MLRF Lecture 05

J. Chazalon, LRDE/EPITA, 2019

Image classification overview

Lecture 05 part 02

Instance recognition *vs* Class recognition

Instance recognition:
Re-recognize a known 2D or 3D rigid object, potentially being viewed from a novel viewpoint, against a cluttered background, and with partial occlusions.

Ex: practice session 3

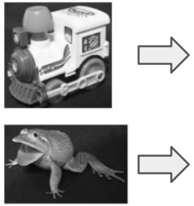
Class recognition:
Recognize any instance of a particular general class such as “cat”, “car”, or “bicycle”.

Aka *category-level* or *generic* object recognition.

More challenging.

This lecture and next practice session.

D. Lowe 2004



“Toy train”

“Toy frog”

Our focus today (and for next practice session)

Image classification

Aka category-level recognition

Aka generic object recognition

Aka category recognition

Aka “is this a muffin or a chihuahua”?

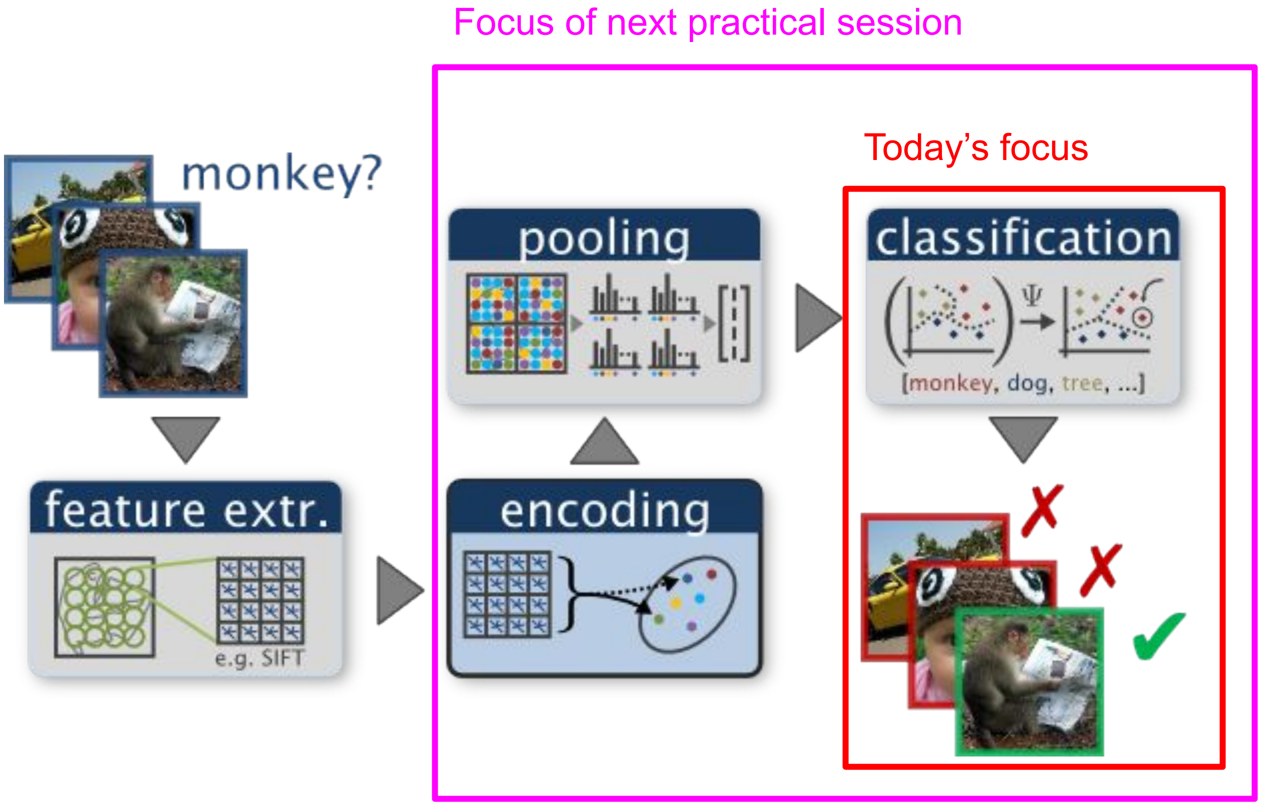
“Toy train”

“Toy frog”

“Can eat”

“Cannot eat”

Pipeline overview



http://www.robots.ox.ac.uk/~vgg/research/encoding_eval/

Our image classification pipeline

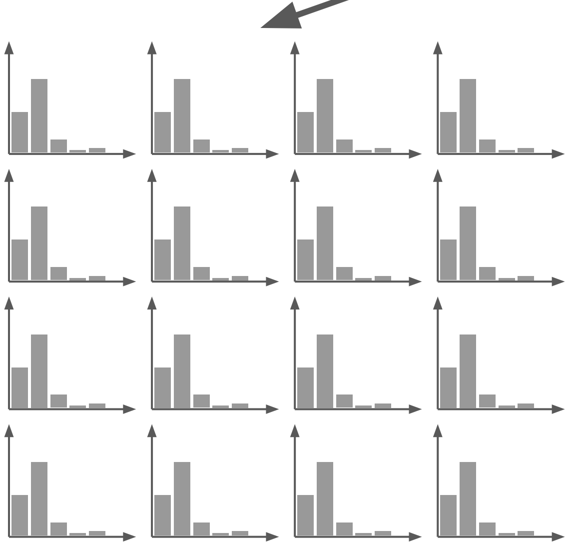
This is a **supervised** machine learning task.

