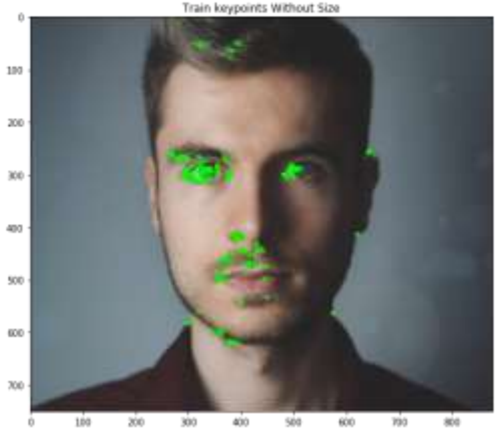
Computer Vision

Exploration & feature engineering

Plan



- ➢ Data quality
  - ○ Visualize
  - ○ Aspect ratio and size
  - ○ Label composition
  - ○ Datasets comparison
  - ○ Normalization & transformation
- ➢ Feature engineering
  - ○ SIFT
  - ○ ORB
- ➢ Dimensionality reduction
  - ○ PCA
  - ○ t-SNE

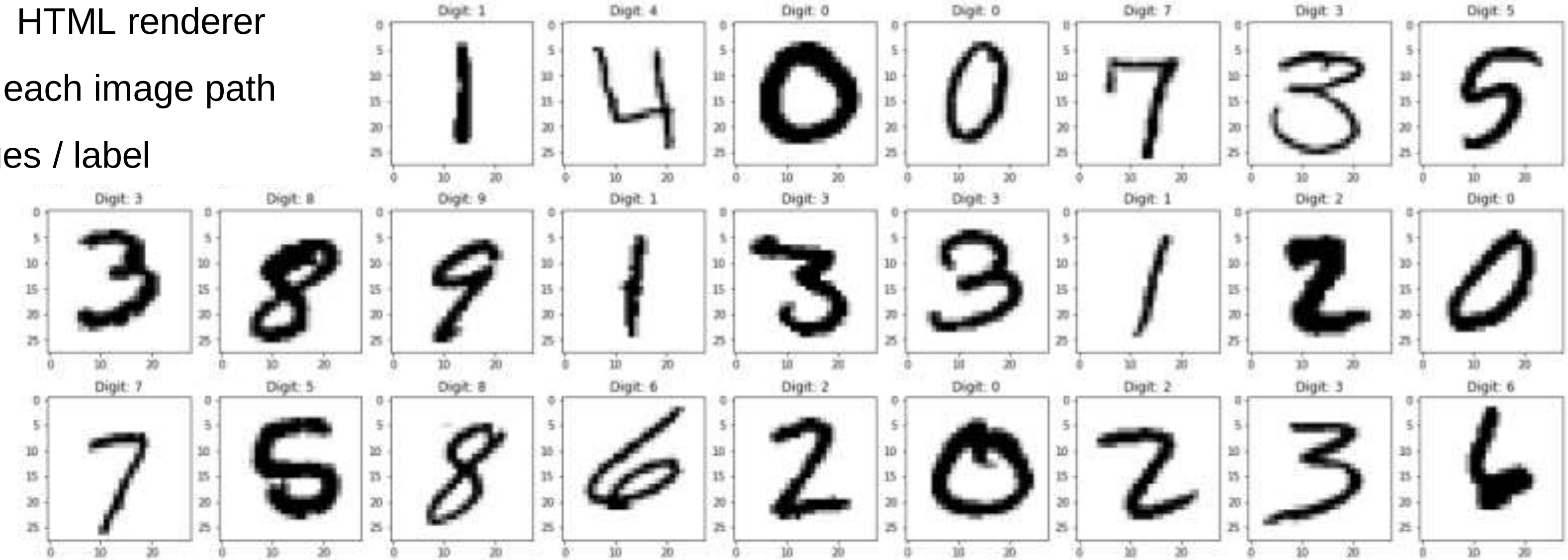
Data quality

- Can I do my job with it ?
- Can I get around with it ?

