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

Challenge 2020

Simon Ruffieux

Elena Mugellini, Stefano Carrino, Omar Abou Khaled

Summary

- Objectives
- Introduction
- Planning
- Rules
- Dataset
- Data formats
- Hints

Objectives

- Apply the theoretical concepts seen in class with real data
- Obtain the best possible accuracy to recognize driving condition
- Resume your work in a report and present it to the class

This challenge represents **30%** of your final grade !

Introduction

- Steps of the challenge

1. You receive a labeled dataset with physiological information about persons driving a car in a simulated environment

2. Implement an algorithm to recognize the *condition* of the car driver

3. Optimize the results of your algorithm

Detailed Tasks

- Load dataset and features
  - Feature engineering (on EDA and *ECG** signals)
  - Prepare data for your algorithm
- Implement a machine learning algorithm
- **Optimize your performances**
  - Search for optimal set of features
  - Search for best hyper parameters

- At the end, you will evaluate your algorithm using unseen Test data and send us your results as a structured text file

* High-level ECG features are already provided

Planning

- 17.11 **Start of Challenge**
  - Start developing & questions
  - Access to train dataset (already avail. – 27.10.2019)
- 24.11 Theory + **Development**
  - Available by mail/Teams (simon.ruffieux)
- 1.12 **Development** & optimization of algorithms
  - No class, we are available via mail/Teams for questions
- 10.12 **Development** & optimization of algorithms
  - No class, **presence upon request** (last request time 09.12)
  - Access to test dataset
- 13.12 (23h55) **End of Challenge**
  - **Submit** your predictions obtained on test set via Moodle
- 15.12 **Presentations**
  - **Submit** your presentation via Moodle
- 22.12 (23h55) **Report**
  - **Submit** your report and code via Moodle

Rules

- Groups of 2 persons
    - Each group must implement (2 out of 3)
        - One Hidden Markov Model (HMM)
        - One Support Vector Machine (SVM)
        - One Random Forest (RF)
        - Late Fusion of algorithms (Fusion)
- Language: Python
    - Note: A group may use different software/library/language – if it is the case come to discuss with us
- *NOTE: one algorithm per student
    - 2 persons -> 2 distinct algorithms

Rules (II)

Even if you are working in a group

the evaluation is individual !

