


Object detection

Object detection is a fundamental task in computer vision where the goal is to locate and classify objects within an image or video.

It is a more advanced form of image classification, where instead of just identifying the class of an entire image, they identify multiple instances of multiple classes within an image and locate them with a bounding box; in other words, it deals with more realistic cases in which multiple objects may exist in an image

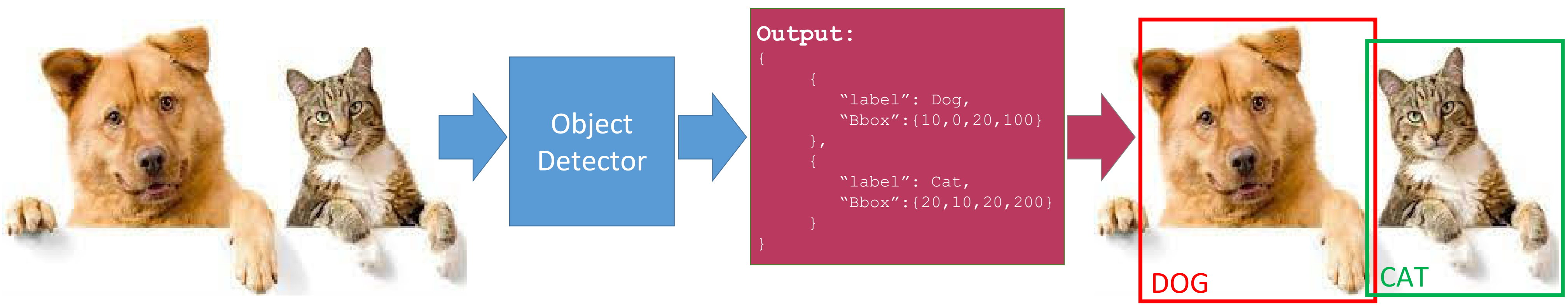
Object detection algorithms can be used in a variety of applications, such as self-driving cars, security systems, and augmented reality.



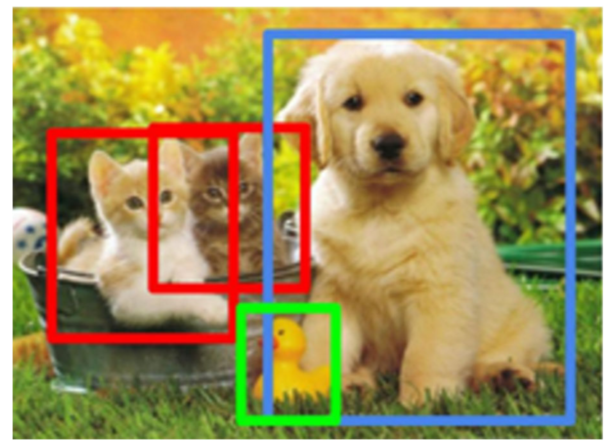
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Object detection



Differences

Image Classification	Localization	Object Detection
		
CAT	CAT	CAT, DUCK, DOG

Importance

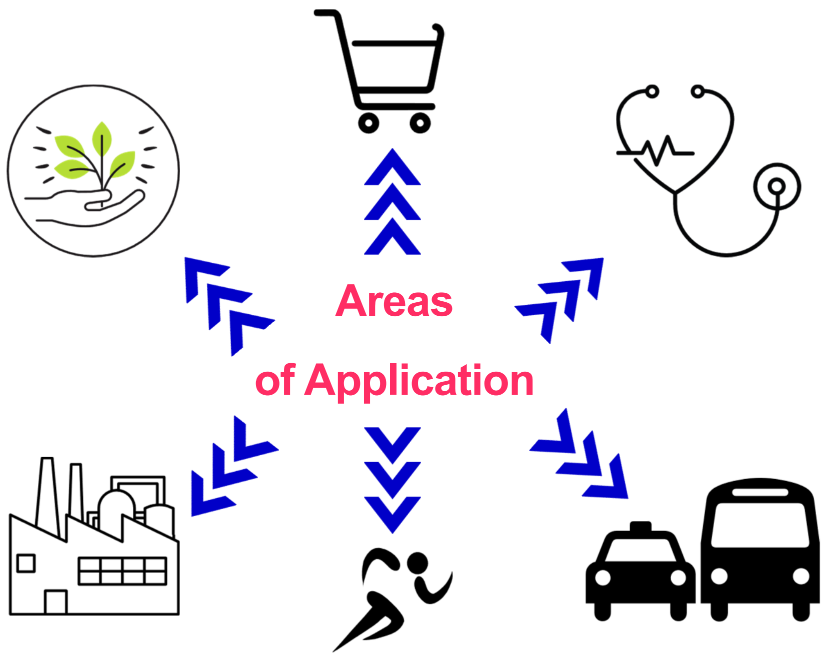
Image classification and object detection are important tasks due to their real-world applications, contributions to the advancement of artificial intelligence, improvements in accuracy and efficiency, and facilitation of human-computer interaction. In particular:

1. **Real-world applications:** Image classification and object detection have numerous real-world applications, including self-driving cars, surveillance systems, and image search engines.
2. **Advancements in artificial intelligence:** These tasks play a significant role in advancing the field of artificial intelligence and computer vision by providing a framework for developing and evaluating new algorithms and models.
3. **Improved accuracy and efficiency:** Image classification and object detection algorithms have been constantly improving in terms of accuracy and efficiency. This enables more robust and reliable systems for various applications.
4. **Facilitation of human-computer interaction:** By automating tasks such as recognizing and locating objects in images, these algorithms make it possible to interact with computers in a more natural and intuitive way.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Areas of application of image classification and object detection



Areas of application of image classification and object detection

Image classification and object detection have a wide range of applications in various fields such as:

1. **Computer Vision:** Image classification and object detection are the fundamental tasks in computer vision and are used in various vision-based applications.
2. **Surveillance Systems:** These technologies are used to detect and classify objects in real-time surveillance systems to improve the accuracy of security and monitoring.
3. **Autonomous vehicles:** Object detection is used in autonomous vehicles for tasks such as lane detection, obstacle detection, and traffic sign recognition.
4. **Medical Imaging:** Image classification and object detection techniques are used in medical imaging for tasks such as lesion detection, tumor segmentation, and diagnosis.
5. **Agriculture:** Object detection is used in agriculture for tasks such as crop counting and monitoring crop growth.
6. **Robotics:** Image classification and object detection are used in robotics for tasks such as object recognition and grasping.
7. **E-commerce:** Image classification and object detection are used in e-commerce for product categorization, image-based search, and automatic tagging.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

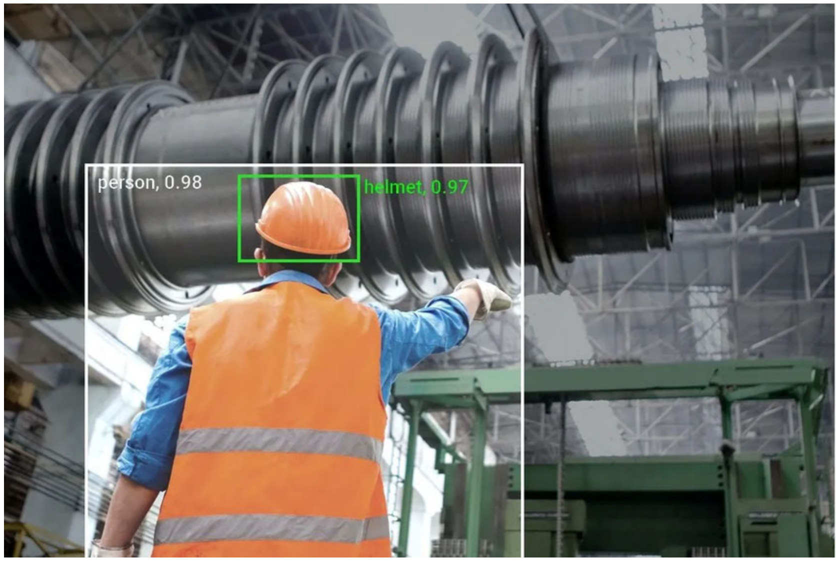
Areas of application of image classification and object detection

Image classification and object detection have a wide range of applications in various fields such as:

8. **Augmented Reality (AR) and Virtual Reality (VR):** Image classification and object detection are used in AR and VR for tasks such as 3D object recognition and tracking.
9. **Sports:** Object detection is used in sports for tasks such as player tracking and ball detection.
10. **Marketing and Advertising:** Image classification and object detection are used in marketing and advertising for tasks such as image-based recommendations, image-based search, and product categorization.
11. **Wildlife conservation:** Image classification and object detection are used in wildlife conservation for tasks such as animal tracking and species identification.
12. **Retail:** Image classification and object detection are used in retail for tasks such as product recognition, price comparison, and visual search.
13. **Face recognition:** Image classification and object detection are used in face recognition systems for tasks such as facial detection, facial landmarks, and facial verification.
14. **Natural Language Processing (NLP):** Image classification and object detection are used in NLP for tasks such as image captioning, visual question answering, and sentiment analysis.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

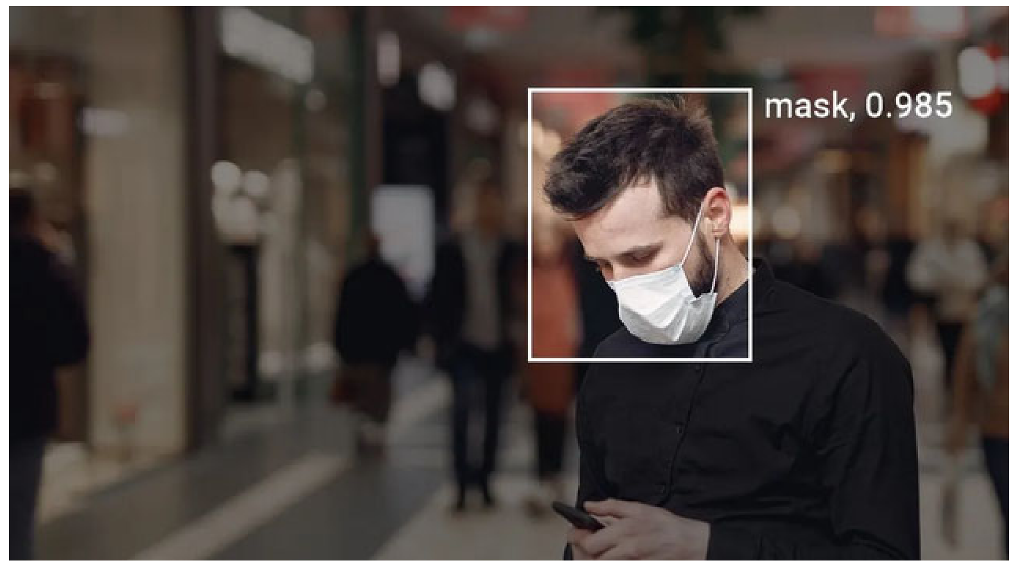
Areas of application of image classification and object detection