Visualize

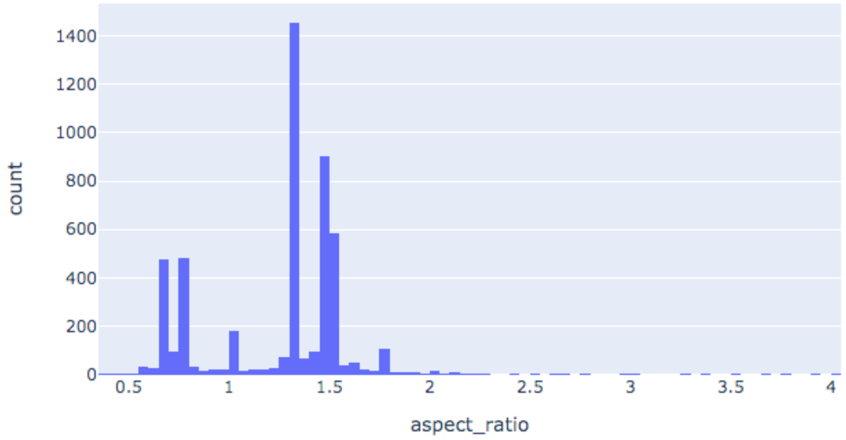
- ➢ Missing labels (mixed datasets)
- ➢ Wrong labeling
- ➢ Outliers

- ➢ Visual inspection
  - ○ Matplotlib / openCV
  - ○ HTML renderer
- ➢ Test each image path
- ➢ Images / label



Aspect ratio and size

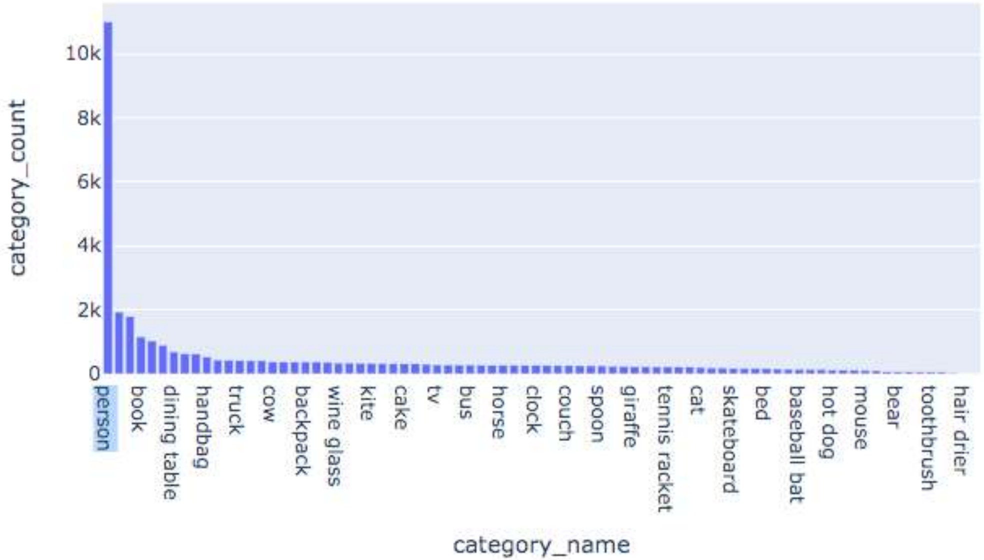
- Size or definition: W X H (1600 X 900)
- Aspect ratio: W / H (16 / 9)
- Distribution (total and per label)
  - Unimodal: resizing or not
  - Bimodal: resizing
  - Multimodal: batch training or else
- Anchor box size per label



