ALFA BK UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGIES
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
International Scientific Conference

UDK:004.421.6:[602:61
DOI:10.46793/ALFATECHproc25.220R
Original scientific paper

PPG signal analysis and wavelet selection for feature extraction

Zlatko RADOVANOVIĆ[1], Stevan JOKIĆ[2], Ivan JOKIĆ[3], Branislav GERAZOV[4], Ana KOVAČEVIĆ[5], Nenad GLIGORIĆ[6]

Abstract: Biomedical signals or biosignals are spatial, temporal or spatio-temporal records of a biological phenomenon. Biosignals contain information that is of great importance for understanding the specific physiological mechanisms of biological phenomena or the systems from which they originate. Signals provide important information about the internal state of organs, response to external stimuli, general state, general state of health and the state of various other parameters and are an indispensable part of modern medical diagnostic practice. The signal that arises as a result of the device's action on the organism, the subject of this paper, is the signal obtained by the photoplethysmography (PPG) method. Certain artificial intelligence methods were used to analyze the PPG signal. In this paper, analysis using wavelet transformation will be used, and a special focus will be on the selection of wavelets (wavelets) that will be used for the purpose of machine learning.

Keywords:characteristic parameters, diastolic peak, photoplethysmography (PPG), pulse width, machine learning, systolic peak, wavelet transform

1 INTRODUCTION

The biomedical signals that are the subject of analysis in this paper were obtained using the photoplethysmography (PPG) technique. PPG is a simple technique used to detect and monitor changes in blood flow in peripheral blood vessels. The measurement is performed on the surface of the skin. The PPG probe consists of an infrared light source (e.g. a photodiode emitting light at a wavelength of 900 nm) and a photodetector (eg a phototransistor). Plethysmograph (eng. plethysmograph) is a word created by the combination of two Greek words - 'plethymos' which means increase, amount, filling and 'graph' which means writing, recording and is a measuring method or instrument for measuring fluctuations (ripples) in the volume of a part of the body or organs.

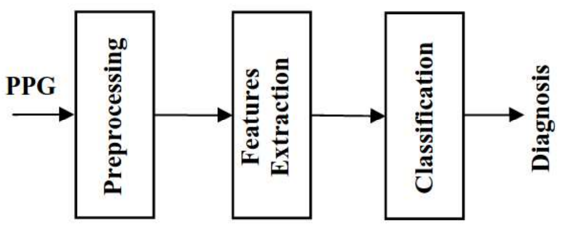

Figure 21: Processing of PPG signals

Photoelectric plethysmography is also known as photo-plethysmography and PTG/PPG is its acronym in some literature. In this work, the abbreviation PPG is used.

Pulse photoplethysmography includes the method of measuring plethysmography. A photoelectric transducer placed on the skin measures the volume pulse in the peripheral parts of the blood vessels. Corresponding volume pulsations modulate the exposed light beam of incoming infrared light, resulting in changes in intensity. They are measured with a phototransistor/photocell/photoresistor. Further processing of intensity changes in the gain stage leads to a visible image of the pulse curve.

The method of photoplethysmography is based on the procedure of measuring the pulse wave in the capillary area. The measurement parameters are the height of the amplitude and the analysis of the pulse waveform, which depend on the increase of the delay time - the beginning of the QRS complex to the beginning of the pulse, the peak time - the beginning to the end of the pulse and the slope time - the time of the rise of the pulse should be determined taking into account the steepest slope.

Dicrotism in the descending part of the curve is also important. [1] [2] [3] [16]

PPG is an easy to set up, efficient device compared to other types of plethysmography [4]. It does not require direct contact with the surface of the skin, as with some other plethysmograph methods.

Hertzman was the first to observe the relationship between the intensity of backscattered light and the volume of a fluid in 1938. [5] This useful simple method based on optical technology provides a significant contribution in health care (eg in primary care where precise and simple diagnostic techniques are desired).

Further development of PPG places this methodology among the more important tools used in the diagnosis of vascular diseases.

The contour of the PPG signal curve (Figure 2) is simple and can be divided into two phases:

- anakrotic phase - at the rising edge of the pulse and is associated with systole
- catacrotic phase - at the falling edge of the pulse and is associated with diastole.

Dicrotic notch appears in the catacrotic phase and is present in subjects with preserved - healthy arteries.

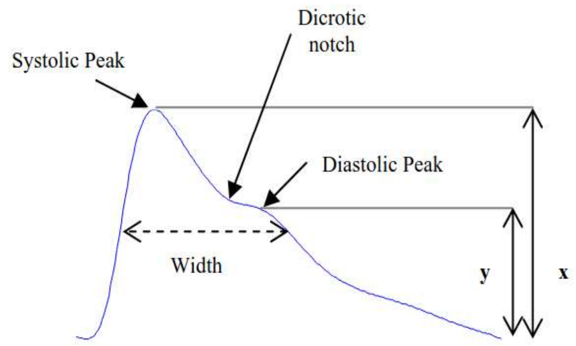

Figure 22: Characteristic parameters of the PPG signal - systolic, diastolic peak and pulse width

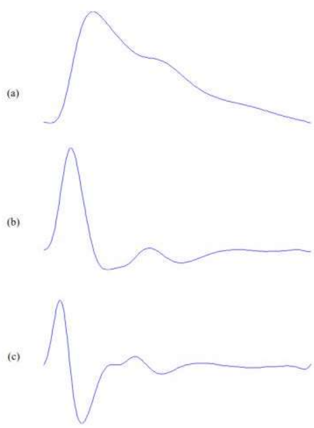

Figure 23: (a) PPG signal, (b) first derivative of PPG signal, (c) second derivative of the PPG signal

The analysis of PPG waveform itself attracts a lot of attention, especially from the aspect of monitoring circulation [6] and respiratory characteristics. [7] The PPG signal curve (Figure 2) can be defined and described with several parameters (characteristic parameters):

1. **Systolic amplitude** - x is an indicator of the change in the amount of blood in the arteries and blood vessels near the place where the measurement is performed. It has been shown that it is suitable for continuous determination of blood pressure. [8]
2. **Pulse width** - it is determined at half the height of the systolic amplitude (peak). It has been shown that it is correlated with systemic vascular resistance, ie. the resistance that must be overcome in order for the blood to move through the bloodstream. [9]
3. **Pulse area** - is defined as the total area under the PPG pulse (Figure 4). In some studies, this area is divided into two parts in relation to a discrete section [10], whereby the ratio of these two surfaces is used as an indicator of peripheral resistance, i.e. resistance of arteries to blood flow. This relationship is denoted by IPA - Inflection Point Area, IPA = A2/A1, (In the literature, Wang states, [10]).

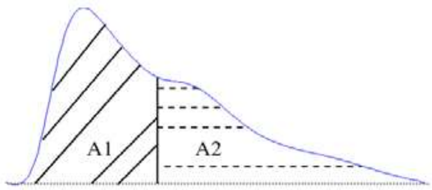

Figure 24: Characteristic parameters of the PPG signal - pulse area

4. **The interval between two peaks** - The time interval between two adjacent P peaks. The R-R interval in the ECG signal is correlated with the P-P interval in the PPG [11] (Figure 5).

