Intelligent Systems Project

A.Y. 2021-2022

Michele Baldassini, Beatrice Lazzerini, Francesco Pistolesi

1. Overview

Pulse Transit Time (PTT) is the time that pulse waves require to travel in blood vessels between two sites. It is an important indicator for many medical properties and recent research focused on its potential for blood pressure monitoring.

A total of 22 participants were involved in the data collection while performing three different activities ('sit', 'walk', 'run'). The dataset contains 66 recordings. A recording contains 11 time series (also *signals* in the following), each representing a physiological signal for more than 40,000 heartbeats.

2. Dataset

The data are in CSV (comma-separated-value) format.
Each recording consists of 2 files, **sXX_YYYY_timeseries** and **sXX_YYYY_targets** whose structures are described below:

sXX_YYYY_timeseries ('s' means *subject,* 'XX' is the subject number (1-22), and 'YYYY' is an activity among 'sit', 'walk', and 'run').

Each file **sXX_YYYY_timeseries** is organized in columns. The first column contains a timestamp. Each of the other columns contains a time series corresponding to a physiological signal. Signals are as follows:

- pleth_1: red wavelength PPG from the distal phalanx (first segment) of the left index finger palmar side (sampling rate 500 Hz)
- pleth_2: infrared wavelength PPG from the distal phalanx (first segment) of the left index finger palmar side (sampling rate 500 Hz)
- pleth_3: green wavelength PPG from the distal phalanx (first segment) of the left index finger palmar side (sampling rate 500 Hz)
- pleth_4: red wavelength PPG from the proximal phalanx (base segment) of the left index finger palmar side (sampling rate 500 Hz)
- pleth_5: infrared wavelength PPG from the proximal phalanx (base segment) of the left index finger palmar side (sampling rate 500 Hz)
- pleth_6: green wavelength PPG from the proximal phalanx (base segment) of the left index finger palmar side (sampling rate 500 Hz)
- lc_1: load cell proximal phalanx (first segment) PPG sensor attachment pressure (sampling rate 80Hz)

- lc_2: load cell (base segment) PPG sensor attachment pressure (sampling rate 80Hz)
- temp_1: distal phalanx (first segment) PPG sensor temperature (°C, sampling rate 10Hz)
- temp_2: proximal phalanx (base segment) PPG sensor temperature in (°C, sampling rate 10 Hz)
- temp_3: ambient temperature (°C, sampling rate 500 Hz)

Each file **sXX_YYYY_targets** contains the target time series. The first column contains a timestamp. The other contains the target time series:

- ecg: 3-channel ECG sampled at 500 Hz

Note that the data recorded by subject 13 are particularly noisy during the walking activity.

3. Requirements (Part 1)

3.1 Estimating *ecg* using neural networks

The aim of this part of the project is to design and develop two multi-layer perceptron (MLPs) artificial neural networks that estimate the ecg of a person (XX) during the three activities (YYYY), respectively, based on sensor data stored in the **sXX_YYYY_timeseries** file.

The MLPs take as input a set of features extracted and/or selected from the sensor data (signals in **sXX_YYYY_timeseries**), and return the value of two features extracted from the ecg time series (mean and standard deviation).

Extracting features is a critical step, as it aims to find the best way of representing the information that the network receives as input. Considered a signal s, you can extract a set F of statistical features from the entire signal as shown in Fig. 1, i.e., from the entire set of signal samples that make it up. Let vector $f_s \in \mathbb{R}^{|F|}$ contain the values of the features in F for signal s. The signal is thus represented by $|F|$ features. If this extraction is performed on all the signals, the network will have $k|F|$ inputs, where k is the number of signals.

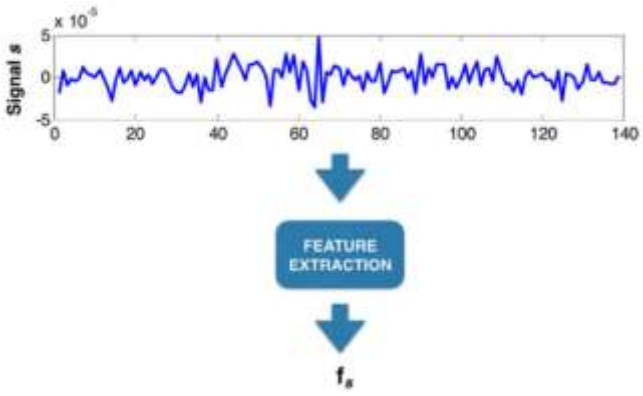

Figure 1. Extracting features from the entire signal.

An alternative way of extracting features is to first partition the samples of signal s into more contiguous/overlapped windows $w_1^s, \dots, w_N^s$, and then extract the same set F of features from the samples of each of these windows. This procedure leads to matrix $\boldsymbol{F_s} = [\boldsymbol{f_{s,1}}, \dots, \boldsymbol{f_{s,N}}] \in \mathbb{R}^{N \text{ x } |F|}$, where column vector $\boldsymbol{f_{s,i}} \in \mathbb{R}^{|F|}$ contains the values of the features in F for window i of signal s, with $i \in \{1, \dots, N\}$. In this case, the network receives N x $|F|$ inputs (features) per signal. Graphical examples of feature extraction from contiguous and overlapped windows are shown in Fig. 2a and Fig. 2b.