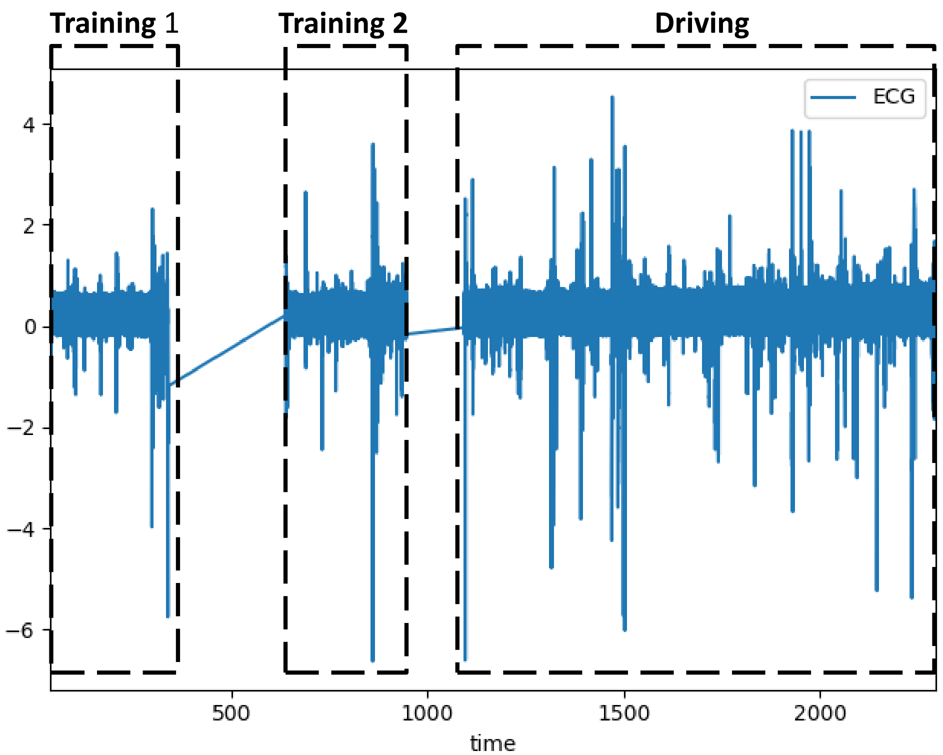
Dataset – Overview

- The dataset contains data recorded in the context of the PhD thesis of Quentin Meteier
  - Various sensors have been used to record physiological data during driving sessions performed with a car simulator
  - Subjects were experiencing Level-3 autonomous driving. That means that they could perform secondary tasks (instead of driving) but had to take back the control of the system when dangerous events occurred (animals crossing, objects on the road, etc.)

- We labeled the driving **conditions/classes** as
  - Driving normally with **N**o **S**econdary **T**ask (NST = Class 0)
  - Driving while performing the **S**econdary **T**ask (ST = Class 1)

Dataset – Overview (II)

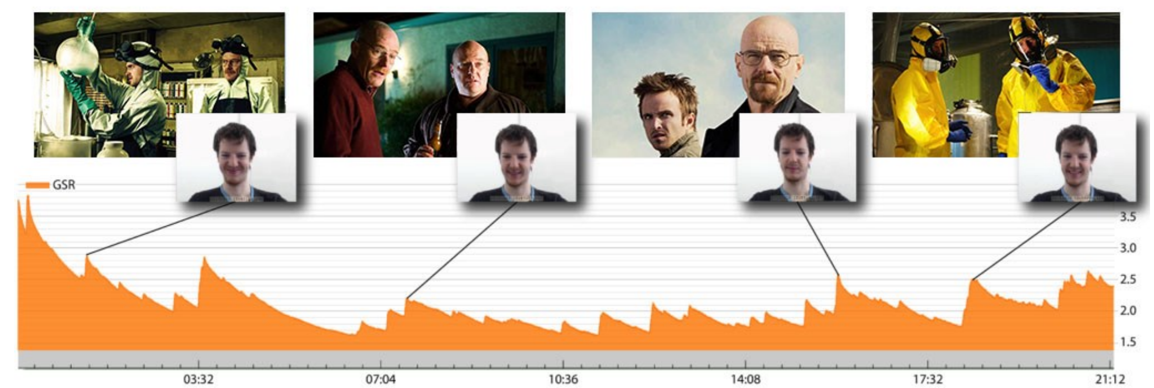
- Secondary Task (ST, NST)
  - Sequential number subtracting (9843, 9840, 9837, etc.)
    - A cognitive task inducing mental workload and stress
- Experiment design
  - **Training 1** session (5 min) -> **Baseline**
    - Subject was tasked to watch Level-3 autonomous driving
  - **Training 2** session (5 min) -> **Discard**
    - Subject was tasked to drive the car
  - **Driving** session (20 min)
    - Subject was task to watch Level-3 autonomous driving and react to take-over requests (=events)
      - ✓ 50% of subjects with ST (ST), 50% of subjects without ST (NST)



Training 1 session is used to collect the uniq
physiological signals of each individual in nor
conditions !
→ Baseline

Dataset - Sensors

- **Galvanic Skin Response** (*GSR, EDA, SC*)
  - measure the sweat gland activity, related to emotional arousal
    - ✓ stress, excitement, engagement, frustration, and anger



- **Electrocardiogram** (*ECG*)
  - measure the electrical activity of the heart
    - stress, emotions, biometry

- ~~Respiration sensor~~ (not provided for challenge)
  - Measures the frequency and amplitude of respiration