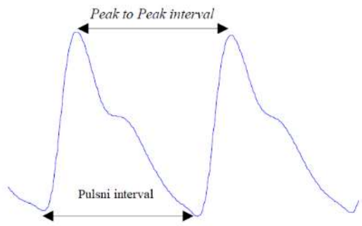

Figure 25: The interval between two adjacent pulses

5. **Pulse interval** - The difference between the beginning and end of the PPG pulse (Figure 5). It is usually used in the analysis when the diastolic segment is more clearly defined and easier to observe compared to the systolic part. [12]
6. **Augmentation index** - Pressure increase AI is a measure of the impact of the return wave on the systolic arterial pressure, it is obtained by measuring the reflected wave that goes from the periphery to the center. Pressure increase is obtained using the formula AI = y/x, where y is the height of the diastolic and x is the height of the systolic peak during one pulse (Figure 2) [13]. An alternative formula is also sometimes used in the literature AI = (x-y)/x (Rubins mentions in the literature, [14]).
7. **Index of stiffness of large arteries** - The time delay between the systolic and diastolic peaks (Figure 6) is related to the transit time of the pressure wave from the origin of the subclavian artery to the visible point of rejection and back. The length of this path is proportional to the height of the subject - h, this leads to the formulation of the contour index of the PPG signal SI, so it can be defined as SI = h/ΔT (Millasseau mentions in the literature, [15]).

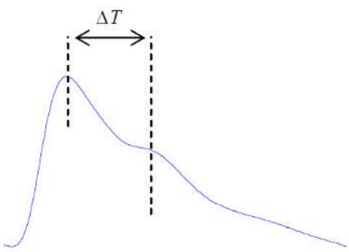

Figure 26: Time delay between systolic and diastolic peak ΔT

Although the curve of the PPG signal is simple, the analysis of the curve is quite complicated because it is not easy to detect changes in the region of the inflection point. For this reason, Takazawa [13] introduced the first and second derivatives of the PPG signal to facilitate the interpretation of the original signal and to facilitate parameter estimation.

In this paper, the topic of discussion is pre-processing, extraction features and parameter estimation using the wavelet transform - WT.

Wavelet transform is often used for representing complex signals and extracting their important features. This transformation is an important link in the development of various techniques that can be significant for specific applications, such as clinical practice in medicine.

In this paper, it is assumed that the PPG signal can be efficiently analyzed using the discrete wavelet transform.

A connection between the PPG signal and the application of discrete wavelet transformation is assumed in order to better visualize its content and estimate the parameters important for diagnosis.

The aim of this work belongs to the domain:

• Development, testing and addition of software in Python for PPG signal analysis.
• Application of wavelet transformation to PPG signal, analysis and application of results.
• Analysis of the results obtained by applying the theory of artificial neural networks to: PPG signal, to coefficients obtained by wavelet transformation.
• Generation of ANN models based on PPG wavelet transform coefficients.
• ANN model and prediction testing.
• Development of new methods for evaluating results and applications.

2 APPLICATION OF WAVELET TRANSFORM IN ANALYSIS OF PPG SIGNAL

2.1 Introduction to wavelet transform

Using the Fourier transform, the signal can be decomposed into spectral components. The problem with the Fourier transformation is that the time resolution is completely lost, which is not favorable for the analysis of non-stationary signals. For this reason, transformation methods were developed that would provide data on both the time and the frequency of an individual component.
Two such methods are:

- Short Time Fourier Transform - STFT
- Wavelet transform - WT

The goal of such analyzes is to enable the simultaneous monitoring of both frequency and time signals by displaying the signal energy distribution in the time-frequency plane. The wavelet transform has emerged as an alternative to the long-used STFT. The difference is that STFT offers analysis with an absolute constant range, while WT presents an analysis with a constant relative range, i.e. a constant Q factor.

The wavelet transform can be seen as a decomposition of the signal into a series of basic functions (wavelets) obtained from one prototype. By correcting (shortening) or dilating (stretching) and shifting in time the basic wavelet, a series of analyzer functions is obtained.

The wavelet prototype can also be interpreted as a bandpass filter. As a consequence of the constant Q factor, the other filters will also be a scaled version of the prototype in that case.

The wavelet method can be applied to both continuous and discrete signals. In STFT, the non-stationary signal is divided into segments with assumed stationarity and the Fourier transformation is applied to the segments thus separated.

The signal $s(t)$ is divided by a function $g(t-\tau)$ into segments, where τ represents the time location of the window.

STFT is actually a Fourier transform of a windowed signal $x(t)g(t-\tau)$ or inner product between the signal and the window:

$$STFT(\tau,f) = \int x(t) g\ \ (t-\tau) e^{-2j\pi f t}\, dt \qquad (1)$$

this equation represents the signal as a two-dimensional function in the time-frequency plane *(τ, f)*.
The wavelet transform is defined from the base wavelet ψ, which represents the analyzer function located at the same time in the time and frequency plane. From this basic wavelet, by scaling (parameter *α*) and translation in time (parameter *τ*), a series of analyzer functions is obtained.
The inner product of the signal and the wavelet sequence gives the Wavelet transform:

$$WT(\tau,\alpha) = \int x\ \ (t)\ \ \ _{\tau,\alpha}(t)dt \qquad (2)$$

gde je $\ _{\tau,\alpha}(t) = \alpha^{-1/2}\ \ \left(\frac{t-\tau}{\alpha}\right)$
(3)

Wavelet extracts time-frequency (via scaling parameters) information from the signal via the inner product between the signal and the scaled version of the wavelet.

Wavelet analysis results in a series of wavelet coefficients that show how close the signal is to a single basis function. In this way, it can be expected that any signal can be decomposed into a series of wavelets, that is, that the original form of the signal can be obtained by synthesis, by adding elementary blocks of constant shape but also of different sizes and amplitudes.
[18] [19] [20] [21]

2.2 Applying the wavelet transform

In this paper, an integrated function is presented for signal decomposition in the Python programming environment. The integrated function used for PPG signal decomposition is a function from the modwtpy library: coef = modwt(x, waveletname, nLevel). modwt (maximal overlap discrete wavelet transform) is a function of the discrete wavelet transform with the maximum number of overlaps, i.e. a series of wavelet coefficients (nLevel can be pre-defined - the signal decomposition level). The parameter x is the input signal to be decomposed, waveletname represents the base wavelet used in the decomposition. The output quantity is a matrix of coefficients where the rows of the matrix represent a series of coefficients of functions that were created by decomposition.

The analyzed signal is a PPG signal that was created using photoplethysmography (Figure 7). [28]

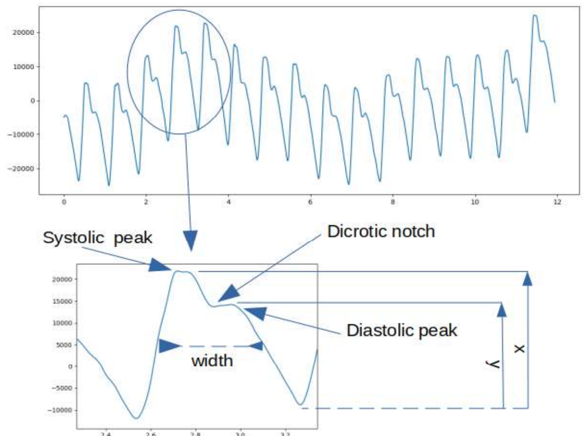

Figure 27: Original signal and characteristic parameters

By applying the Fourier transformation to the PPG signal (Figure 7), four dominant frequency peaks can be discerned (Figure 8).