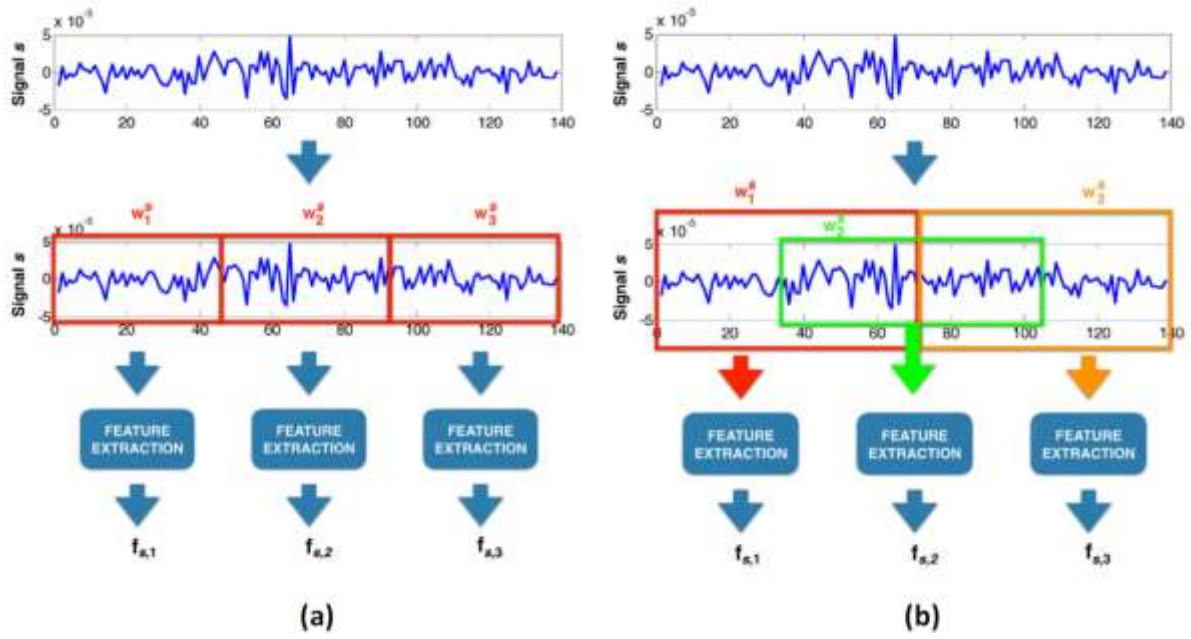

Figure 2. Extracting features from three contiguous windows (a), and from three overlapped windows (b).

The features to extract can be chosen from both the time and frequency domains. Remember that a signal *s* in the time domain can be represented as the sum (or integral) of sine waves at different frequencies, each representing a frequency component of *s*, as shown in Fig. 3.

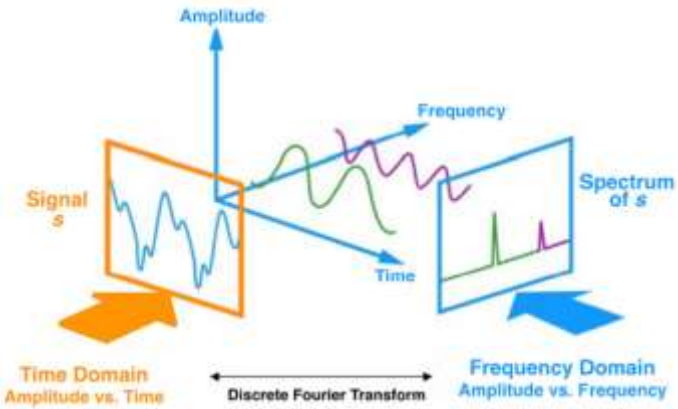

Figure 3. Signals represented in the time (orange square) and frequency (blue square) domains. The peaks in the frequency domain indicate the frequencies of the sine waves that form the spectrum of s.

The set of extracted features should be reduced by selecting the most significant features to predict the output. One way is to use the sequential feature selection (implemented by the *sequentialfs* MATLAB function), with an MLP as a criterion function that assesses the accuracy of each subset of features in estimating each target. The suggested maximum number of features to select is 10 for each network. Once the search for the best set of features is completed, the next step is to find the best architecture for both MLPs (i.e., the number of hidden layers and the number of neurons in each of these layers).

The last step of this part is to design and train two radial basis function (RBF) networks that do the same thing as the previously developed MLPs.

3.2 Determining a person's activity using neural networks

The aim of this part of the project is to design and develop a multi-layer perceptron (MLP) that classifies a person's activity among 'sit', 'walk' and 'run'. The MLP takes as input a set of features and returns the corresponding activity.

