Hes·so
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences
Western Switzerland

MSE | MASTER OF SCIENCE IN ENGINEERING

Multimodal Processing, Recognition and Interaction

Final Challenge Description

Simon Ruffieux

Elena Mugellini, Jean Hennebert, Stefano Carrino

Summary

- Introduction
- Overview
- Objectives
- Planning
- Rules
- Database
- Data Formats
- Hints
- Practical Info

Introduction

- My thesis is about evaluation of multimodal gesture recognition systems
- The proposed dataset has been acquired in this context
  - Multimodal dataset for research
  - Multiple possible recognition tasks
- The OPEGRA platform has been developed in this context
  - Facilitate download of datasets
  - Facilitate evaluation and comparison of algorithms

Overview of challenge

- You receive a labeled dataset containing data of segmented gestures

- You must implement machine learning algorithms for the task of recognizing gestures

- You must optimize your algorithm

Objectives of the challenge

- Apply the theoretical concepts seen in class with real data

- Obtain the best recognition rate !

- This challenge represents 30% of your final grade!

Planning

- 04.05: Start of Challenge
  - Group composition
  - Access to platform & data
- 11.05 & 18.05: Development & optimization of algorithms
  - No theory, we are available via mail for questions
- 31.05: End of challenge
  - Your results must be on the platform before 23H55
    - Warning, evaluation might take some time
- 01.06: Presentations & Prices
  - Powerpoint presentations
  - Nomination of winner(s)
- 08.06: Submission of reports

Rules

- 9 Groups of 3 or 2 persons
  - Each group must implement
    - One Hidden Markov Model (HMM)
    - One Support Vector Machine (SVM)
    - One Neural Network (NN)
    - [Optional] Fusion method

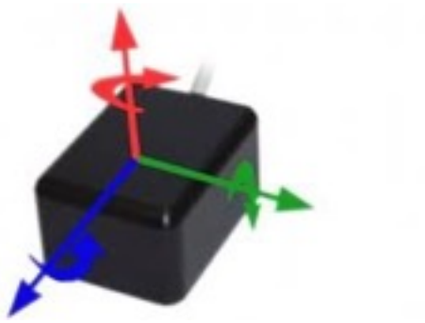
- Language: C# or Matlab
  - A group may use different software/library/language

Database (I)

- 9 subjects
- 11 right-arm gestures (=classes)
- ~1100 instances of gestures
  - ~100 instances per class

- Dataset provided as
  - .mat for Matlab
  - .dset for C# (a library is provided for loading/ reading)

Kinect upper-skeleton



4 Xsens inertial motion unit

Database (II)

- Gestures:
  - Performed while seating
  - Variable duration amongst
    - different classes
    - a same class
- Sensors
  - Not the same frequency
  - Partially redundant information

#	Name of gesture
0	Calibration*
1	Swipe left
2	Swipe right
3	Push to screen
4	Take from screen
5	Palm-up rotation
6	Palm-down rotation
7	Draw a circle I
8	Draw a circle II
9	Wave hello
10	Shake hand

* This class has fewer samples

Database (III)

- You receive the Training set (75%)
  - The data is labeled for supervised training
    -> Split it as you wish to train & optimize your algorithm
    2 sets, cross-validation, k-fold, etc.

- The Test set is kept on the platform (25%)
  - This data is not labeled, otherwise it has the same format as the training data

Data Format – INPUT (I)

Xsens data

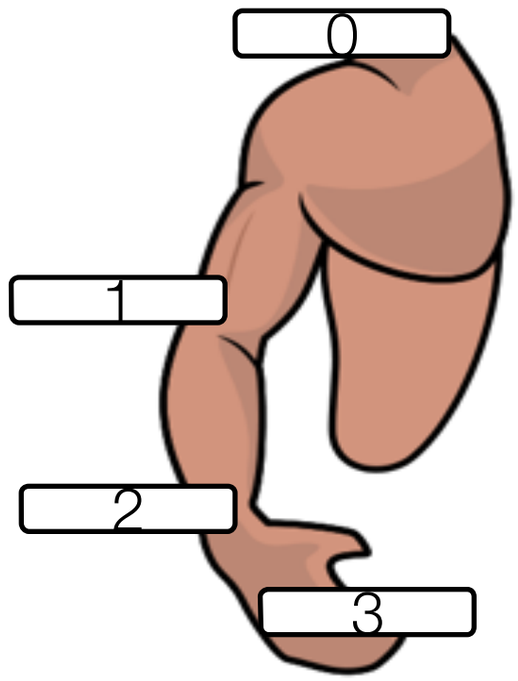
17 features per sensor:

LinAccX LinAccY LinAccZ …
AngVelX AngVelY AngVelZ …
Yaw Pitch Roll …
MagX MagY MagZ …
Barometer …
QuatX QuatY QuatZ QuatW

4 sensors attached to the user

-> A total of 68 features

4 IMU (xsens) located at the following positions on the right arm of subjects

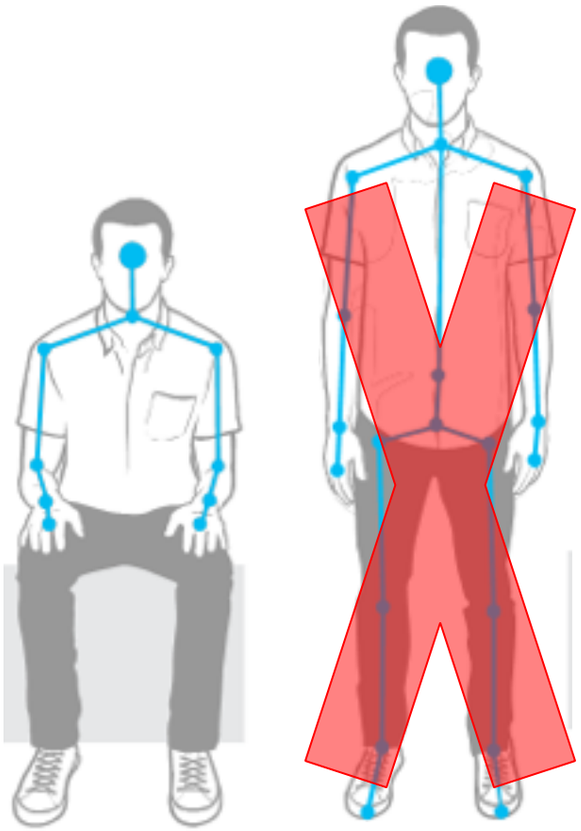


Data Format – INPUT (II)