⇒ We need a dataset with **samples** and **target values** (ground truth)

MUF	MUF	MUF	MUF
MUF	MUF	MUF	MUF
CHI	CHI	CHI	CHI
CHI	CHI	CHI	CHI

Our image classification pipeline

Images will be represented as **BoVW vectors** of **fixed size**.

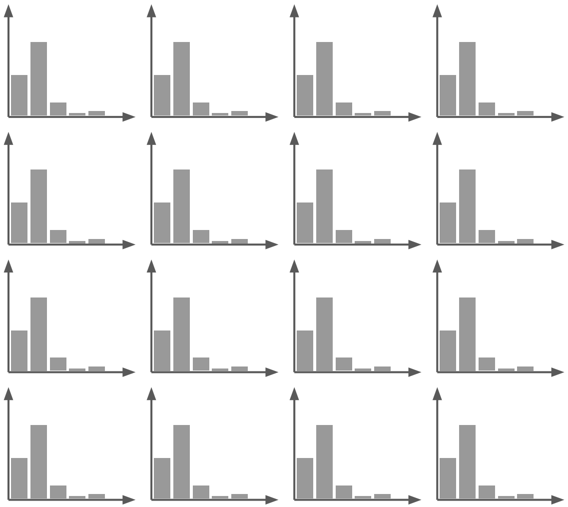


MUF	MUF	MUF	MUF
MUF	MUF	MUF	MUF
CHI	CHI	CHI	CHI
CHI	CHI	CHI	CHI

Our image classification pipeline

Images will be represented as **BoVW vectors** of **fixed size**.

Targets will be **encoded as <u>integers</u>**.

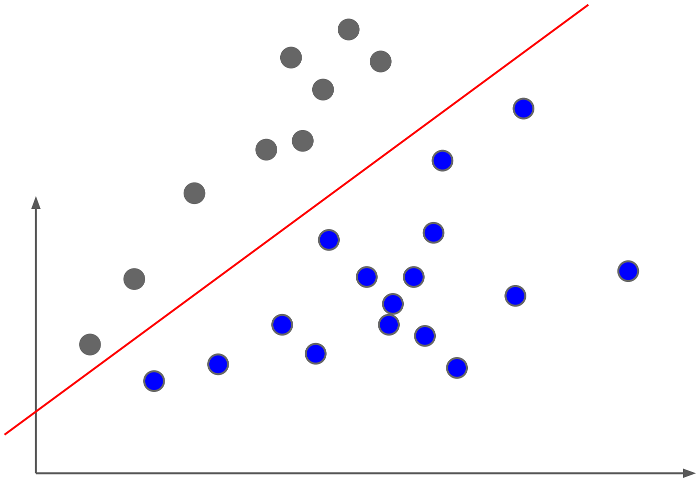


0	0	0	0
0	0	0	0
1	1	1	1
1	1	1	1

Our image classification pipeline

This is a very usual data representation for a classification problem.

Classifier inputs = “samples” with “features” / Classifier outputs = “labels”



Now we just need to select an appropriate method, prepare our data, run some training, test the results, adjust some parameters, compare approaches, display results…

Data preparation

NumPy formatting

one sample

$$X = \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix} \qquad y = \begin{pmatrix} 1.6 \\ 2.7 \\ 4.4 \\ 0.5 \\ 0.2 \\ 5.6 \\ 6.7 \end{pmatrix}$$

one feature

outputs / labels

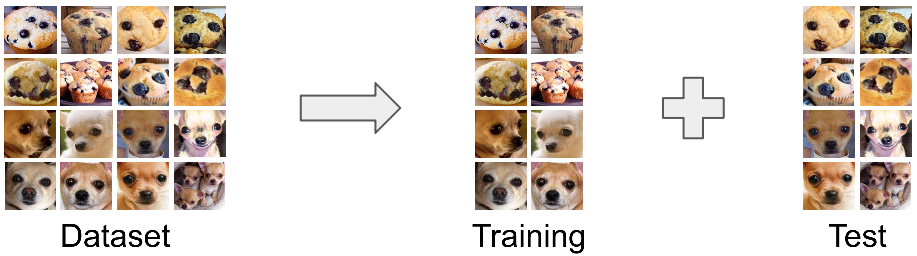
from A. Müller

Training/validation/test separation

More on that later in this lecture.

For now just remember that:

- **You <u>cannot estimate</u> the <u>generalization performance</u> of your predictor/estimator/classifier on its <u>training set</u>** (makes sense, right?)
- So you need to keep some samples aside for later evaluation
  Do not use them during training!
- “Validation” another separate set used to tune parameters



training set

$$X = \begin{pmatrix} 1.1 & 2.2 \\ 6.7 & 0.5 \\ 2.4 & 9.3 \\ 1.5 & 0.0 \\ 0.5 & 3.5 \\ 5.1 & 9.7 \\ 3.7 & 7.8 \end{pmatrix} \quad y = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

test set

Other “funny” things to do IRL

Collect data

Clean data

Check data

Clean again

Annotate

Check

Compute / convert / scale features...

“Data curator” is a new job title.

Manual annotation drives crazy.

Many “data *something*” jobs.

Feature selection

Feature selection

Consists in dropping some data columns.

Can help later stages:

- Less data to process
- Better properties (like decorrelated features, etc.)

Which columns?

- Hard problem in general
    - Because features may be informative **as a group**
- Some simpler and helpful techniques:
    - Remove features with low variance
    - Dimensionality reduction techniques are not exactly feature selection, but can have a similar effect