Visual Inspection of Equipment

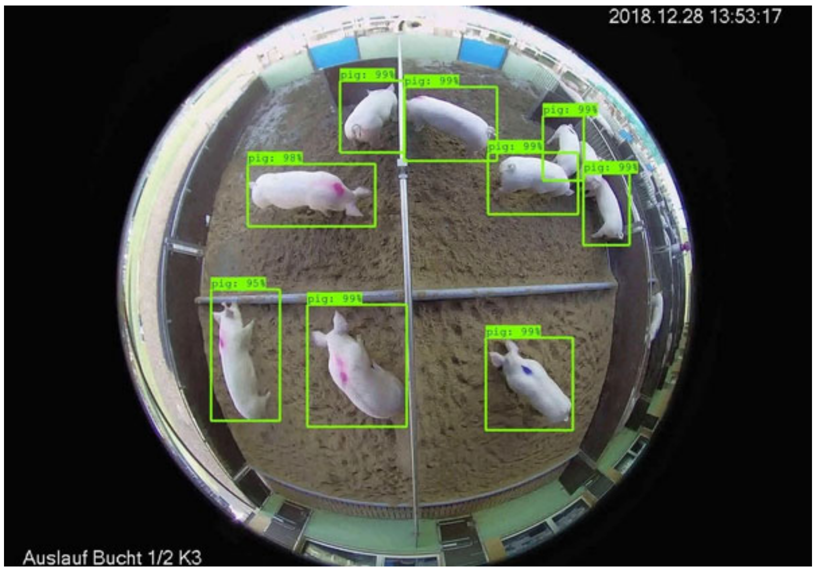
Areas of application of image classification and object detection

Smart health care: pose detection

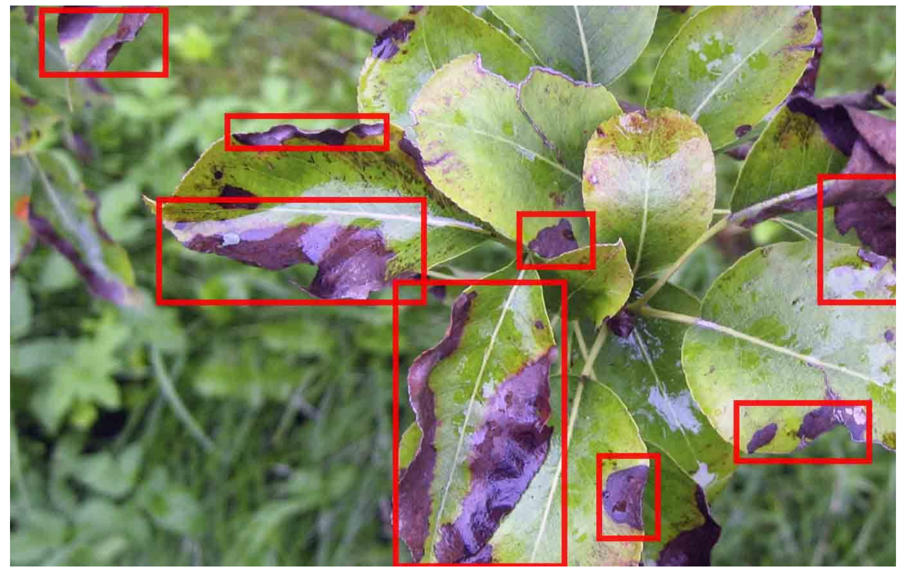


Smart health care: mask detection

Areas of application of image classification and object detection

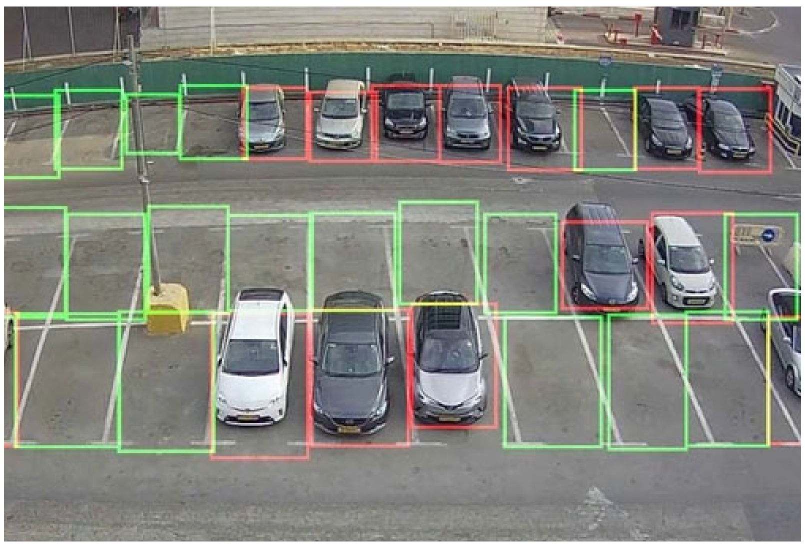


Smart agriculture: animal monitoring

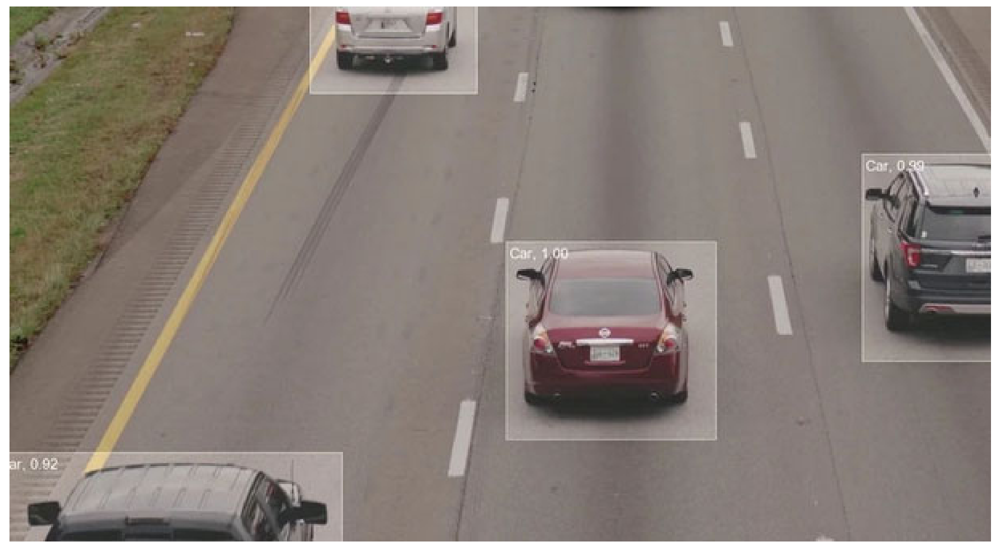


Smart agriculture: Plant Disease Detection

Areas of application of image classification and object detection



Parking Occupancy Detection



Vehicle Classification

Approaches to the problem

Unsupervised

Unsupervised learning is a type of machine learning where the model is trained on an unlabeled dataset, and the algorithm tries to find patterns or relationships within the data without any prior knowledge of the expected outcome. The goal is to group similar data points together or to find lower-dimensional representations of the data. Examples of unsupervised learning algorithms are clustering and dimensionality reduction.

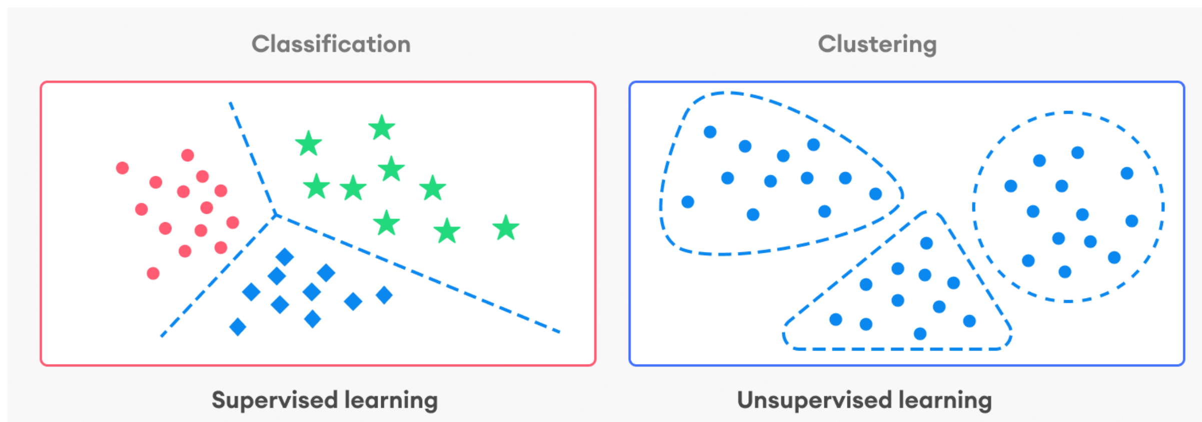
Supervised

Supervised learning, on the other hand, is a type of machine learning where the model is trained on labeled data, where the desired output is already known. The goal is to learn a mapping from inputs to outputs based on the labeled examples. The algorithm uses this mapping to make predictions on new unseen data. Examples of supervised learning algorithms are linear regression, decision trees, and support vector machines.

Approaches to the problem

Unsupervised

Supervised



Unsupervised Image Classification

Unsupervised Image Classification is where the outcomes (groupings of pixels with common characteristics) are based on the software analysis of an image without the user providing sample classes.