- Destructive resizing
- Padding method
- Visual inspection

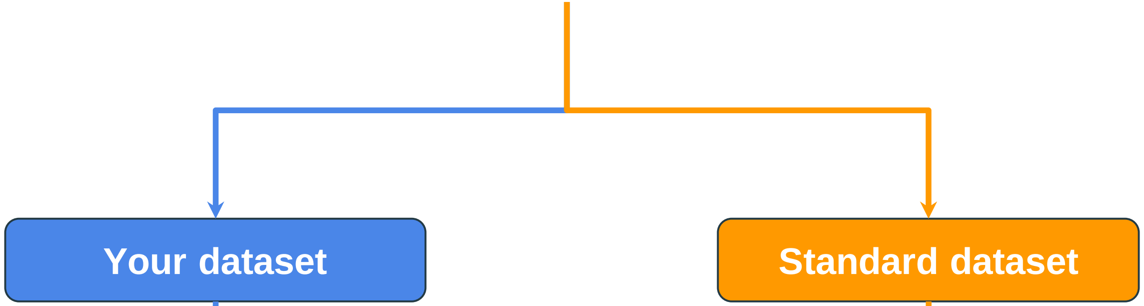
Label composition

- Oversample
- Downsample
- Update dataset
- Balance model weights



Datasets comparison

- ➢ What's your target model (architecture) ?
- ➢ Best performance obtained on which dataset ?



- ➢ Image sizes
- ➢ Aspect ratios
- ➢ Anchor box sizes

Normalisation & transformation

Increase training speed

- ➢ **Pixel Normalization**: pixel values range 0-1.
- ➢ **Pixel Centering**: pixel values have a zero mean.
- ➢ **Pixel Standardization**: pixel values have a zero mean and unit variance (gaussian distrib).
- ➢ Per image / batch / **dataset**

Increase efficiency and robustness

- ➢ **Histogram equalization**: improve feature appearance by increased contrast
- ➢ **Crop / zoom / rotations**: generate more or less possibilities
- ➢ **Denoising**: improve feature detection

- ➢ This is all theory
- ➢ Look for proofs

Feature Engineering

Algorithms

Feature notions

Characteristics

- ➢ Feature = interesting image zones
- ➢ Characteristics:
  - ○ Repeatable
  - ○ Distinct
  - ○ Local (minimum neighbors impact)
- ➢ Based on gradient disruption

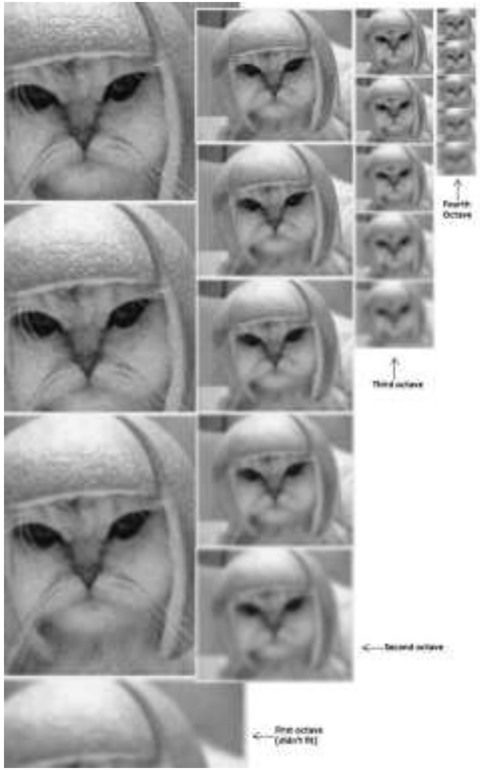
Algorithms

- ➢ Detectors: HARRIS, FAST
- ➢ Descriptors: BRIEF
- ➢ Both: SIFT, ORB, Conv layer

Scale Invariant Feature Transform (SIFT)