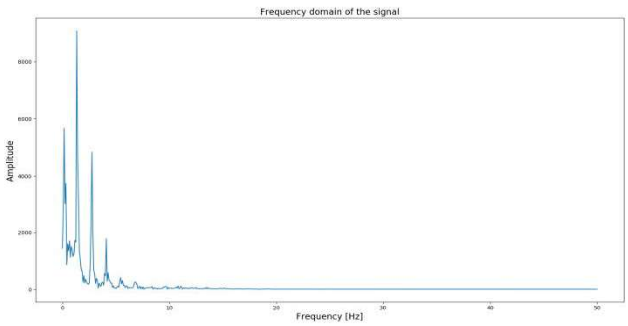

Figure 28: Frequency domain of the original signal

WT and signal decomposition using modwt function is shown (Figure 10). In this example, 7 levels of decomposition were performed using the rbio6.8 basic wavelet (Figure 9).

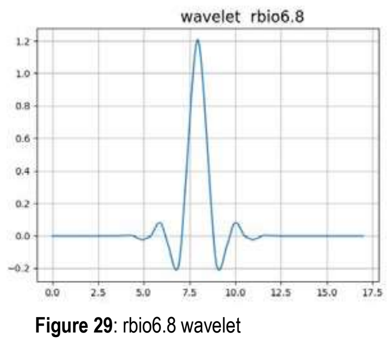

Figure 29: rbio6.8 wavelet

By changing wavelet transform coefficients {coef 0, coef 1, coef 2, coef 3, coef 4, coef 5, coef 6} the reconstructed signal will differ from the original.

Preprocessing in the WT domain can be done in three steps:

1. signal decomposition (Figure 10) ,
2. modification of the obtained coefficients (Figure 11),
3. signal reconstruction based on modified coefficients (Figure 12).

If the characteristic signals created by decomposition (Figure 10) are compared with the frequency domain (Figure 8), it can be concluded that the decomposed signals can be divided into those with high frequencies and those with low frequencies.

Modification and shortening of coefficients:
{coef 0, coef 1, coef 2, coef 3, coef 4, coef 5, coef 6} ->
-> {mod coef 0=0, mod coef 1=0, mod coef 2=0, mod coef 3=coef 3, mod coef 4=coef 4, mod coef 5=coef 5, mod coef 6=0}

Signals with high or very low frequencies can be removed as irrelevant components for PPG analysis (Figure 11).

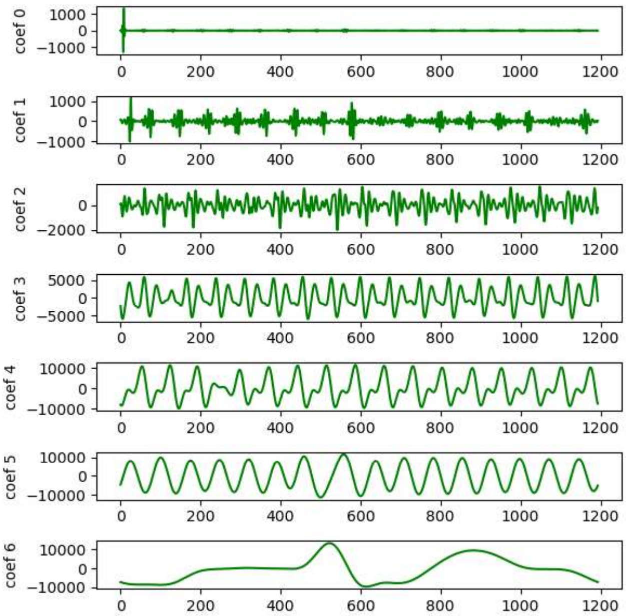

Figure 30: WT transformation coefficients

Elimination of HF noise, i.e. modification of coefficients with high or low frequency can be done using thresholding. Noise removal by applying a threshold should be carried out by making a compromise between the presence of noise and changes in important signal characteristics.

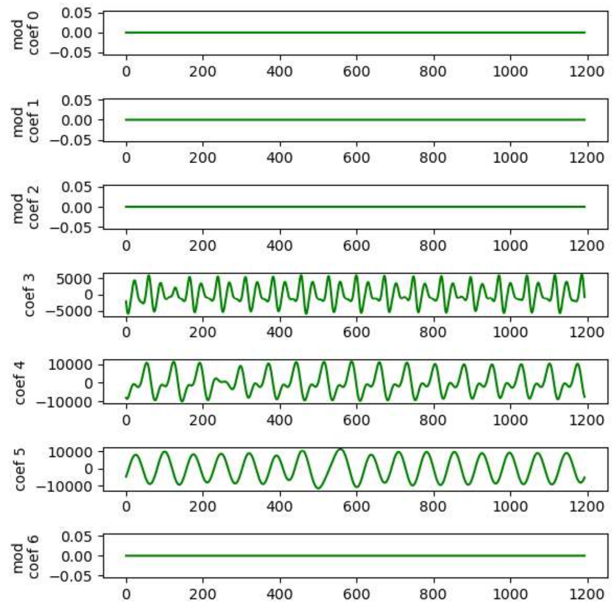

Figure 31: Modified WT transformation coefficients

The frequency domain of the reconstructed signal contains 3 peaks (Figure 13) of corresponding signal frequencies with modified coefficients (Figure 12).

It can be assumed that these 3 signals (Figures 14, 15) are directly related to PPG characteristic parameters such as (Figure 16):

- Systolic amplitude,
- pulse width,
- pulse area,
- interval between two peaks,
- pulse interval,
- index of increase,
- stiffness index of larger arteries, etc...

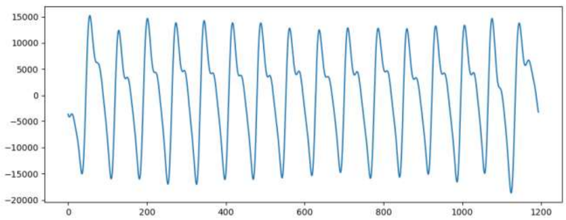

Figure 32: Reconstructed signal

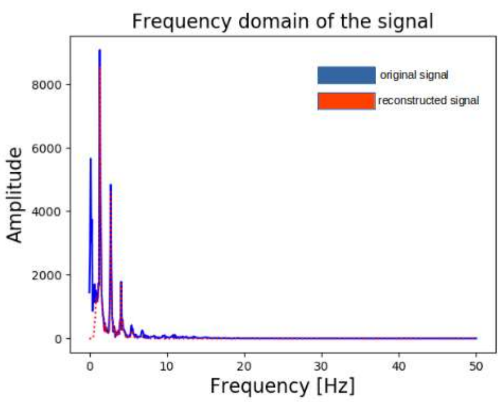

Figure 33: Frequency domain of the original and reconstructed signal

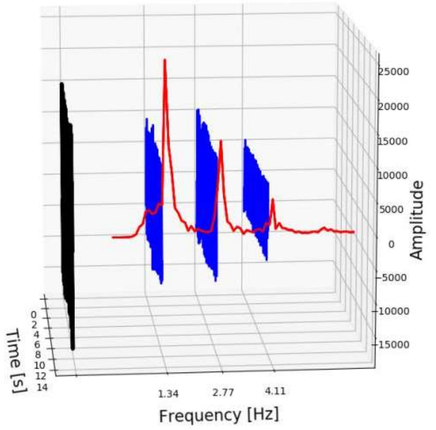

Figure 34: A reconstructed signal consisting of 3 component signals of different frequencies