Pixel Space

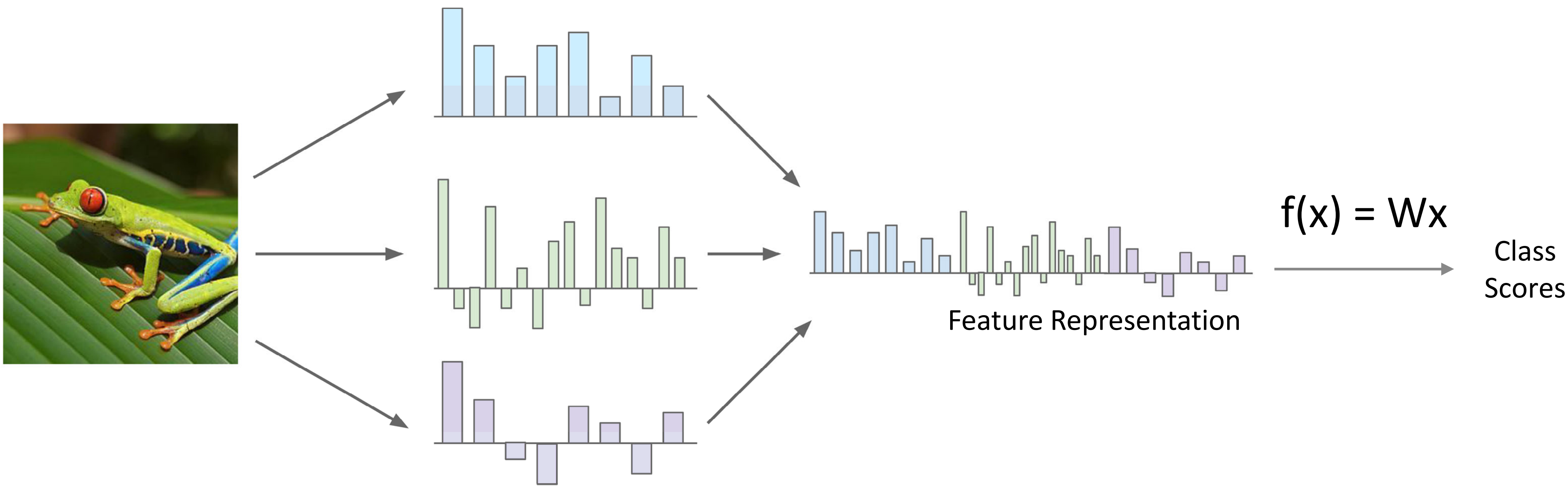
Class scores

$$f(x) = Wx$$

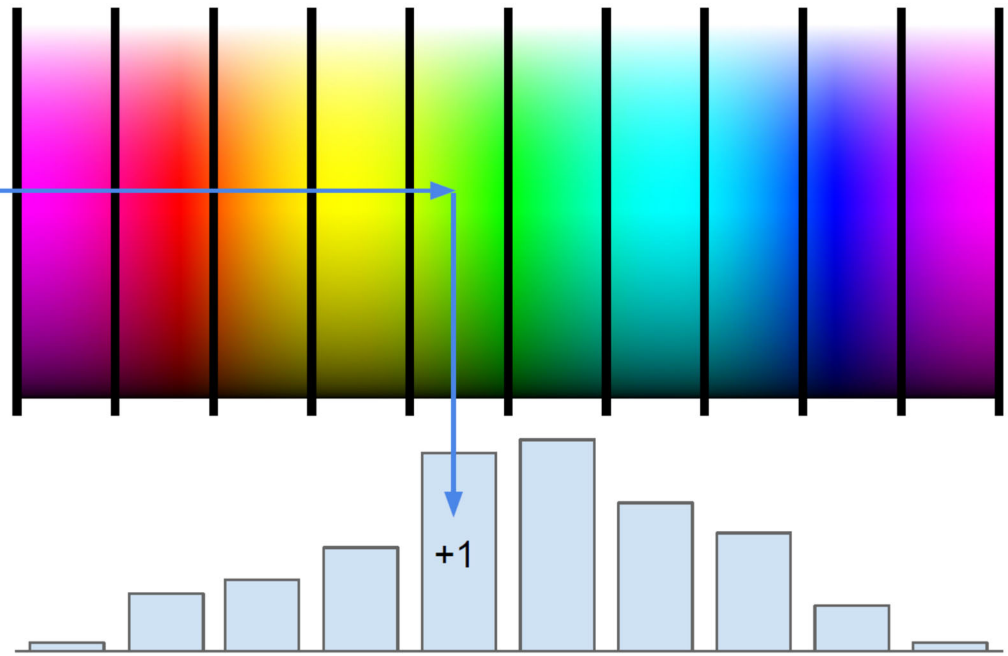
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Features

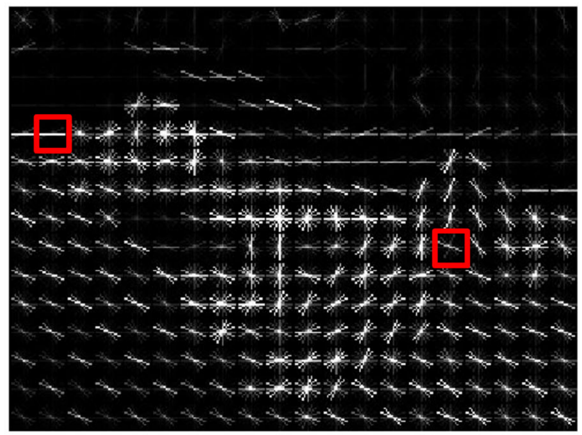


Example: Color Histogram



Example: Histogram of Oriented Gradients (HoG)

Divide image into 8x8 pixel regions
Within each region quantize edge direction into 9 bins



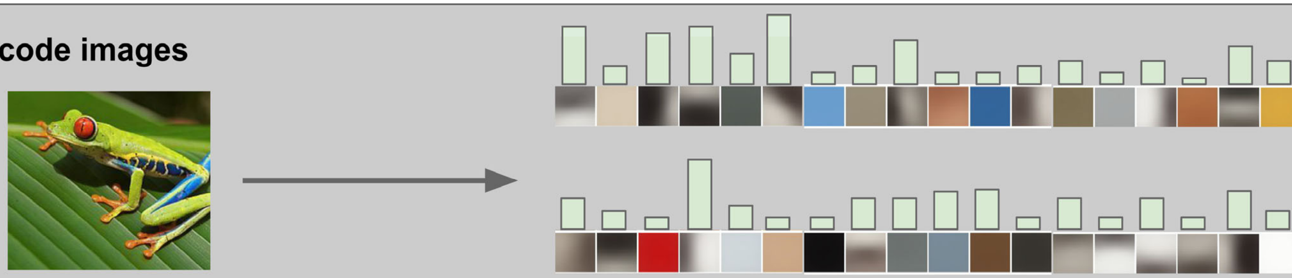
Example: 320x240 image gets divided into 40x30 bins; in each bin there are 9 numbers so feature vector has 30*40*9 = 10,800 numbers

Lowe, “Object recognition from local scale-invariant features”, ICCV 1999
Dalal and Triggs, "Histograms of oriented gradients for human detection," CVPR 2005

Unsupervised Image Classification

Step 1: Build codebook

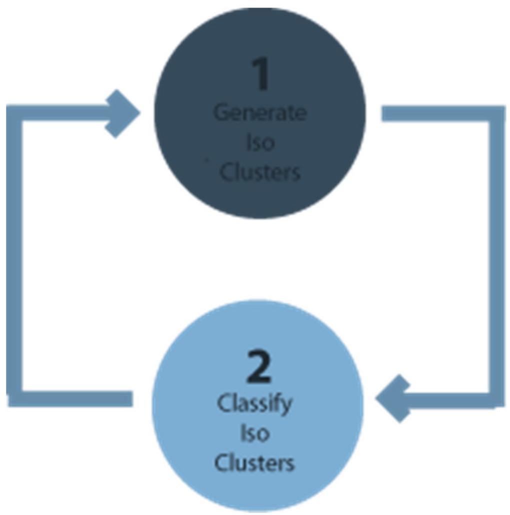
Step 2: Encode images



Fei-Fei and Perona, "A bayesian hierarchical model for learning natural scene categories", CVPR 2005

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Unsupervised Image Classification



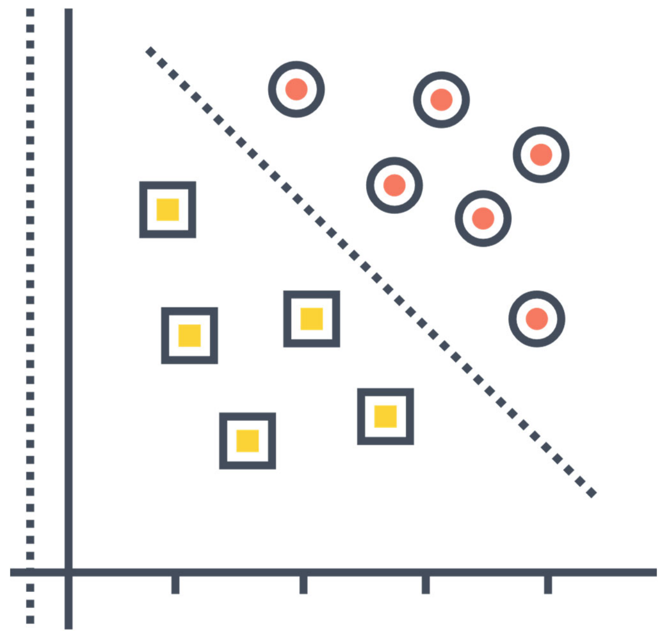
In unsupervised classification, it first groups pixels into "clusters" based on their properties.

Then, you classify each cluster with a class.

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Steps for unsupervised image classification

1. Choose Clustering Algorithm
2. Class Identification
3. Edit\Evaluate Signatures
4. Class Evaluation



Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Unsupervised image classification: Clusterization

With unsupervised algorithms, **no pre-existing tags** are given to the system, **only raw data**. The system interprets the data, recognizes patterns, and draws unique conclusions from the data without human interference.

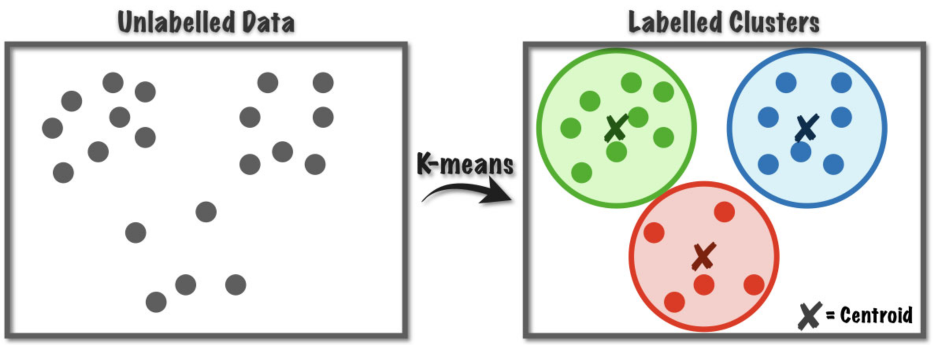
Unsupervised classification makes use of a concept called **clusterization**. Clusterization is the unsupervised, natural locating and grouping (or "clustering") of data into groups. However, you will not give get a class automatically. You'll only have the different clusters, which you'd need to decide a class for in another way. There are a plethora of different clusterization algorithms e.g., K-Means, Agglomerative Clustering, BIRCH, ISODATA, DBSCAN etc.