Feature Detector

- ➢ Scale space



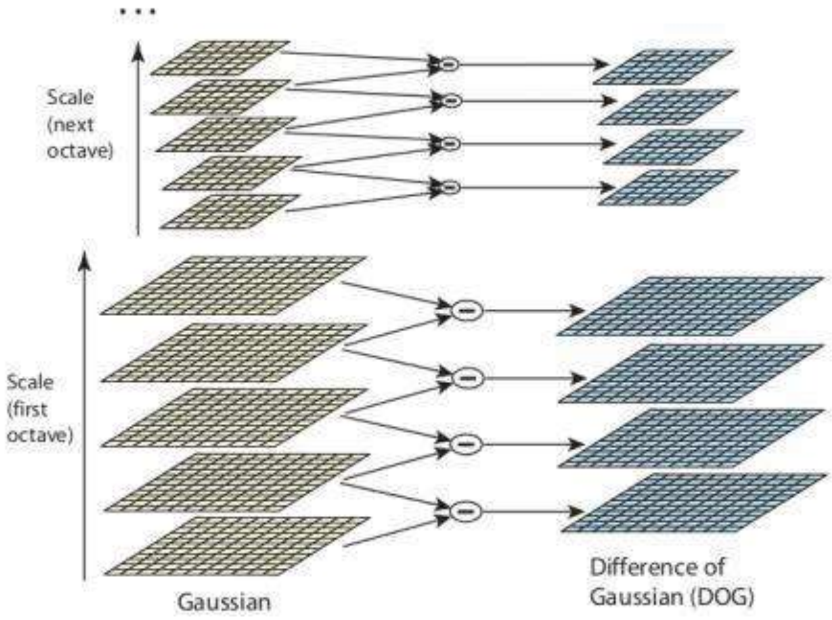
Scale Invariant Feature Transform (SIFT)

Detector

- ➢ Scale space
- ➢ Keypoints (extremes in Gaussian differences)

Descriptor

- ➢ Vectors of n dimensions
- ➢ Orientation
- ➢ Brightness normalisation
- ➢ Feature matching

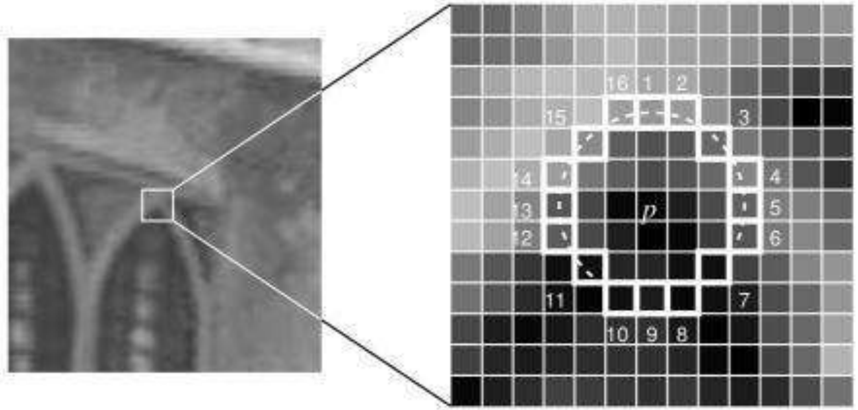


Oriented FAST and Rotated BRIEF (ORB)

- ➢ Open source
- ➢ Fastest
- ➢ Most efficient

Detector (FAST)

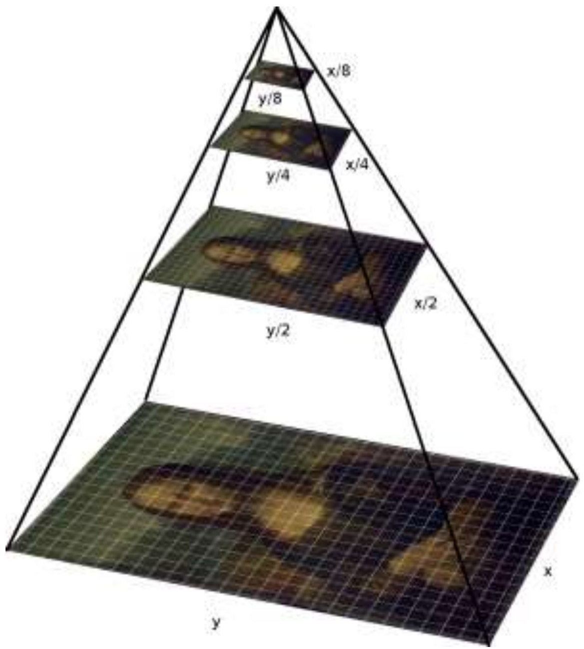
- ➢ > 8 pixels darker / lighter = feature



Oriented FAST and Rotated BRIEF (ORB)

Detector (FAST)

- > 8 pixels darker / lighter = feature
- Orientation based on multiscale pyramid



Oriented FAST and Rotated BRIEF (ORB)

Descriptor (BRIEF)

- Binary vector of 128 to 512 bits
- Feature matching

Dimensionality Reduction

Why and how ?