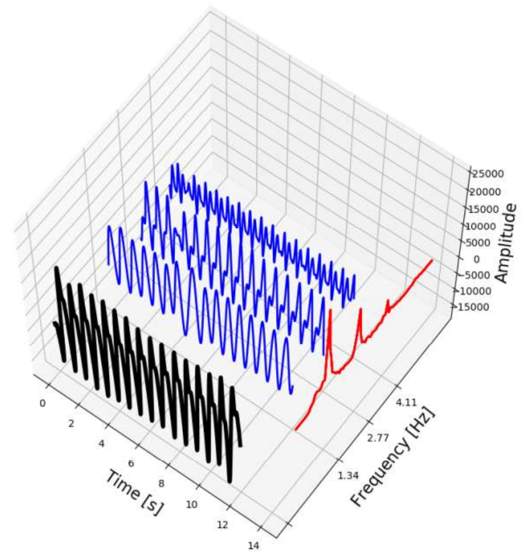

Figure 35: A reconstructed signal consisting of 3 component signals of different frequencies

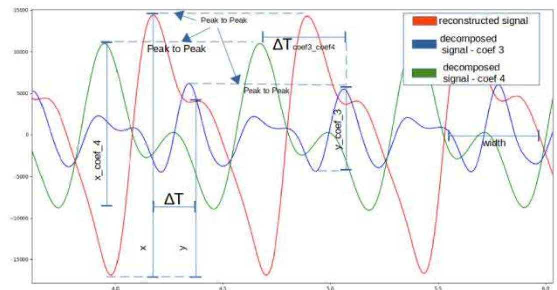

Figure 36: Comparing the basic signal and the signal obtained by decomposition

2.2.1 Conclusion of application of wavelet transformation

Based on the exposed properties of the wavelet transform, it can be concluded that there is a great potential application of this method in the analysis of PPG signals.

It can be assumed that the signal coefficients coef 3, coef 4 and coef 5 are relevant for PPG analysis and estimation of characteristic parameters. By accessing databases and

measured PPG signals, it can be determined whether there is a direct connection between e.g.
(Figure 16):

- x_coef_4 and x (systolic amplitude),
- y_coef_3 and y (diastolic amplitudes),
- ∆T_coef_3_coef_4 and ∆T (the ratio of the ∆T transformation coefficients and original signal),
- Peak to Peak analysis of the original signal and signals obtained by wavelet decomposition, etc...

Wavelet transformation can determine in a compromise way when certain changes occur in the signal and what their frequency content is, which represents the basic parameters in any signal analysis. This analysis proved to be more efficient than the conventional method for time-frequency analysis with a constant range. WT favors time resolution when analyzing high frequencies, that is, frequency resolution at low frequencies, in contrast to STFT, which has a constant resolution in the time and frequency domains. WT provides percentage constant range analysis which makes this transformation very suitable for PPG signal analysis.

This type of analysis is most suitable for signals consisting of high-frequency components of short duration and low-frequency components of long duration, which makes this transformation very effective for the practical application of PPG signal analysis in practice. PPG signals are characterized by a certain harmonic content followed by relatively fast transients where WT is very suitable for analysis.

The assumed characteristic parameters of PPG signals obtained by WT should be compared with other methods by accessing databases and measurements of biomedical signals and their direct connection should be confirmed, which would result in the development of this method for analysis, processing and creation of new algorithms based on WT for the most accurate and successful extraction of parameters for reliable analysis.

3 WAVELET SELECTION FOR PPG SIGNAL ANALYSIS - TEST AND RESULTS

Based on the conclusions and assumptions from the previous chapter, that the wavelet transformation is suitable for detailed analysis of PPG signals, in this chapter the focus will be on choosing an adequate wavelet from the wide range of built-in wavelet functions available in the Python program. The selected wavelet or several wavelets, which prove to be adequate for the estimation of characteristic parameters, will be used in the transformation and tested on PPG signals.

In order to support these assumptions with a practical example, this paper will attempt to extract the AI parameter (Augmentation index). The AI parameter in combination with other characteristic parameters can be an important factor in the method for assessing the subject's condition based on the characteristics of the PPG signal. The extraction of other characteristic parameters with the help of wavelet transformation is a topic for further work in this area. [13] [14]
In the method used in this example, it is assumed that:

1. The characteristic parameters of the PPG signal can be obtained by wavelet transformation. If a wavelet transformation is performed on the signal, then the transformation coefficients are directly related to the characteristic parameters of the PPG signal (assumption).
2. In the example, it was assumed that: if a wavelet transformation of level 7 is performed on the signal, then the transformation coefficients 2 and 3 are directly proportional to the characteristic parameters of the PPG signal such as: systolic and diastolic 'peak' and that in this way these parameters can be extracted from the signal.
   A more detailed analysis of the stated assumption under point 2 is a topic for further work in this area.
3. Assumption: if the PPG signal is "typical" then it is AI = y/x (AI - *Augmentation index*), where y is the diastolic 'peak' and x is the systolic 'peak', then it is 0<AI<1.

Below is an example of a method for selecting and testing wavelets while confirming some of the above assumptions. The display of wavelet testing for PPG signal analysis will be performed on the example of generating a neural network model for predicting the age of the subject based on PPG signals. In this example, a wavelet transformation (level 7) was performed on the PPG signals.

When generating the inverse wavelet transform, coefficients 2, 3 will be used. The entire procedure, as well as the Python programming tools that were used, will be described in the following sections.

The program code used in this example is thematically divided into 5 Python scripts:

1. "moderated_excel" - code for generating a file with PPG signals suitable for further processing;
2. Wavelet selection - code to test about 23 embedded wavelets in Python on 11 selected "typical" PPG signals;
3. Wavelet transformation - code for performing wavelet transformation on a file from "sorted excel" using the selected wavelet, generates a file with transformed PPG signals;
4. generation of a neural network model for prediction where a file with transformed PPG signals is used to train the network;
5. testing the prediction network on files with PPG signals;

Python scripts are described in the following chapters along with comments and analysis of the tests and results.

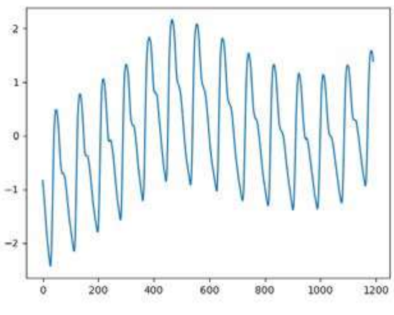

Figure 37: PPG signal

3.1 Preparing the file using a Python script

Data on PPG signals - measured PPG signals [28] - are prepared in a suitable structure suitable for further processing. The most convenient data structure for displaying PPG signals and applying a wavelet transform to them is a sequence of numbers. Below is a presentation of the DB preparation code that generates a *.csv file with PPG signals ready for further processing:

```
import pandas as pd
import numpy as np
```

```
import random as rand
import matplotlib.pyplot as plt
import pywt

# read the excel file
df = pd.read_excel('../db1.xlsx');
# extract ID and age columns
id_age_df = df[['data_id', 'age']];
# extract PPG data columns
ppg_data_df = df.iloc[:, 2:] # Select columns from C onwards
# Create a new DataFrame with ID, age, and PPG data
output_df = pd.DataFrame(columns=['id', 'age', 'data']);
for idx, row in id_age_df.iterrows():
   data_id = row['data_id'];
   age = row['age'];
   ppg_data = ppg_data_df.iloc[idx].dropna();
   ppg_data = pd.to_numeric(ppg_data, errors = 'coerce');
   ppg_data = ppg_data[~np.isnan(ppg_data)].tolist();
   output_df = pd.concat( [output_df, pd.DataFrame({'id': data_id, 'age': age, 'data':
        [ppg_data]})], ignore_index=True);
# save the DataFrame to a CSV file
output_df.to_csv('moderated_excel.csv', index=False);
print('Data saved to moderated_excel.csv');
```

The result of executing this script is the PPG signals placed in a file "moderated_excel.csv".