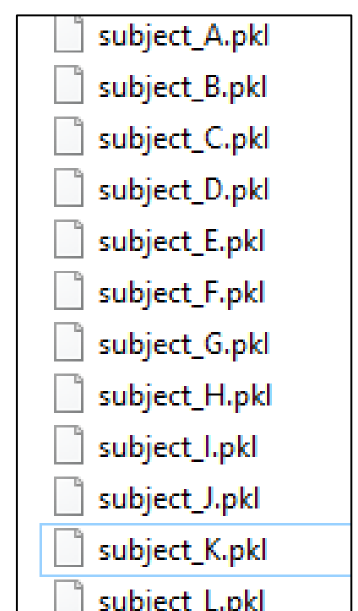
Dataset - Features

- **Galvanic Skin Response** (GSR, aka EDA or SC)
  - Provided: raw data only
- **Electrocardiogram** (ECG)
  - Provided: engineered features (HRV indicators) !
    - BPM and IBI shall not be used in the context of this challenge !

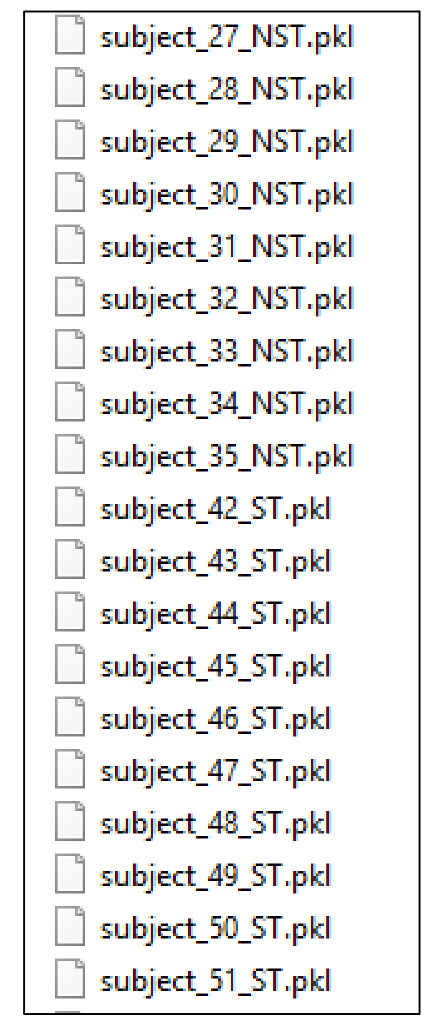
Indicator	Description	Formula
BPM (Beats per minute)	Amount of heart beats per minute	
IBI (inter-beat interval)	Mean distance of intervals between heartbeat	$\frac{1}{N-1}\sum_{i=1}^{N} RR_i$
SDNN	Standard deviation of intervals between heartbeats	$\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(RR_i - \overline{RR})^2}$
SDSD	Standard deviation of successive differences between adjacent R-R intervals	$\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(RR_{diff\ i} - \overline{RR_{diff}})^2}$
RMSSD	Root mean square of successive differences between adjacent R-R intervals	$\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(RR_{diff\ i})^2}$
NN50	Number of successive differences between adjacent R-R intervals greater than 50ms	
pNN50	Percentage of NN50 over all intervals	

Dataset - Division

- The datasets has been prepared for you
  - “Trainset” and “Testset” have the same format
    - Except <Test set> does not contain label in its name!
  - <u>Training set</u>:
    - Given at the beginning of the challenge
    - Balanced (same amount of each class)
    - 70 drivers
  - <u>Test set</u>:
    - Given at end of challenge
    - 12 drivers