There isn't a single best choice out of these clusterization algorithms. Instead, it is optimal to test various ones until you settle on the one that works best with the specific task at hand.

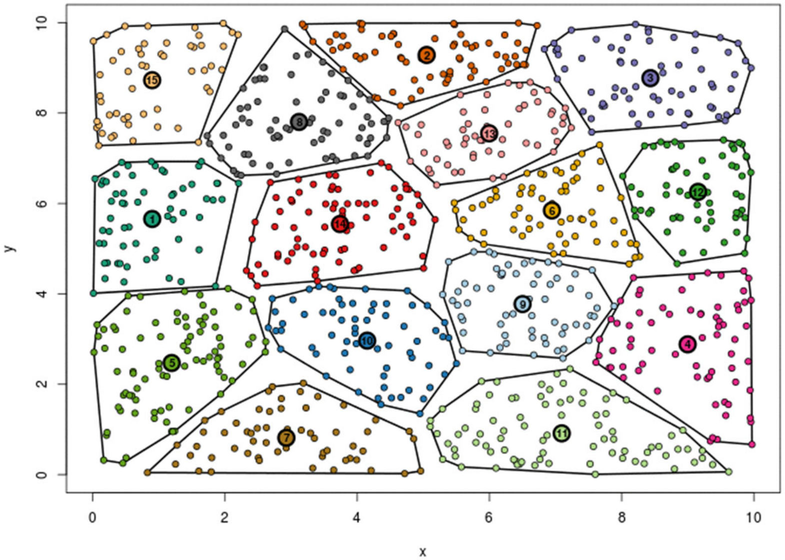
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Unsupervised image classification: Clusterization

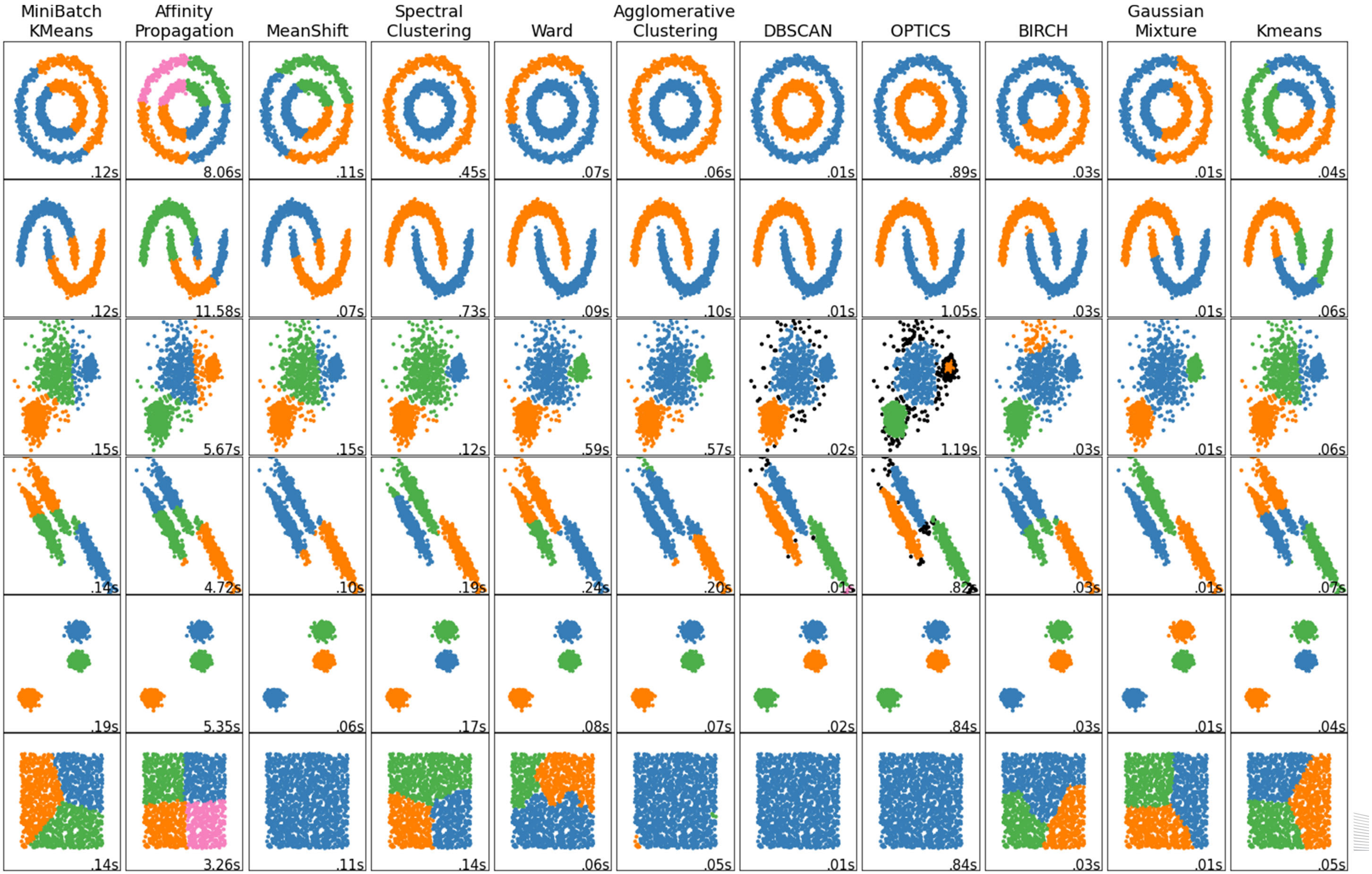


K-means



ISODATA

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423


MiniBatch KMeans
Affinity Propagation
MeanShift
Spectral Clustering
Ward
Agglomerative Clustering
DBSCAN
OPTICS
BIRCH
Gaussian Mixture
Kmeans
.12s 8.06s .11s .45s .07s .06s .01s .89s .03s .01s .04s
.12s 11.58s .07s .73s .09s .10s .01s 1.05s .03s .01s .06s
.15s 5.67s .15s .12s .59s .57s .02s 1.19s .03s .01s .06s
.14s 4.72s .10s .19s .24s .20s .01s .82s .03s .01s .07s
.19s 5.35s .06s .17s .08s .07s .02s .84s .03s .01s .04s
.14s 3.26s .11s .14s .06s .05s .01s .84s .03s .01s .05s

Supervised Image Classification

Supervised classification is based on the idea that a user can select sample pixels in an image that are representative of specific classes and then direct the image processing software to use these training sites as references for the classification of all other pixels in the image.

In supervised classification, you select representative samples for each class. The software then uses these "training sites" and applies them to the entire image.

Supervised Image Classification

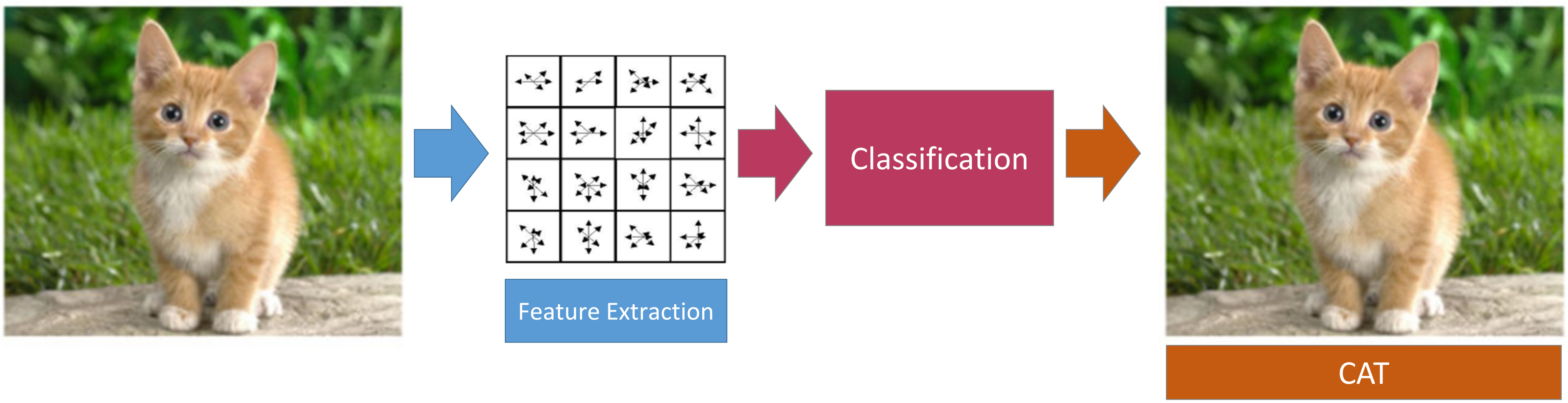
Supervised classification involves **pre-training** the system with a **set of reference data**, allowing it to use the acquired information to classify new visual materials. The algorithm compares the new input with the previously trained data, using the patterns learned from the training data to classify the new images.

Supervised image classification algorithms can be divided into **single-label classification** and **multi-label classification**. Single-label classification refers to a singular label that is assigned to an image as a result of the classification process. While single-label classification assigns an image to a single category, multi-label classification allows an image to be assigned to an unlimited number of categories. Multi-label classification can be particularly useful in cases where an image contains multiple features or attributes. For instance, in medicine, a medical image may reveal multiple diseases or abnormalities in a patient.