Kinect data

A total of 84 features:

TrackingState posX posY posZ …
Foreach BodyPart:
[TrackingState posX posY posZ]*



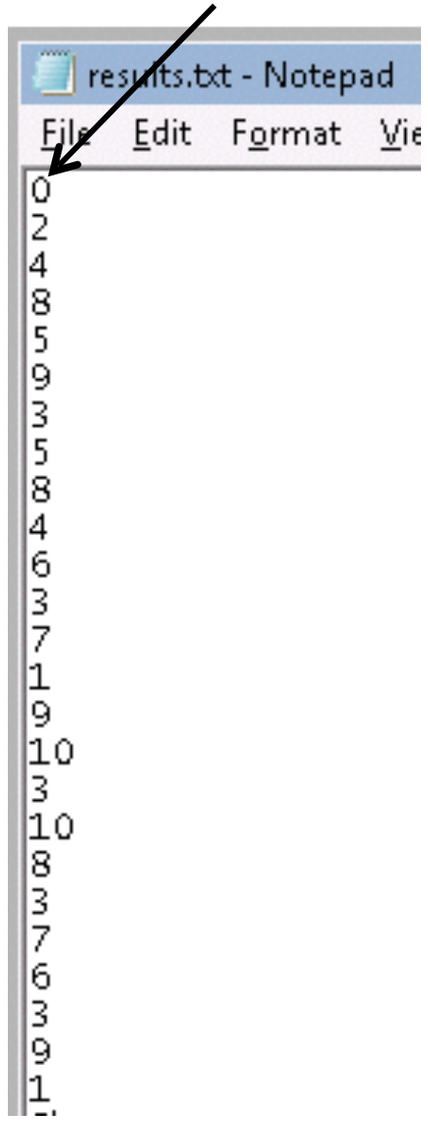
Body parts:

HipCenter Spine ShoulderCenter **Head ShoulderLeft ElbowLeft WristLeft HandLeft ShoulderRight ElbowRight WristRight HandRight** HipLeft KneeLeft AnkleLeft FootLeft HipRight KneeRight AnkleRight FootRight

Data Format – OUTPUT

- Classification results (Evaluation set)
  - Output your results in ‘results.txt’ file in the working directory of your algorithm
    - *One label per line*
- *See platform tutorial and provided sample algorithms for more details*

Label (integer: 0 to 10)



Detailed Task

- Load dataset and process it into features
- Implement a machine learning algorithm
- Split data (train/test, cross-validation, etc.)
- Optimize your performances
  - Search for optimal set of features
  - Search for best parameters

- At the end, upload your algorithm on the platform for the official evaluation
  - You may upload it several times (keep the same name)

Hints

- Think about features!
  - What features make more sense or provides more information ?
  - More features != better
  - Processing features may take time!
  - How to represent temporal aspect of gestures ?

- Pay attention to algorithm used
  - HMM != SVM != NN

Practical Info - Final Handouts

- A Report and a Presentation per group
  - Introduction
  - For each algorithm
    - Implementation
    - Description of the features
    - An evaluation of obtained performances
    - Discussion
  - Comparison of the results
  - General discussion
  - Conclusion

Individual

Practical Info – Presentations

- The day of the presentations (01.06.2015)
  - Presentation (~2 hours)
    - 15 minutes / group
    - Every participant should present its part
  - Summary Challenge results (~15 minutes)
    - We will present the winners of the challenge

Practical Info – Grading

- Grading is also individual !
- Criteria
  - Algorithm performances (weighting = x2)
    - F1-Score (see provided pseudo-code)
  - Quality of presentation
  - Quality of report
    - Choices and justifications
      - Algorithms, features, etc.
  - Amount of work

Practical Info – Results Submission

- If you encounter problem during evaluation on the platform, do not hesitate to contact me by mail !
- If you follow precisely the instructions, it should be fine
  - Labels in ‘results.txt’ (0-10)
  - Your algorithm must terminate at the end of the evaluation
- An algorithm that does not terminate after 15 minutes is killed
- Do not upload and evaluate your algorithm at the last minute!
  - An evaluation might take some time to finish

Practical Info – F1-Score

- $CM = Confusion\ Matrix$
- $Precision(i) = \frac{CM.GetDiagonal(i)}{CM.GetColumn(i).Sum()}$
- $Recall(i) = \frac{CM.GetDiagonal(i)}{CM.GetRow(i).Sum()}$
- $MacroPrecision = Precision.Average()$
- $MacroRecall = Recall.Average()$

$$\boldsymbol{F1 - Score} = \frac{2 * MacroPrecision * MacroRecall}{MacroPrecision + MacroRecall}$$

Sokolova, M. and Lapalme, G.A systematic analysis of performance measures for classification tasks. *Information Processing & Management 45*, 4 (2009), 427–437.

Contacts

- Simon Ruffieux
  - simon.ruffieux@hefr.ch
- Stefano Carrino
  - stefano.carrino@hefr.ch