	A	B	C D E F G H I J
1	id	age	data
2	2013338	0.3	[-2075, -1009, 28, 1037, 1990, 2858, 3603, 4160, 4528, 4707, 4745, 4650, 4471, 4245, 400
3	2013323	0.34	[-11627, -11584, -11670, -12006, -12696, -13626, -14522, -15108, -15323, -15384, -15539,
4	2018532	0.35	[18174, 17403, 16653, 15922, 15212, 14543, 13914, 13336, 12809, 12312, 11835, 11369,
5	2018531	0.57	[-13403, -11313, -9301, -7451, -5849, -4402, -2937, -1284, 539, 2235, 3502, 4119, 4188, 38
6	2018521	0.49	[-18072, -19190, -20238, -21126, -21795, -22065, -21566, -19909, -16744, -12222, -7072, -
7	2018515	0.65	[-3533, -4546, -5471, -6206, -6683, -6912, -6961, -6902, -6792, -6653, -6514, -6395, -6335,
8	2018524	0.58	[-16051, -16536, -17030, -17525, -18051, -18587, -19144, -19721, -20298, -20835, -21288,
9	2018532	0.35	[18174, 17403, 16653, 15922, 15212, 14543, 13914, 13336, 12809, 12312, 11835, 11369,
10	2018514	0.65	[-396, -1145, -1885, -2625, -3365, -4068, -4727, -5304, -5801, -6252, -6685, -7145, -7650, -
11	2018522	0.49	[-8398, -8751, -9104, -9466, -9838, -10209, -10609, -11027, -11473, -11956, -12476, -1304
12	2018511	0.46	[16104, 15677, 15260, 14865, 14502, 14181, 13925, 13754, 13648, 13583, 13530, 13466,
13	2018490	0.31	[-25000, -23521, -21599, -18827, -14824, -9936, -4976, -717, 2391, 4427, 5499, 5729, 531
14	2018489	0.59	[-15756, -16114, -16442, -16681, -16810, -16870, -16950, -17129, -17437, -17786, -18064,
15	2018488	0.49	[9998, 9615, 9161, 8555, 7736, 6659, 5386, 3988, 2510, 1041, -356, -1584, -2590, -3374, -
16	2018487	0.4	[-11070, -9440, -7910, -6580, -5520, -4760, -4250, -3980, -3900, -3960, -4110, -4310, -4520
17	2018485	0.57	[-8132, -8581, -9040, -9490, -9949, -10417, -10886, -11354, -11833, -12320, -12817, -1332
18	2018482	0.54	[7120, 6449, 5789, 5166, 4601, 4103, 3643, 3203, 2791, 2379, 1987, 1584, 1192, 828, 483
19	2018478	0.3	[3244, 3092, 2940, 2747, 2523, 2300, 2117, 2026, 2016, 1955, 1731, 1264, 655, 66, -370, -

Figure 38: PPG signals written in "moderated_excel.csv"

3.2 Wavelet transformation of the signal from DB

3.2.1 Analysis of signals from DB and selection of wavelets

It was assumed that if a wavelet transformation is performed on the 'typical' PPG signal (Figure 17), then the obtained transformation coefficients are directly proportional to the systolic and diastolic 'peak'. In order to confirm and test this assumption on a practical example, a database with 500 PPG signals will be used [28]. From this database by cursory analysis and random drawing (plt.plot fnc in Python) of signals 11 signals were selected which are approximate, i.e. most resemble a 'typical' PPG signal.

Selected signals (s_0....s_10, in total 11) where their serial numbers from the specified DB are written in a series of serial numbers named "test_samples":

test_samples = [480, 280, 264, 301, 55, 401, 198, 106, 110, 230, 101];

Figure 19 shows the signals s_0 do s_10 where s_0 is a signal 480 ..etc.. s_5 is a signal 401..etc..s_10 is a signal 101.

Wavelet transforms will be performed on each of these 11 PPG signals, using built-in wavelets available in the Python environment. The list of wavelets that will be used in testing (Figures 20, 21, 22) is defined as an array named "wavelet" with a total of 23 wavelets:

wavelet =['db1', 'db3', 'db5', 'db7', 'sym2', 'sym3', 'sym5', 'sym7', 'coif1', 'coif3', 'coif5', 'coif7', 'rbio1.1', 'rbio1.5', 'rbio2.6', 'rbio3.3', 'rbio3.5', 'rbio3.7', 'rbio3.9', 'rbio4.4', 'rbio5.5', 'rbio6.8', 'dmey'];

Each of the selected 11 PPG signals will be transformed with each of these 23 wavelets and then inverse transformed so that there will be 23 'versions' of the transform for each signal.

To obtain the AI parameter (AI = y/x, ratio of diastolic to systolic 'peak') will be used the ratio of the mean value of the signal amplitude obtained by inverse transformation of the coefficients "2" for the systolic 'peak' and the coefficients "3" for the diastolic 'peak'.

In this way, the AI calculation procedure will be repeated for each wavelet.

The AI parameter is relevant to the selection of the wavelet by assuming that if the wavelet is optimal then the ratio of the transformation coefficients obtained by this wavelet is: AI = y/x -> 1.

If we assume that a 'typical' PPG signal has a pronounced diastolic 'peak', then the average amplitude of coefficients "3" divided by the average amplitude of coefficients "2" should limit to 1 (assumption in chapter 2.2.1).

Based on this criterion, in this work only based on graphical analysis, the wavelet would be selected (Figure 23, 24) which will be used during further analysis and testing and 'machine learning' analysis.