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Image Classification Using Traditional Machine learning



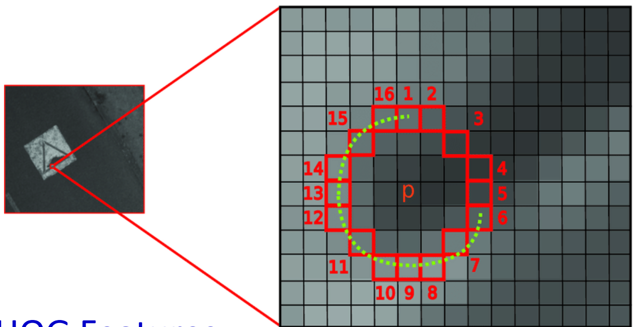
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

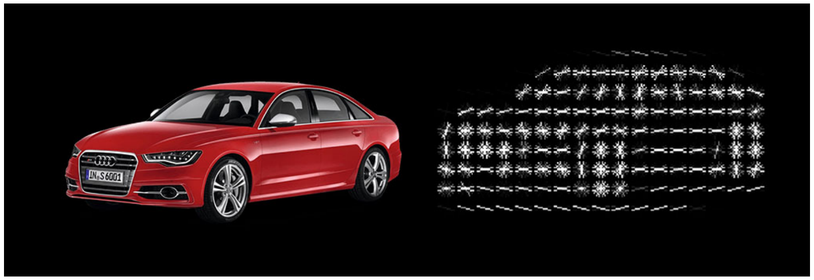
Again, some feature extraction techniques:

- HOG Features
  - Histogram of Oriented Gradients (HOG) is a feature descriptor used in computer vision and image processing. It is used to represent the shape and structure of objects in an image. HOG works by dividing an image into small cells and computing the gradient orientation histogram for each cell.
- Accelerated segment test (AST)
  - The Accelerated Segment Test is a fast and efficient algorithm for detecting changes in the mean of a time-series. It is often used in change point detection problems, where the goal is to identify a point in time when the underlying distribution of the time-series changes.
- Scale Invariant Feature Transform (SIFT)
  - Scale Invariant Feature Transform (SIFT) is an algorithm for detecting and describing local features in images. It is used for tasks such as object recognition, image matching, and texture classification. SIFT works by detecting distinctive, invariant features in an image that are robust to changes in scale, orientation, and illumination.
- Oriented FAST and Rotated BRIEF (ORB)
  - Oriented FAST and Rotated BRIEF (ORB) is a feature detection and description algorithm in computer vision. It is a combination of the FAST (Features from Accelerated Segment Test) corner detector and the BRIEF (Binary Robust Independent Elementary Features) descriptor. FAST is a fast corner detection algorithm that is used to detect features in an image. BRIEF is a binary feature descriptor that describes the local appearance of a feature using a binary string.

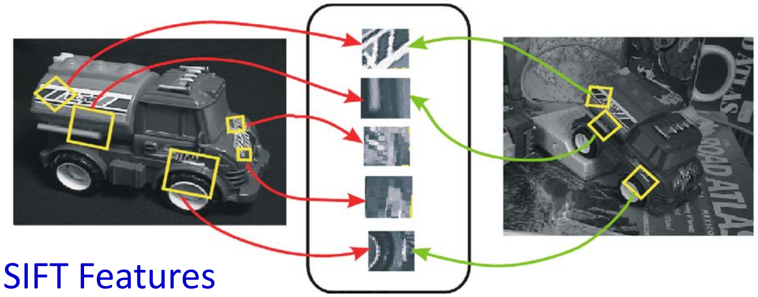
Again, some feature extraction techniques:



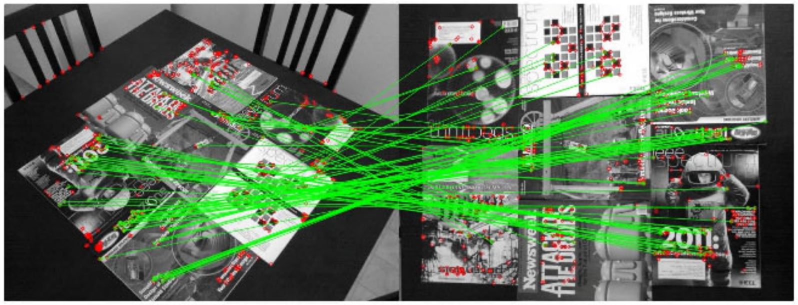
HOG Features



AST Features



SIFT Features



ORB Features

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Example of Classifiers

- Decision Tree classifier
  - A Decision Tree classifier is a simple and popular machine learning algorithm used for solving classification problems. It is a type of decision tree algorithm, where the tree is used to make predictions by recursively partitioning the data into smaller subsets based on the values of the input features. Each node in the tree represents a feature, and the branches represent the possible values of that feature. The leaves of the tree represent the class labels, and the path from the root to a leaf represents a decision rule for making predictions. To make a prediction for a new input, the algorithm follows the path through the tree that corresponds to the values of the input features.
- Random Forest Classifier
  - In a Random Forest Classifier, a large number of decision trees are grown, and each tree is trained on a randomly selected subset of the data. When making a prediction for a new input, the Random Forest Classifier aggregates the predictions made by each individual decision tree and outputs the class label that is predicted by the majority of trees.
- Naive Bayes classifier
  - The Naive Bayes classifier is a probabilistic machine learning algorithm used for classification problems. It is based on Bayes' theorem, which states that the probability of a hypothesis (e.g., a class label) given some observed evidence (e.g., input features) can be estimated based on prior probabilities of the hypothesis and the probability of the evidence given the hypothesis.
- Support vector machine
  - Support Vector Machine (SVM) is a type of supervised learning algorithm used for classification and regression analysis. It is a boundary-based algorithm that finds the maximum-margin boundary that separates the classes in the data.

Data-driven approaches

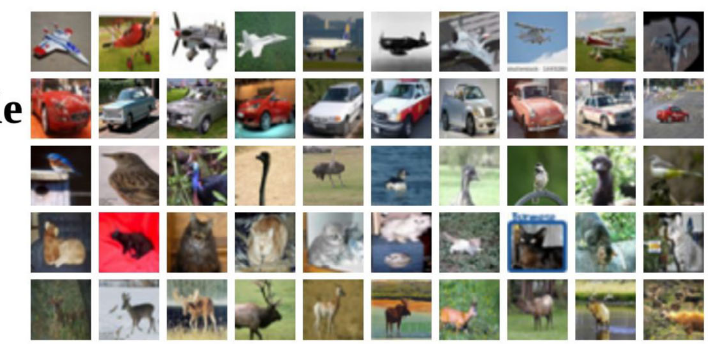
1. Collect a dataset of images and labels
2. Use Machine Learning algorithms to train a classifier
3. Evaluate the classifier on new images

```
def train(images, labels):
  # Machine learning!
  return model
```

```
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```

Example training set

airplane
automobile
bird
cat
deer



First classifier: Nearest Neighbor

```
def train(images, labels):
  # Machine learning!
  return model
```

→ Memorize all data and labels

```
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```

→ Predict the label of the most similar training image

First classifier: Nearest Neighbor

deer bird plane cat car

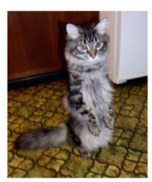
Training data with labels

?

query data

Distance Metric |

,



| $\rightarrow \mathbb{R}$

First classifier: Nearest Neighbor

Distance Metric to compare images

L1 distance: $$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

$$\left| \underbrace{\begin{array}{|c|c|c|c|}\hline 56 & 32 & 10 & 18 \\ \hline 90 & 23 & 128 & 133 \\ \hline 24 & 26 & 178 & 200 \\ \hline 2 & 0 & 255 & 220 \\ \hline \end{array}}_{\text{test image}} - \underbrace{\begin{array}{|c|c|c|c|}\hline 10 & 20 & 24 & 17 \\ \hline 8 & 10 & 89 & 100 \\ \hline 12 & 16 & 178 & 170 \\ \hline 4 & 32 & 233 & 112 \\ \hline \end{array}}_{\text{training image}} \right| = \underbrace{\begin{array}{|c|c|c|c|}\hline 46 & 12 & 14 & 1 \\ \hline 82 & 13 & 39 & 33 \\ \hline 12 & 10 & 0 & 30 \\ \hline 2 & 32 & 22 & 108 \\ \hline \end{array}}_{\text{pixel-wise absolute value differences}} \xrightarrow{\text{add}} 456$$

A good implementation:
https://github.com/facebookresearch/faiss

First classifier: Nearest Neighbor

```
import numpy as np

class NearestNeighbor:
  def __init__(self):
    pass

  def train(self, X, y):
    """ X is N x D where each row is an example. Y is 1-dimension of size N """
    # the nearest neighbor classifier simply remembers all the training data
    self.Xtr = X
    self.ytr = y

  def predict(self, X):
    """ X is N x D where each row is an example we wish to predict label for """
    num_test = X.shape[0]
    # lets make sure that the output type matches the input type
    Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

    # loop over all test rows
    for i in xrange(num_test):
      # find the nearest training image to the i'th test image
      # using the L1 distance (sum of absolute value differences)
      distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
      min_index = np.argmin(distances) # get the index with smallest distance
      Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

    return Ypred
```

Memorize training data

For each test image:
- Find closest train image
- Predict label of nearest image

Q: With N examples, how fast are training and prediction?

Ans: Train O(1), predict O(N)

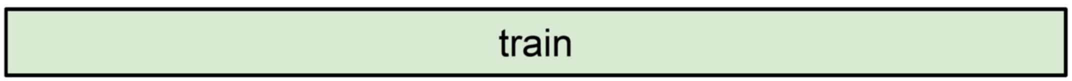
This is bad: we want classifiers that are **fast** at prediction; **slow** for training is ok

First classifier: Nearest Neighbor

Setting Hyperparameters

Idea #1: Choose hyperparameters that work best on the **training data**

BAD: K = 1 always works perfectly on training data



First classifier: Nearest Neighbor

Setting Hyperparameters

Idea #1: Choose hyperparameters that work best on the **training data**

BAD: K = 1 always works perfectly on training data

train

Idea #2: choose hyperparameters that work best on **test** data

BAD: No idea how algorithm will perform on new data

train	test

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

First classifier: Nearest Neighbor

Setting Hyperparameters

Idea #1: Choose hyperparameters that work best on the **training data**

BAD: K = 1 always works perfectly on training data

train

Idea #2: choose hyperparameters that work best on **test** data

BAD: No idea how algorithm will perform on new data

train	test

Idea #3: Split data into **train**, **val**; choose hyperparameters on val and evaluate on test

Correct!!!

train	validation	test

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

First classifier: Nearest Neighbor

Setting Hyperparameters

train

Idea #4: **Cross-Validation**: Split data into **folds**,
try each fold as validation and average the results

fold 1	fold 2	fold 3	fold 4	fold 5	test
fold 1	fold 2	fold 3	fold 4	fold 5	test
fold 1	fold 2	fold 3	fold 4	fold 5	test

Useful for small datasets, but not used too frequently in deep learning

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

First classifier: Nearest Neighbor

Example Dataset: **CIFAR10**

10 classes
50,000 training images
10,000 testing images

Alex Krizhevsky, "Learning Multiple Layers of Features from Tiny Images", Technical Report, 2009.