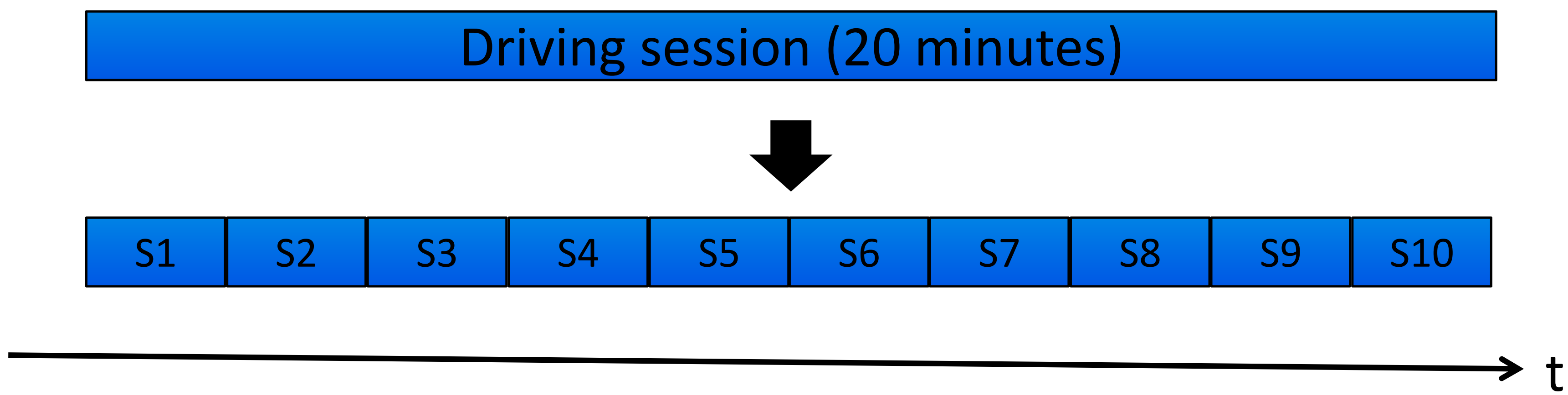
Testset



Trainset

Dataset – Additional infos

- Temporal segmentation
  - Test sessions can be divided in multiple short temporal segments to gain/maintain temporal information (**Allowed levels: [1, 2, 4, 5, 10, 20]**)



    - Question: Does segmenting improve accuracy ?
  - **Warning:**
    - HMM require temporal inputs, therefore it must use segmented data !
- Use the data from the "Training 1" session to reduce inter-subjects differences !
- The dataset is quite small, take it into account !

Output Data Format

- Submit your results before the deadline !
  - One file per algorithm
    - *AlgoName_GroupName_LastName*.csv

```
np.savetxt("RF_Teachers_Ruffieux.csv", results.astype(int), fmt='%i')
```

  - One file per group for late fusion
    - Fusion_*GroupName*_Fusion.csv

- Format
  - 12 integers (1 per subject), one integer per line

```
results = np.array([0, 1, 0, 0, 1])
np.savetxt("RNG_Teachers_Ruffieux.csv", results.astype(int), fmt='%i')
```

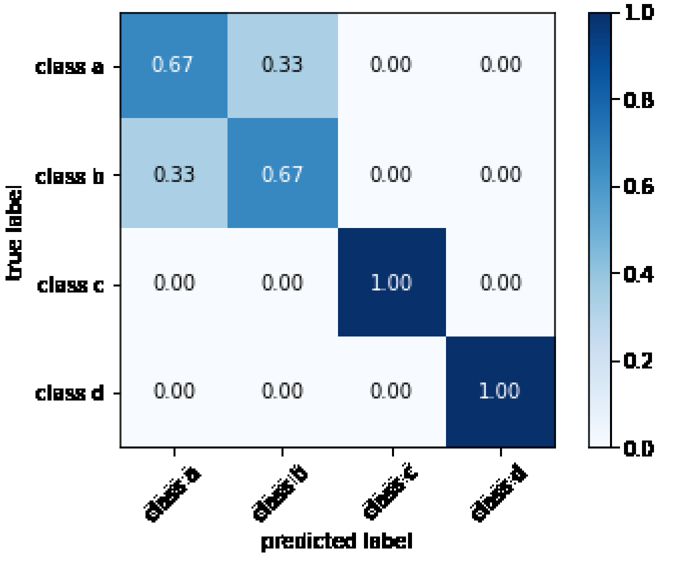
The results <u>must</u> be outputted in the same order as the subjects received in the Test set

Evaluation Metrics

- Macro F1-Score

```
Score = f1_score(labels, predictions, average='macro')
```

- Confusion matrix
  - Use it to analyze results
  - Show them in your report/presentation



Starting Material

All this material is available on Moodle as a <*.zip*> file !