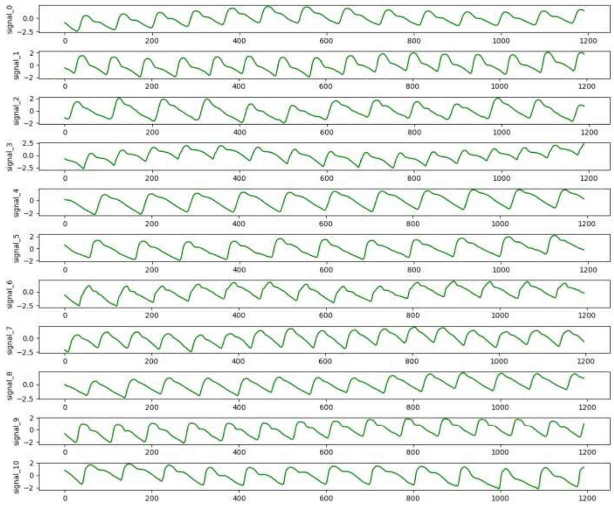

Figure 39: PPG Signals used for testing and selecting wavelets

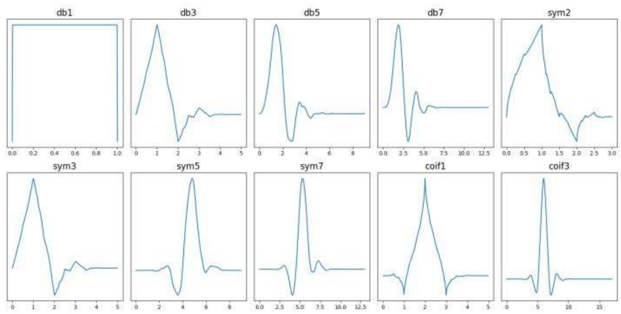

Figure 40: Tested wavelets

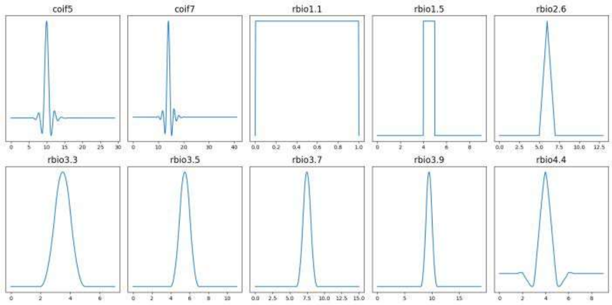

Figure 41: Tested wavelets

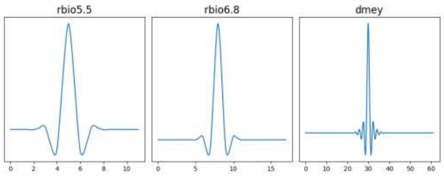

Figure 42: Tested wavelets

A preview of the Python script used to analyze the wavelet below:

```
import matplotlib.pyplot as plt
import numpy as np
import pywt
import pywt.data
import pandas as pd

mode = pywt.Modes.smooth
input_csv = "../../moderated_excel_db1_500.csv"
# Normalizing

def normalize_ppg(ppg_signal):
   # Normalize all values to be between -1 and 1
   signal = ppg_signal;
   return (signal - np.mean(signal)) / np.std(signal)
# get ppg data from csv file
def get_ppg_data(input_csv, test_samples):
   input_df = pd.read_csv(input_csv, sep = ',');
   data_id = (input_df["id"][test_samples]);
   data_age = (input_df["age"][test_samples]);
   ppg_data = eval(input_df["data"][test_samples]);
   return data_id, data_age, ppg_data
def signal_decomp(data, w, title):
"""Decompose and plot a signal S.
   S = An + Dn + Dn-1 + ... + D1
"""
   w = pywt.Wavelet(w)
   a = data
   ca = []
   cd = []
   for i in range(7):
   (a, d) = pywt.dwt(a, w, mode)
      ca.append(a)
      cd.append(d)
   rec_a = []
   rec_d = []
   for i, coeff in enumerate(ca):
      coeff_list = [coeff, None] + [None] * i
      rec_a.append(pywt.waverec(coeff_list, w))
   coeffs = [];

   coeffs = cd.copy();
   coeffs.append(ca[6]);
   coeffs.reverse();
   return coeffs, rec_a, rec_d
def plot_signal_decomp(data, w, title):
"""Decompose and plot a signal S.
   S = An + Dn + Dn-1 + ... + D1
"""
   w = pywt.Wavelet(w)
   a = data
   ca = []
   cd = []
   for i in range(7):
         (a, d) = pywt.dwt(a, w, mode)

      ca.append(a)
      cd.append(d)
   rec_a = []
   rec_d = []
   for i, coeff in enumerate(ca):
      coeff_list = [coeff, None] + [None] * i
      rec_a.append(pywt.waverec(coeff_list, w))
         for i, coeff in enumerate(cd):

      coeff_list = [None, coeff] + [None] * i
      rec_d.append(pywt.waverec(coeff_list, w))
         coeffs = [];

   coeffs = cd.copy();
   coeffs.append(ca[6]);
   coeffs.reverse();
   return coeffs
def signal_waverec(coeffs, wavelet, coeffs_to_rec):
"""waverec signal on coefitient base.
    coefitient choose with coeffs_to_rec
   S = An + Dn + Dn-1 + ... + D1
"""
   wavelet = pywt.Wavelet(wavelet)
   coeffs  = coeffs.copy();
   v = coeffs_to_rec
   if (v[0]<1): coeffs[0] = np.zeros(len(coeffs[0]));
   if (v[1]<1): coeffs[1] = np.zeros(len(coeffs[1]));
   if (v[2]<1): coeffs[2] = np.zeros(len(coeffs[2]));
   if (v[3]<1): coeffs[3] = np.zeros(len(coeffs[3]));
   if (v[4]<1): coeffs[4] = np.zeros(len(coeffs[4]));
   if (v[5]<1): coeffs[5] = np.zeros(len(coeffs[5]));
   if (v[6]<1): coeffs[6] = np.zeros(len(coeffs[6]));
   if (v[7]<1): coeffs[7] = np.zeros(len(coeffs[7]));
   signal_rec = pywt.waverec(coeffs, wavelet);
   return signal_rec;
ppg_data = [];
test_samples = [480, 280, 264, 301, 55, 401, 198, 106, 110, 230,        101];
for index in range(len(test_samples)):
    ppg_data.append(get_ppg_data(input_csv, test_samples[index]));
wavelet =['db1', 'db3', 'db5', 'db7', 'sym2', 'sym3',
         'sym5', 'sym7', 'coif1', 'coif3', 'coif5', 'coif7',  'rbio1.1', 'rbio1.5', 'rbio2.6', 'rbio3.3', 'rbio3.5',
'rbio3.7', 'rbio3.9', 'rbio4.4', 'rbio5.5', 'rbio6.8', 'dmey'];
plot_title = [  'db1', 'db3', 'db5', 'db7', 'sym1', 'sym3',
         'sym5', 'sym7', 'coif1', 'coif3', 'coif5', 'coif7', 'rbio1.1', 'rbio1.5', 'rbio2.6', 'rbio3.3', 'rbio3.5',
'rbio3.7', 'rbio3.9', 'rbio4.4', 'rbio5.5', 'rbio6.8', 'dmey'];
#ppg signal information is "stored" in signal_wt_data
signal_wt_data = [];
x_axis = []; #x - wavelet
y_axis = []; #y - ratio AI
for i in range(len(ppg_data)):
   wt_data_1 = [];
   coeffs = [];
   plot_AI_data_x = [];
   plot_AI_data_y = [];
   for index in range(len(wavelet)):
      signal = [];
      ppg_data_n = normalize_ppg(ppg_data[i][2]);
      coeffs.append(signal_decomp(ppg_data_n, wavelet[index],        plot_title[index]))
      #coeffs.append(signal_decomp(ppg_data[i][2],
            wavelet[index], plot_title[index]))
      signal_0 =  signal_waverec(coeffs[index][0],               wavelet[index],
[0,0,1,1,1,1,1,1]);
      signal_1 =  signal_waverec(coeffs[index][0],
            wavelet[index], [0,0,1,0,0,0,0,0]);
      s_1_mean = np.mean(abs(signal_1));
      signal_2 =  signal_waverec(coeffs[index][0],
            wavelet[index], [0,0,0,1,0,0,0,0]);
      s_2_mean = np.mean(abs(signal_2));
      signal_3 =  signal_waverec(coeffs[index][0],
            wavelet[index], [0,0,1,1,0,0,0,0]);
      signal_rec = [signal_0, signal_1, signal_2, signal_3];
      AI = s_2_mean/s_1_mean;
      signal_rec = [signal_rec, wavelet[index], AI]
      signal.append(signal_rec);
      plot_AI_data_x.append(wavelet[index]);
      plot_AI_data_y.append(AI);
      plot_AI_data = [plot_AI_data_x, plot_AI_data_y]
      #wt_data = [coeffs, signal, plot_AI_data];
      wt_data = [coeffs, signal];
      wt_data_1.append(wt_data);
   #signal_wt_data.append(wt_data_1);
   wt_data_2 = [wt_data_1, plot_AI_data]
   signal_wt_data.append(wt_data_2);
level = 7;
```

```
fig, axarr = plt.subplots((level+1), ncols=1, figsize=(12,12));
for i in range (level+1):
    axarr[i].plot(coeffs[1][0][i] , 'g');
    axarr[i].set_ylabel("coeffs {}".format(i), fontsize=10,      rotation=90);
    plt.tight_layout();
    plt.show();
coeffs_1, rec_a, rec_d = signal_decomp(ppg_data[0][2], 'sym5', 'sym5');
plt.figure()
for i in range(len(signal_wt_data)):
    x_axis = signal_wt_data[i][1][0];
    y_axis = signal_wt_data[i][1][1];
    plt.plot(x_axis, y_axis, label="s_%d" % (i));
plt.legend(loc='upper right')
plt.show()
fig, axarr = plt.subplots((len(ppg_data)), ncols=1, figsize=(12,12));
for iii in range (len(ppg_data)):
    signal_to_plot = normalize_ppg(ppg_data[iii][2])
    axarr[iii].plot(signal_to_plot , 'g');
    axarr[iii].set_ylabel("signal_{}".format(iii), fontsize=10, rotation=90);
    plt.tight_layout();
    plt.show();
```