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

First classifier: Nearest Neighbor

Example Dataset: **CIFAR10**

10 classes
50,000 training images
10,000 testing images

Test images and nearest neighbors

Alex Krizhevsky, "Learning Multiple Layers of Features from Tiny Images", Technical Report, 2009.

K-Nearest Neighbors: Summary

In **image classification** we start with a **training set** of images and labels, and must predict labels on the **test set**

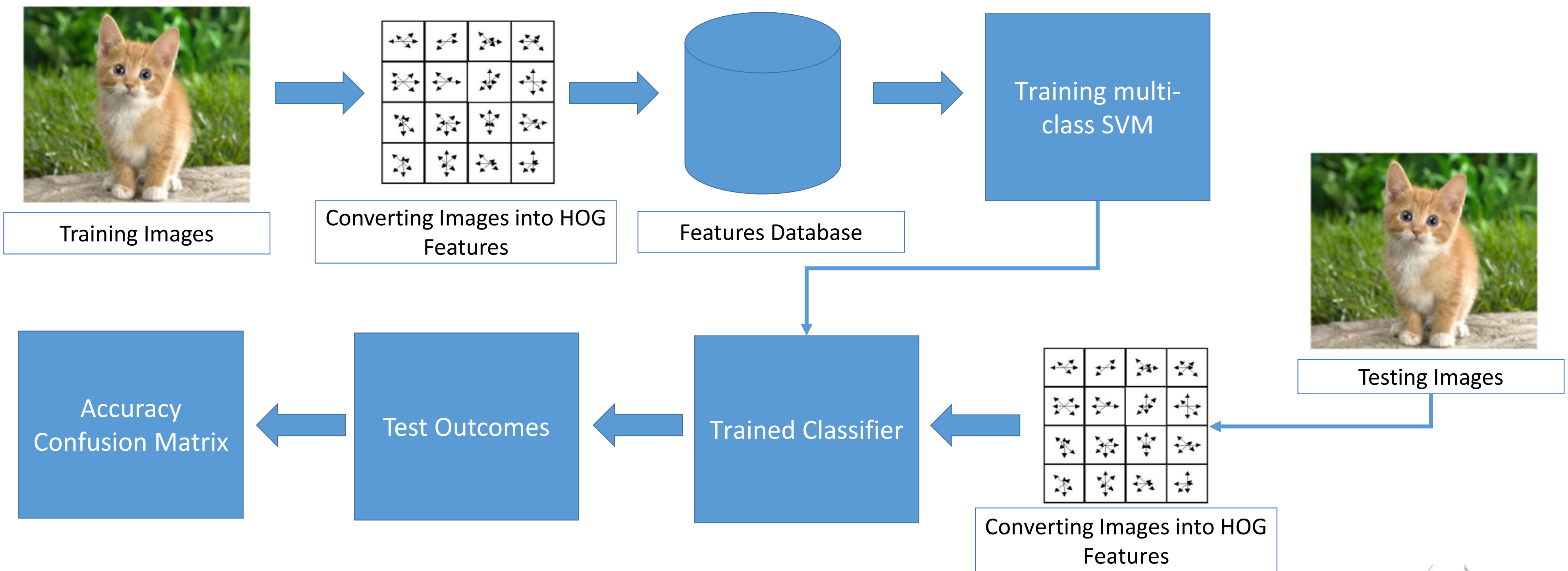
The **K-Nearest Neighbors** classifier predicts labels based on the K nearest training examples

- Distance metric and K are **hyperparameters**
- Choose hyperparameters using the **validation set**;
- Only run on the test set once at the very end!

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Traditional Image Classification: *Example of using HOG and SVM*



This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Parametric Approach*

10x1 10x3072

$$\boxed{f(w,X)} = \boxed{W}\boxed{x} + \boxed{b}$$

10x1

3072x1

Image

Array of **32x32x3** numbers
(3072 numbers total)

$\longrightarrow f(x, W) \longrightarrow$ **10** numbers giving class scores

W

parameters
or weights

Co-financed by the European Union
Connecting Europe Facility

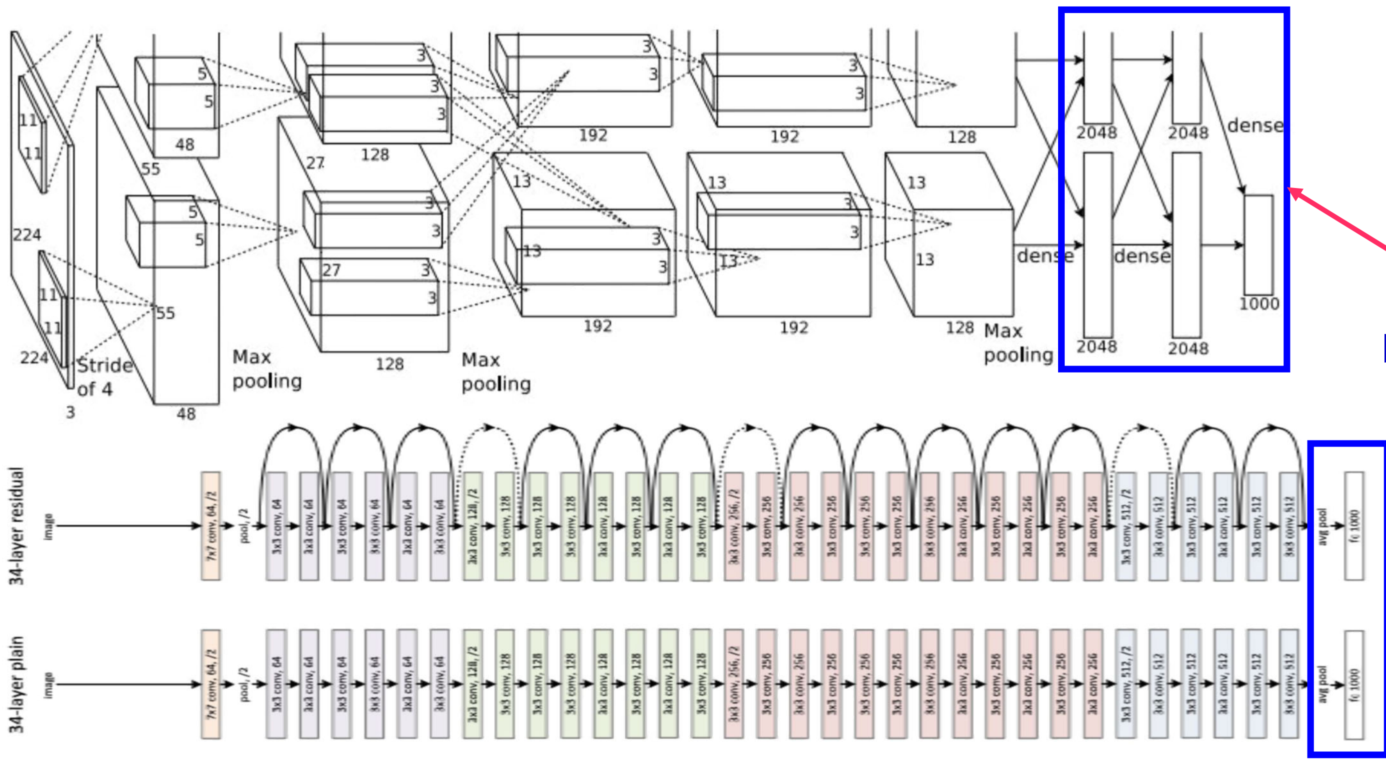
This Master is run under the context of Action
No 2020-EU-IA-0087, co-financed by the EU CEF Telecom
under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Parametric Approach*

This image is CC0 1.0 public domain

A **neural network** consists of several **linear classifiers**

Linear Classifier: *Parametric Approach*



[Krizhevsky et al. 2012]

linear layers

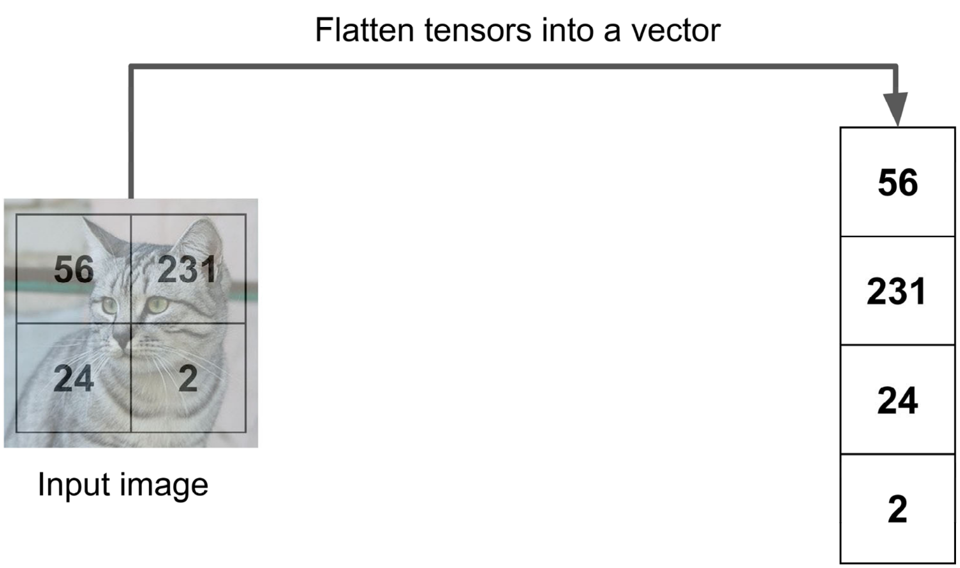
[He et al. 2015]

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Parametric Approach*

Example with an image with 4 pixels, and 3 classes (cat/dog/ship)