The set of features to use consists of the union of those selected and used to train the networks developed in Section 3.1. The goal is to find the best architecture for the MLP, i.e., the number of hidden layers and the number of neurons in each of these layers.

3.3 Fuzzy inference system

The aim of this part is to design and develop a fuzzy inference system to classify a person's activity using the *k* most relevant features ($k \leq 5$) in the set of features used to train the classifier developed in section 3.2. Fig. 4 shows a fuzzy system that uses $k = 3$ features.

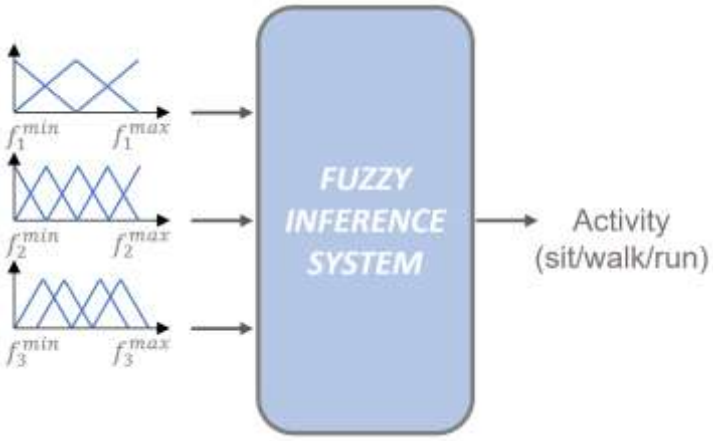

Figure 4. The fuzzy inference system.

All the variables used will have to undergo an appropriate fuzzy modeling.

4. Requirements (Part 2)

4.1 Improve *ecg* estimation using convolutional neural networks

The aim of this part of the project is the same as in Section 3.1, with the exception of using a convolutional neural network (CNN) instead of an MLP. The value to be estimated (mean/standard deviation) is the one that achieved the worst performance using the MLP in section 3.1. The goal is to find the best CNN architecture along with the best hyperparameter values (filter size, number of filters, etc.). Fig. 5 shows an example of one-dimensional CNN (1D CNN) applied to univariate time series analysis.

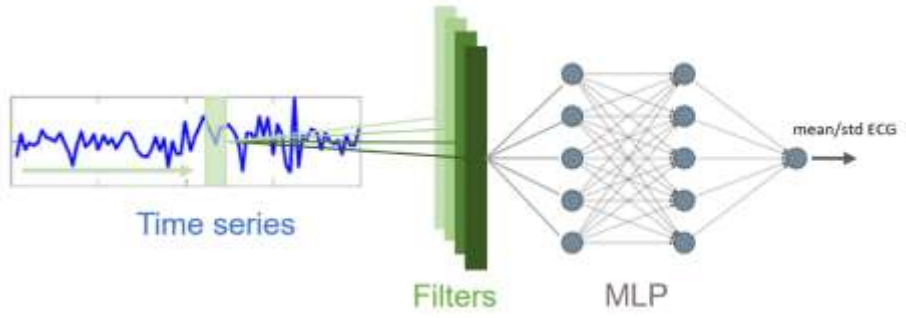

Figure 5. Overview of the 1D-CNN applied to univariate time series analysis.

4.2 Predict *ecg* value(s) using recurrent neural networks

The aim of this part of the project is to design and develop a recurrent neural network (RNN) that predicts one value of a person's ecg, based on part or all of the signals in Section 2. The RNN takes these signals at time steps $(t-k, \dots, t)$ as input along with the corresponding ecg values, and returns the ecg value at time step $t+1$. The goal is to find the best architecture of the RNN, along with the best hyperparameter values (size of the time window, etc.). Fig. 6 shows an example of RNN performing the forecasting of a univariate time series.
The RNN can be implemented as a standard RNN, a GRU or an LSTM.

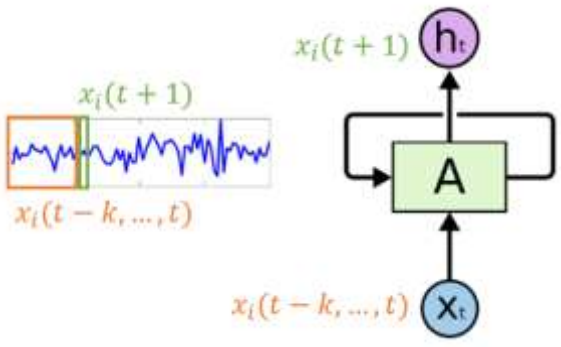

Figure 6. Overview of univariate time series forecasting using RNN.

4.3 Multi-step forecasting of *ecg* values

The aim of this part of the project is to design and develop an RNN that predicts a set of consecutive future values of a person's ecg.
The goal is to find the best architecture of the RNN, along with the best hyperparameter values.

Tasks to be carried out

This project can be carried out by groups of up to three students.

The optional tasks for each type of group are as follows:

- one-member groups: **the last step of 3.1, 4.3**
- two-member groups: **4.3**