Executing the above code generates a graph showing the ratio of amplitudes AI = y/x for different types of wavelets. By simple analysis of the graph, it can be concluded that the ratio is amplitude 0<AI<1.
Values AI->1 (values that limit according to 1) are grouped around wavelets rbior3.3 (Figure 21, 22). It is assumed that this wavelet is optimal for signal transformation, extracting characteristic parameters and creating an ANN model on the basis of which the prediction will be made.

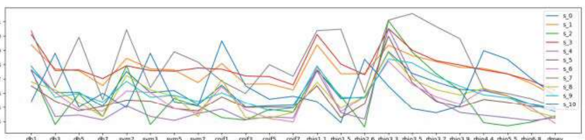

Figure 43: Display of ratio of amplitudes of coefficients [2] and [3] for different types of wavelets

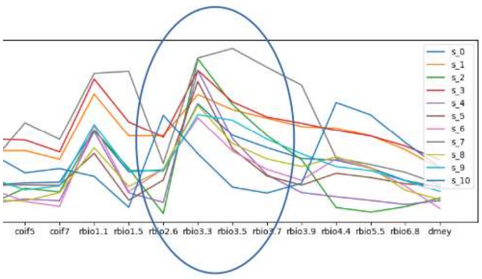

Figure 44: Display of ratio of amplitudes of coefficients [2] and [3] for different types of wavelets - more in detailmore in detail (rbio3.3)

3.2.2 Wavelet transformation of the signal from DB using the selected wavelet

The following example will show the generation of a file containing signals that have been wavelet transformed with the selected wavelet. The transformation will be performed as follows: PPG signal is transformed by wavelet transformation, transformation coefficients "0, 1, 4, 5, 6, 7" are modified ie. overwritten with 0 (Figure 25), the inverse transformation of the modified signals is performed, so that the reconstructed signal is used for training the network. In the example, two PPG signal files that were created by generating through the application listed in references [28], will be used:

1. File of 500 PPG signals

filename = 'moderated_excel_db1_500.csv'

2. File of 3000 PPG signals

filename = 'ppg_3000.csv'

```
import csv
import pywt
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# Normalizing
def normalize_ppg(ppg_signal):
    # Normalize all values to be between -1 and 1
 return 2.*(ppg_signal - np.min(ppg_signal))/np.max(ppg_signal)-1
# define the wavelet function and level of decomposition
wavelet = 'rbio3.3'
level = 7
filename   = 'moderated_excel_db1_500.csv'; # 1 DB
outputfile = 'wavelet_transformed_rbio3.3_level_7_db1_500.csv'; # transforma DB 1
#filename   = 'ppg_3000.csv'; # 2 DB
#outputfile = #'wavelet_transformed_rbio3.3_level_7_ppg_3000.csv'; # transform #DB 2
# read the input csv file
with open(filename, 'r') as file:
  reader = csv.reader(file)
  next (reader) # skip the header row if present
# create an output csv file for storing the wavelet coeffs
  with open(outputfile, 'w', newline ='') as output_file:
    writer = csv.writer(output_file);
    writer.writerow(['data_id', 'age', 'data']);
                  # write header row
    for row in reader:
      data_id = row[0];
      age = row[1];
      ppg_data = eval(row[2]); # convert the string to a list
      # perform wavelet transformation
      coeffs = pywt.wavedec(ppg_data, wavelet, level=level);
           coeffs[0] = np.zeros(len(coeffs[0]));
      coeffs[1] = np.zeros(len(coeffs[1]));
      #coeffs[2] = np.zeros(len(coeffs[2]));
      #coeffs[3] = np.zeros(len(coeffs[3]));
      coeffs[4] = np.zeros(len(coeffs[4]));
      coeffs[5] = np.zeros(len(coeffs[5]));
      coeffs[6] = np.zeros(len(coeffs[6]));
      coeffs[7] = np.zeros(len(coeffs[7]));
      # inverse discrete waveled transform
      ppg_rec = pywt.waverec(coeffs, wavelet)
           # normalize signal
      samples = normalize_ppg(ppg_rec);
      # convert the coefficients to a string representation
      coeffs_str = ','.join(str(c) for c in samples);
      # write the data to the output csv file
      writer.writerow([data_id, age, coeffs_str]);
print ('Inverse discrete wavelet transformation completed and              saved to', outputfile);
```

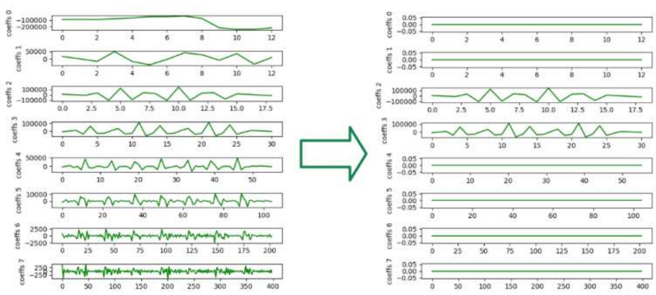

Figure 45: Modification of wavelet transform coefficients

3.3 Generating ANN models

A neural network is a machine learning model inspired by biological neural networks, consisting of sets of artificial neurons (often called layers) that process data and perform complex tasks such as classification, regression, shape recognition, prediction...etc.

In Python, neural networks can be implemented using various libraries such as TensorFlow, Keras, PyTorch, and others.[23] [24] [27]

The following code shows the generation of an ANN training model. [23] It could be said that training a prediction network and testing it on the same data is actually a methodological error: a model that just repeated the patterns it was just familiar with would have a perfect score, but it would fail to predict anything useful on the data it was unfamiliar with.

This situation is "jargonically" called "overfitting". To avoid this, it is common practice when performing (supervised) network training, and to keep part of the available data as a test data set.