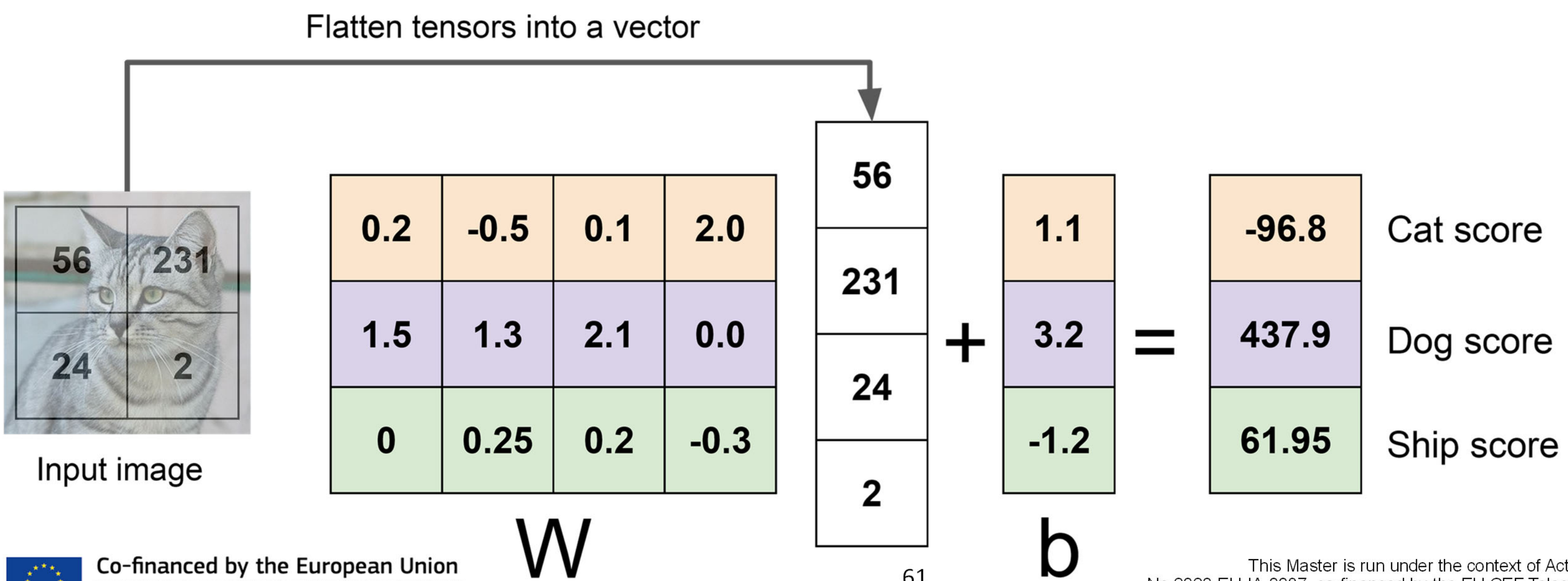
Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Parametric Approach*

Example with an image with 4 pixels, and 3 classes (cat/dog/ship)

Algebraic Viewpoint



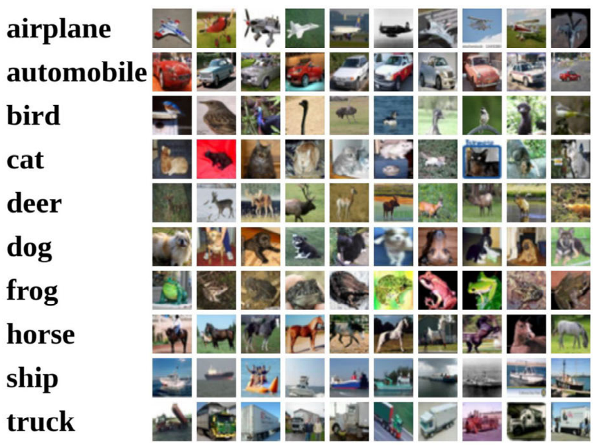
Co-financed by the European Union
Connecting Europe Facility

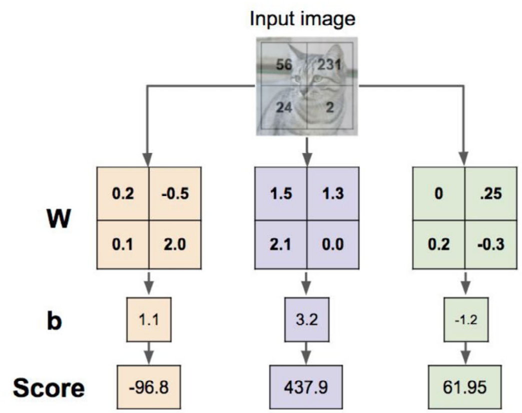
This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Parametric Approach*

Interpreting a Linear Classifier

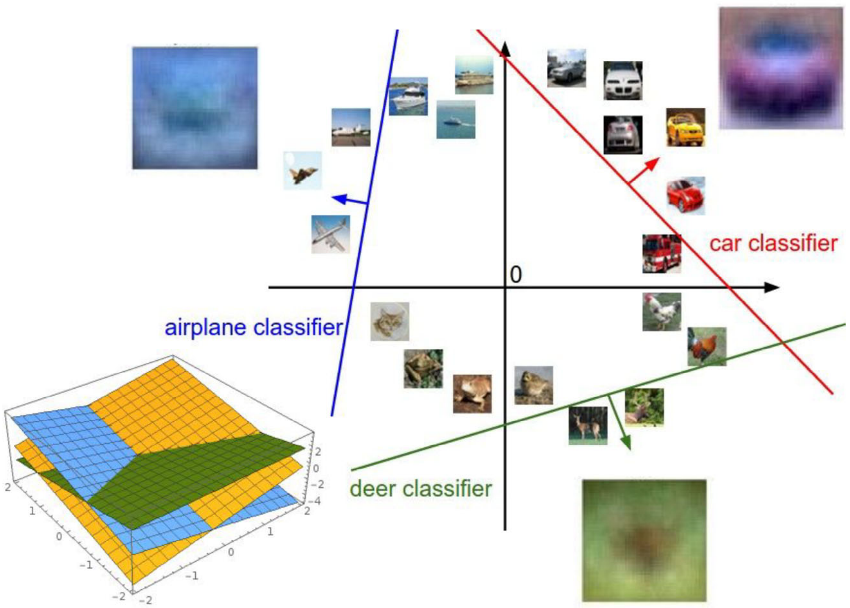
Visual Viewpoint





Linear Classifier: *Parametric Approach*

Interpreting a Linear Classifier: Geometric Viewpoint



$$f(x,W) = Wx + b$$

Array of **32x32x3** numbers
(3072 numbers total)

Plot created using Wolfram Cloud

Linear Classifier

Hard cases for a linear classifier

Class 1:
First and third quadrants

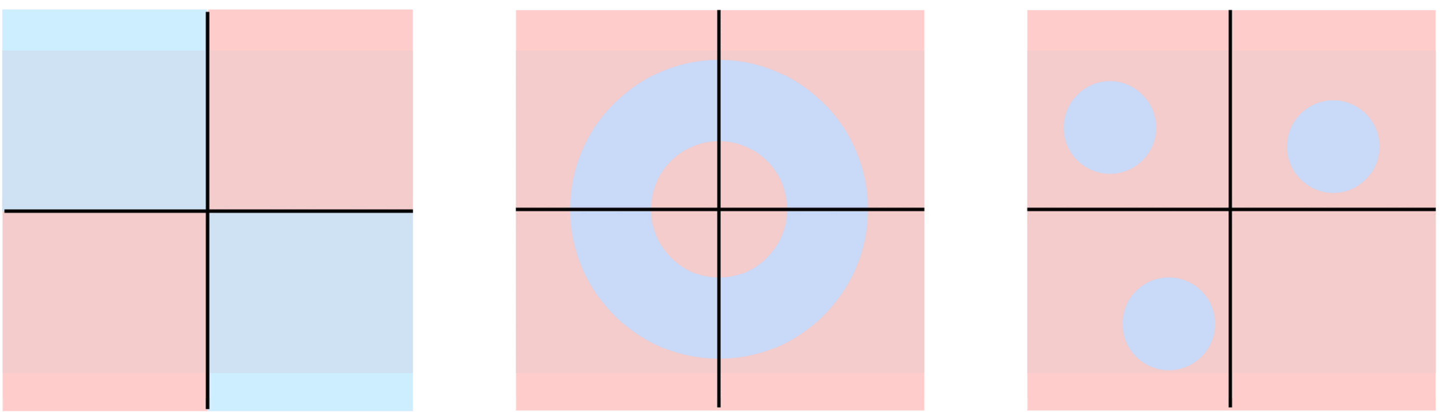
Class 2:
Second and fourth quadrants

Class 1:
1 <= L2 norm <= 2

Class 2:
Everything else

Class 1:
Three modes

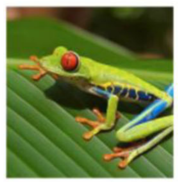
Class 2:
Everything else



Linear Classifier: *SVM*

Linear Classifier – Choose a good W





airplane	-3.45	-0.51	3.42
automobile	-8.87	**6.04**	4.64
bird	0.09	5.31	2.65
cat	**2.9**	-4.22	5.1
deer	4.48	-4.19	2.64
dog	8.02	3.58	5.55
frog	3.78	4.49	**-4.34**
horse	1.06	-4.37	-1.5
ship	-0.36	-2.09	-4.79
truck	-0.72	-2.93	6.14

TODO:

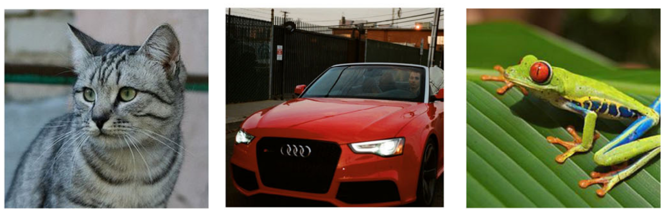
1. Define a **loss function** that quantifies our unhappiness with the scores across the training data.
2. Come up with a way of efficiently finding the parameters that minimize the loss function. **(optimization)**

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *SVM*

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x,W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

A **loss function** tells how good our current classifier is

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^{N}$$

Where x_i is image and
y_i is (integer) label

Loss over the dataset is a average of loss over examples:

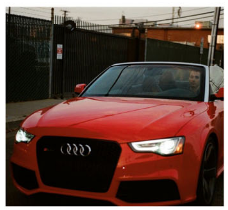
$$L = \frac{1}{N} \sum_i L_i(f(x_i, W), y_i)$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *SVM*

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:


cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

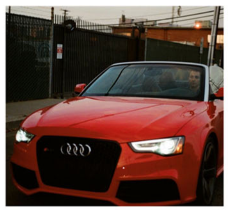
and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases}$$
$$= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Linear Classifier: *SVM*

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:


cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9		

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

= max(0, 5.1 - 3.2 + 1)
+max(0, -1.7 - 3.2 + 1)
= max(0, 2.9) + max(0, -3.9)
= 2.9 + 0
= 2.9

Linear Classifier: *SVM*

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

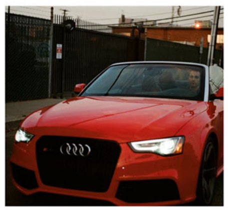
the SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

= max(0, 1.3 - 4.9 + 1)
+max(0, 2.0 - 4.9 + 1)
= max(0, -2.6) + max(0, -1.9)
= 0 + 0
= 0

Linear Classifier: *SVM*

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

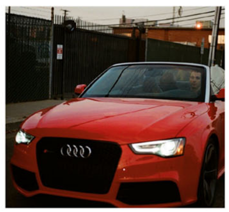
the SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

= max(0, 2.2 - (-3.1) + 1)
+max(0, 2.5 - (-3.1) + 1)
= max(0, 6.3) + max(0, 6.6)
= 6.3 + 6.6
= 12.9

Linear Classifier: *SVM*

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example (x_i, y_i) where x_i is the image and where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

Loss over full dataset is average:

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i$$

L = (2.9 + 0 + 12.9)/3
= **5.27**

Linear Classifier: *SVM*

Multiclass SVM loss:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$

cat	1.3
car	**4.9**
frog	2.0
Losses:	0

Q1: What happens to loss if car scores decrease by 0.5 for this training example?