Purpose

- 1 pixel = 1 variable
- 64 x 64 pixels = **4096** variables !
- Features detectors = too many variables !

Can we get rid of some of them ?

Code exemple: https://www.kaggle.com/hamishdickson/preprocessing-images-with-dimensionality-reduction

Variance

- ➢ How far from the mean
- ➢ Low = all points are similar
- ➢ High = Very different points

Assumptions:

- ➢ Background and object = high variance
- ➢ I can delete some pixels and keep high variance

Example

Vector(10): 2, 3, 3, 3, 4, **4**, 4, 5, 5, **6**

Variance = s^2 = 1.4333333
Mean = 3.9

Vector(9): 2, 3, 3, 3, 4, 4, 5, 5, 6

Variance = s^2 = 1.6111111
Mean = 3.9

14.3%

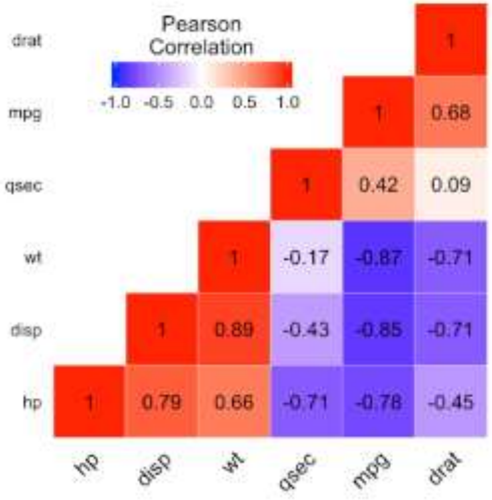
Vector(9): 2, 3, 3, 3, 4, 4, 4, 5, 5

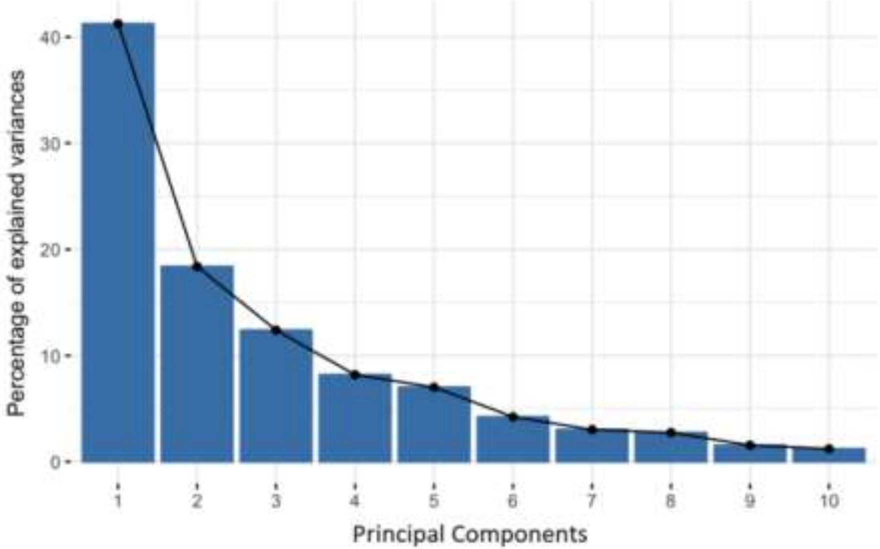
Variance = s^2 = 1
Mean = 3.7

28.6%

Principal Component Analysis

- Reduce the number of variables of a data set, while preserving as much information as possible
- Requires **dataset pixel standardization**





t-SNE

Steps:

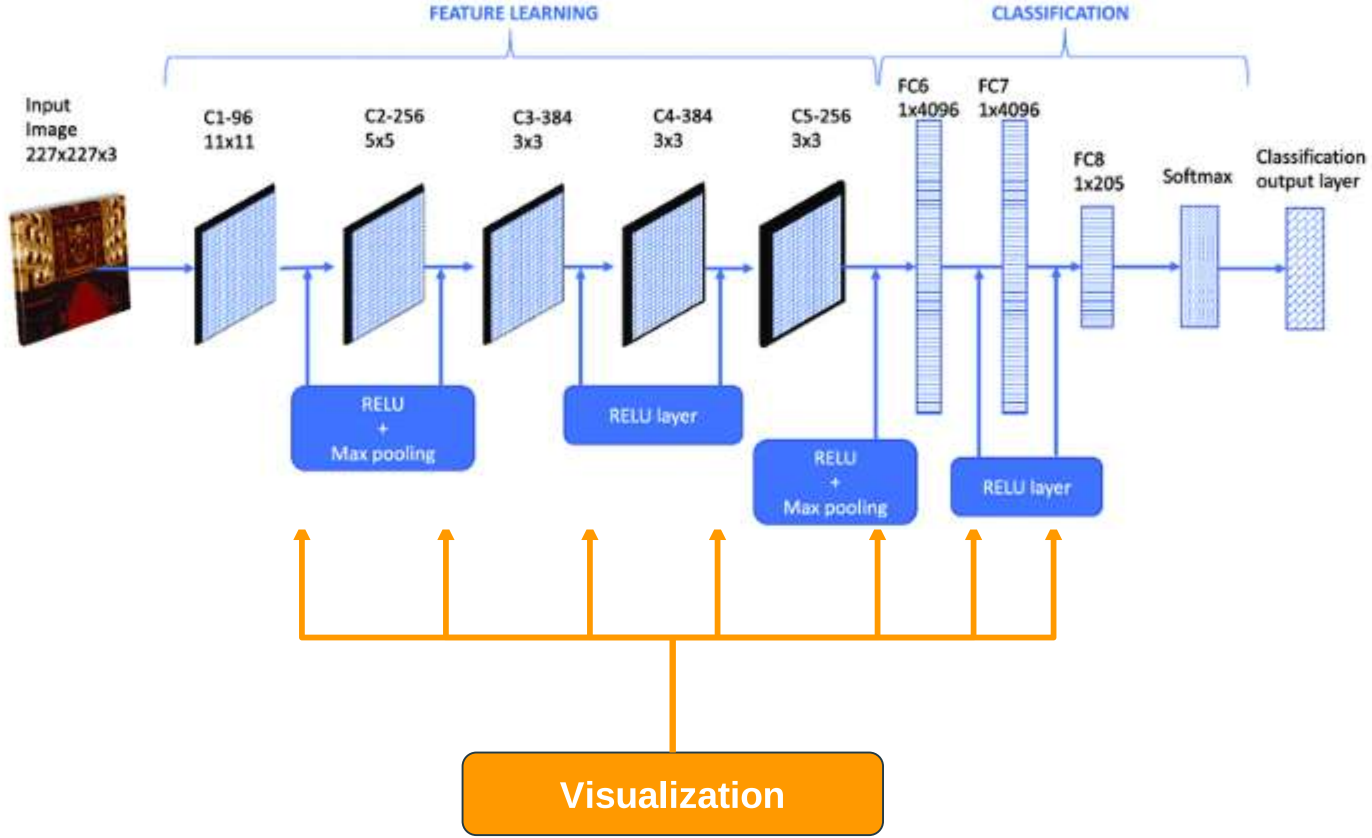
- ➢ Probability of difference in high dimension
- ➢ Probability of difference in low dimension

- ➢ Hyperparameters
- ➢ Computationally expensive

UMAP:

- ➢ Slightly better projection than t-SNE
- ➢ Much faster

Use Case


FEATURE LEARNING
CLASSIFICATION
Input
Image
227x227x3
C1-96
11x11
C2-256
5x5
C3-384
3x3
C4-384
3x3
C5-256
3x3
FC6
1x4096
FC7
1x4096
FC8
1x205
Softmax
Classification
output layer
RELU
+
Max pooling
RELU layer
RELU
+
Max pooling
RELU layer
Visualization

Exercices

Real cases

Quizz Kahoot !

- ➢ Link: https://kahoot.it/
- ➢ Pin: **08400066**

Ai design: dataset / concept

- Select a dataset or an app idea
- Create your team !
- Define your strategy
- Prototype and test