- **Skeleton Python code**
  - Load training data
  - Plot the ECG raw signal data
  - ECG feature engineering (95% given)
    - You still need to take advantage of the baseline …
  - EDA very basic feature engineering example (5% given)
  - Sample generic algorithm implementation
  - Write predicted results to file
- **Example** of prediction **output file** (with random values)
- **Training set** as dataframes (*Trainset.zip*) [400 MB]
- **Test set** as dataframes, without labels (*Testset.zip*) [70 MB]
    - Will be made available later (10.12) !

General Hints

- Start with basic data/features
  - Get a first working pipeline with minimalistic features and test it on a validation set before doing more advanced stuff
- Feature engineering
  - EDA is raw data
    - What type of higher level features can be extracted from the signal ?
  - ECG high-level features are already extracted
    - You may want to take advantage of the baseline to improve them …
- Optimize
  - Explore features and features combinations
  - Find best hyper-parameters
- More advanced score estimation
  - Traditional method for leave-one-out score estimation
    - Try rotating drivers used for learning and drivers used for validation to get a better estimation of your final score (= cross-validation for LOO)
- What Fusion mechanism will you use ?

Report/Presentation Hints

- One report/presentation per team
  - Each person present his/her own algorithm
  - Presentation of 5 minutes per person
    - teams of 3-> 15 minutes
    - teams of 2-> 10 minutes)
- Describe your specific algorithm architecture
  - Input features, layers, states, kernel, hyper-parameters, fusion, etc.
- Describe your feature engineering
  - What operations did you perform on features ?
- Describe your optimization methodology
  - Grid-search, manual, etc. ?
- Describe your evaluation methodology
  - Leave-one-out , other approaches ?
- Describe and **analyze** your results
  - Macro F1-score, confusion-matrix
  - Log your results during your experimentations to show the improvement

Algorithm Hints - SVM

- Use “sklearn.svm.SVC” or “sklearn.svm.LinearSVC”
  - Features are most important
    - Maintain some temporal information ?
  - Play with the hyper-parameters (kernels, C, …)

The SVM algorithm is simple to implement for classification, but tuning the hyper-parameters is crucial.

Nevertheless, do not forget a sound evaluation methodology and feature engineering

Hints - HMM

- Use “hmm-learn” framework
  - Use the temporally segmented data (2+)
  - One HMM model per label

    (Split training_set per label before training algorithm…)
  - Start with ergodic HMM models
  - Investigate the number of states to maximize performances

  - Other types of HMM models possible or meaningful?
  - Look for more advanced features

  - Warning - Explainability / logic can be hard to grasp while applying HMM to the challenge dataset

Hints - RF

- Use “sklearn.ensemble RandomForestClassifier” class
  - Features are most important
    - Maintain some temporal information ?
  - Only few hyper-parameters

Random Forest algorithm is simple to implement for classification, thus try focusing on evaluation methodology and feature engineering (and hyper-parameters)

Questions ?

- 17.11 **Start of Challenge**
  - Start developing & questions
  - Access to train dataset (already avail. – 27.10.2019)
- 24.11 Theory + **Development**
  - Available by mail/Teams (simon.ruffieux)
- 1.12 **Development** & optimization of algorithms
  - No class, we are available via mail/Teams for questions
- 10.12 **Development** & optimization of algorithms
  - No class, **presence upon request** (last request time 09.12)
  - Access to test dataset
- 13.12 (23h55) **End of Challenge**
  - **Submit** your predictions obtained on test set via Moodle
- 15.12 **Presentations**
  - **Submit** your presentation via Moodle
- 22.12 (23h55) **Report**
  - **Submit** your report and code via Moodle