Q2: What is the min/max possible SVM loss L_i?

Q3: At initialization W is small so all s ≈ 0. What is the loss L_i, assuming N examples and C classes?

Q4: What if the sum was over all classes? (including j = y_i)

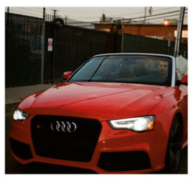
Q5: What if we used the mean instead of sum?

Q6: What if we used $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)^2$

Linear Classifier: *SVM*

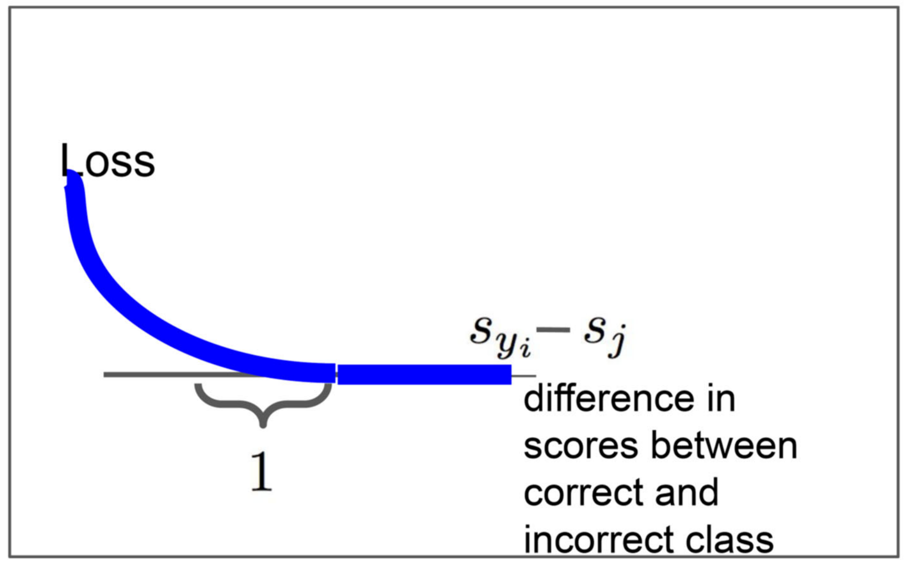
Multiclass SVM loss:

$$\boxed{L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)}$$



cat	1.3
car	**4.9**
frog	2.0
Losses:	0

Multiclass SVM loss:



Q6: What if we used $L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)^2$

Linear Classifier: *Softmax*

Softmax Classifier (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$\boxed{s = f(x_i; W)}$$

$$\boxed{P(Y = k|X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}}$$ Softmax Function

$$L_i = -\log P(Y = y_i|X = x_i)$$

	Unnormalized log-probabilities / logits	exp →	Probabilities must be >= 0: unnormalized probabilities	normalize →	Probabilities must sum to 1: probabilities
cat	**3.2**		**24.5**		$\boxed{\textbf{0.13}}$
car	5.1		164.0		0.87
frog	-1.7		0.18		0.00

→ $L_i = -\log(0.13)$
$= \mathbf{2.04}$

Maximum Likelihood Estimation
Choose weights to maximize the likelihood of the observed data

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Softmax*

Softmax Classifier (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$\boxed{s = f(x_i; W)}$$

$$\boxed{P(Y = k|X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}}$$ Softmax Function

	Unnormalized log-probabilities / logits		unnormalized probabilities		probabilities		Correct probs
			Probabilities must be >= 0		Probabilities must sum to 1		
cat	**3.2**	exp →	**24.5**	normalize →	$\boxed{\mathbf{0.13}}$	→ compare ←	**1.00**
car	5.1		164.0		0.87		0.00
frog	-1.7		0.18		0.00		0.00

Kullback–Leibler divergence

$$D_{KL}(P\|Q) = \sum_y P(y) \log \frac{P(y)}{Q(y)}$$

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Softmax*

Softmax Classifier (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

$$P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax Function

Probabilities must be >= 0

Probabilities must sum to 1

	Unnormalized log-probabilities / logits	exp →	unnormalized probabilities	normalize →	probabilities	→ compare ←	Correct probs
cat	**3.2**		**24.5**		$\boxed{\mathbf{0.13}}$		**1.00**
car	5.1		164.0		0.87		0.00
frog	-1.7		0.18		0.00		0.00

Cross Entropy

$$H(P, Q) =$$
$$H(p) + D_{KL}(P \| Q)$$

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Softmax*

Softmax Classifier (Multinomial Logistic Regression)

Want to interpret raw classifier scores as **probabilities**

$$\boxed{s = f(x_i; W)}$$

$$\boxed{P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}}$$

Softmax Function

Maximize probability of correct class

$$L_i = -\log P(Y = y_i | X = x_i)$$

Putting it all together:

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

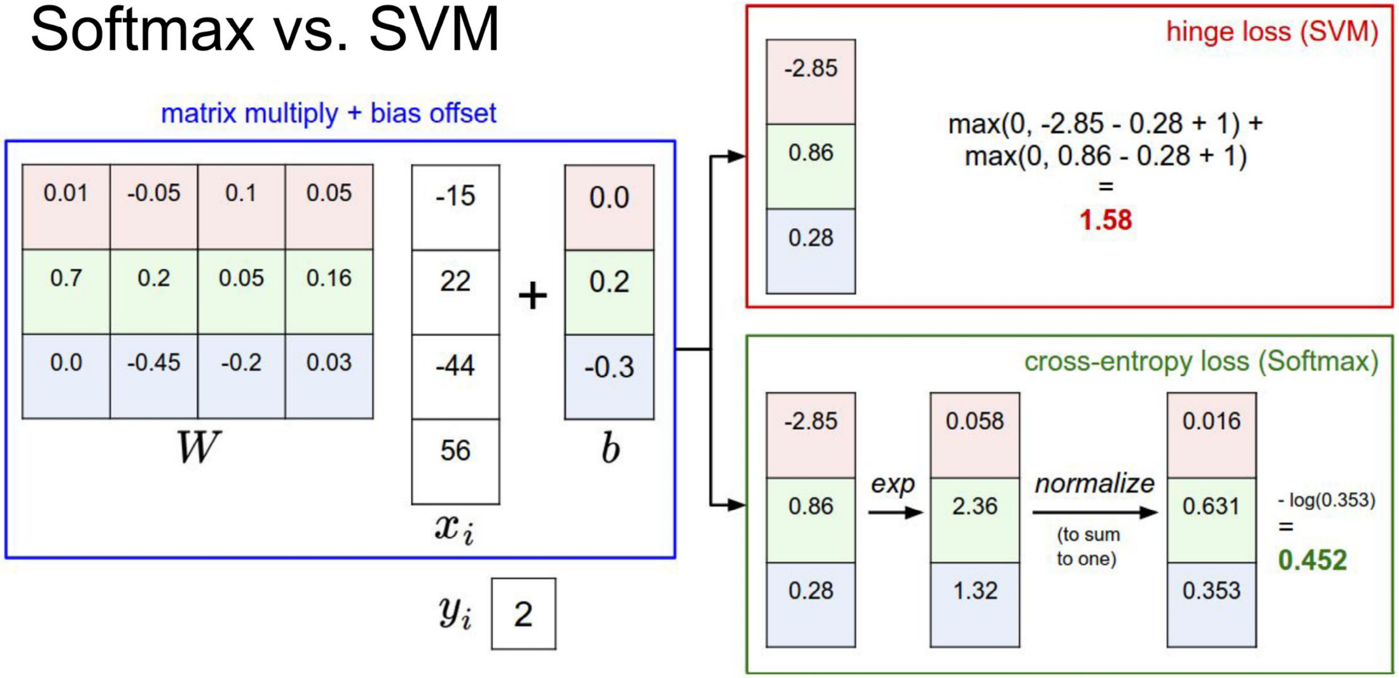
cat	**3.2**
car	5.1
frog	-1.7

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

Linear Classifier: *Softmax*

Softmax vs. SVM



$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

Next Courses

- Brief discussion on regularization & optimization techniques
- Image Classification with CNNs
  - Training, Visualizing and Understanding
- Object Detection and Image Classification
  - Recurrent Neural Networks
  - Attention and Transformers

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423

MAI4CAREU | Master programmes in Artificial Intelligence 4 Careers in Europe

Research in Deep Camera

Alessandro Artusi
Team Leader
DeepCamera Group

email: a.artusi@cyens.org.cy

Research Interests:

Machine Learning, Deep Learning and its applications in Computer Vision, High Dynamic Range Imaging, Image Processing applied on Computer Graphics and Color Science

https://www.cyens.org.cy/en-gb/research/pillars-groups/visual-sciences/deep-camera/people/alessandro-artusi/

Deep Camera

CYENS CENTRE OF EXCELLENCE

https://deepcamera.cyens.org.cy/about-us/

deepcamera.ai@gmail.com

+357 227 475 81

Dimarchias Square 23 STOA, Nicosia
Nicosia, Nicosia 1016, Cyprus

Thank you!

See you next week

Co-financed by the European Union
Connecting Europe Facility

This Master is run under the context of Action No 2020-EU-IA-0087, co-financed by the EU CEF Telecom under GA nr. INEA/CEF/ICT/A2020/2267423