The following code shows the training of the network, where the transformed PPG signal file [28] from the previous section is used for training:

```
import ast
import pandas as pd
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.utils import pad_sequences

from keras.callbacks import History
history = History()


# load the .csv file
csv_waverec_file = 'wavelet_transformed_rbior3.3_level_7_db1_500.csv';
df = pd.read_csv(csv_waverec_file);
# Convert 'Coefficients' from string to list using ast.literal_eval
df['data'] = df['data'].apply(lambda x: ast.literal_eval(x) );
sequences = df['data'].values;
# padding
X = pad_sequences(    sequences,
                maxlen=2000,
                padding='post',
            );

y = df['age'].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1,random_state=42);
# Model architecture remains the same
model = Sequential()
model.add(Dense(200, input_dim=len(X[0]), activation='tanh'))
model.add(Dense(30, activation='sigmoid'))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam', metrics=['accuracy'])
model.fit(X_train, y_train, epochs=300, batch_size=8)
loss = model.evaluate(X_test, y_test)
print('Test loss:', loss)
output_file = '../model/small_200_30_sigmoid_wt_rbior3.3_l_7_db_1_500_1_1.h5';
model.save(output_file);
print ('model saved to', output_file);
```

Script execution:

```
.
.
57/57 [==============================] - 0s 7ms/step - loss: 1.3101e-04
Epoch 284/300
57/57 [==============================] - 0s 7ms/step - loss: 6.8416e-05
Epoch 285/300
57/57 [==============================] - 0s 6ms/step - loss: 5.9931e-05
Epoch 286/300
57/57 [==============================] - 0s 6ms/step - loss: 7.9257e-05
Epoch 287/300
57/57 [==============================] - 0s 6ms/step - loss: 9.0686e-05
Epoch 288/300
57/57 [==============================] - 0s 6ms/step - loss: 1.4148e-04
Epoch 289/300
57/57 [==============================] - 0s 7ms/step - loss: 1.2856e-04
Epoch 290/300
57/57 [==============================] - 0s 7ms/step - loss: 1.0635e-04
Epoch 291/300
.
.
```

Figure 46: Generating ANN models

3.4 Testing ANN models

Python code for predicting the subject's age based on the PPG signal transformed by wavelet transformation.

Testing the generated model from the previous section will be performed twice on two PPG signal files [28], variable **csv_waverec_file** will be defined in two ways:

1. File of 500 PPG signals
csv_waverec_file = 'wavelet_transformed_rbior3.3_level_7_db1_500.csv'

2. File of 3000 PPG signals
csv_waverec_file = 'wavelet_transformed_rbior3.3_level_7_ppg_3000.csv'

```
import csv
import pandas as pd
import matplotlib.pyplot as plt
from tensorflow.keras.utils import pad_sequences
from tensorflow.keras.models import load_model
from sklearn.metrics import mean_absolute_error, mean_squared_error

# load the model
model = load_model('../model/small_200_30_sigmoid_wt_rbior3.3_l_7_db_1_500_1.h5');
# load the .csv file
csv_waverec_file = 'wavelet_transformed_rbior3.3_level_7_db1_500.csv'; # 1 DB
#csv_waverec_file = 'wavelet_transformed_rbior3.3_level_7_ppg_3000.csv'; # 2 DB
input_data = pd.read_csv(csv_waverec_file);
# extract data from the DataFrame
ppg_signals_wt = input_data['data'].apply(eval).tolist();
correct_ages = input_data['age'].values;
data_id = input_data['data_id'].values;
ppg_signals_wt = pad_sequences(ppg_signals_wt, maxlen = 2000, padding = 'post');
# predict the age for each PPG signal
ages_predicted = model.predict(ppg_signals_wt)*100;
outputfile = '../prediction/prediction_rbior3.3_db_1_500_rbior3.3_level_7_db_2_308_1.csv'
y_true = []; y_pred = [];
with open(outputfile, 'w', newline ='') as output_file:
    writer = csv.writer(output_file);
    writer.writerow(['data_id', 'actual_age', 'predicted_age']); # write header row
    for i in range(len(correct_ages)):
        print(f"Actual Age: {round(correct_ages[i])},
                    Predicted Age: {round(ages_predicted[i][0])}");
        # write the data to the output csv file
        writer.writerow( [data_id[i], round(correct_ages[i]) ,
         round(ages_predicted[i][0])] );
        y_true.append(round(correct_ages[i])); y_pred.append(round(ages_predicted[i][0]));
        y_true_1 = y_true.copy(); y_pred_1 = y_pred.copy();
print ('prediction saved to', outputfile);
fig, ax = plt.subplots(figsize=(10,5))
ax.plot(range(len(y_true)), y_true, '-b',label='Actual')
ax.plot(range(len(y_pred)), y_pred, '-r', label='Predicted')
ax.legend(loc="lower right")
plt.show()
```

Executing scripts for a 500 PPG signal file:

```
.
.
data_id:2020564, Actual Age: 28, Predicted Age: 28
data_id:2020563, Actual Age: 36, Predicted Age: 38
data_id:2020571, Actual Age: 30, Predicted Age: 31
data_id:2020569, Actual Age: 45, Predicted Age: 43
data_id:2020564, Actual Age: 28, Predicted Age: 28
data_id:2020554, Actual Age: 30, Predicted Age: 31
data_id:2020543, Actual Age: 62, Predicted Age: 62
data_id:2020540, Actual Age: 64, Predicted Age: 72
data_id:2020539, Actual Age: 56, Predicted Age: 48
.
.
```

Figure 47: Testing the ANN prediction model on a file of 500 PPG signals

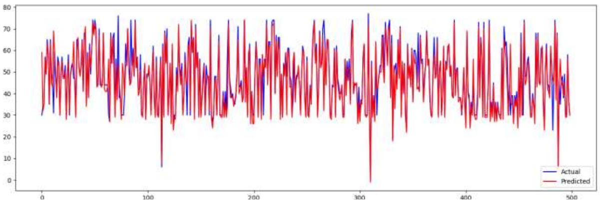

Figure 48: Testing the ANN prediction model on a file of 500 PPG signals

Executing scripts for a file of 3000 PPG signals:

```
.
.
data_id:2510263.0, Actual Age: 73, Predicted Age: 37
data_id:2510262.0, Actual Age: 71, Predicted Age: 27
data_id:2510261.0, Actual Age: 23, Predicted Age: 37
data_id:2510260.0, Actual Age: 30, Predicted Age: 38
data_id:2510254.0, Actual Age: 54, Predicted Age: 31
data_id:2510252.0, Actual Age: 61, Predicted Age: 70
data_id:2510250.0, Actual Age: 45, Predicted Age: 33
data_id:2510248.0, Actual Age: 41, Predicted Age: 36
data_id:2510245.0, Actual Age: 30, Predicted Age: 37
.
.
```

Figure 49: Testing the prediction model on a file of 3000 PPG signals

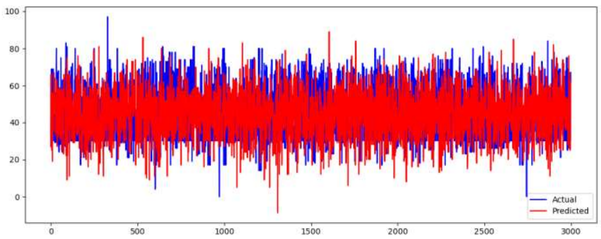

Figure 50: Testing the prediction model on a file of 3000 PPG signals

4 DISCUSSION

The analysis of PPG signals that evaluates biological age could play a vital role in smart city concepts by enhancing personalized health services. For example, cities could implement kiosks equipped with PPG sensors that automatically assess citizens' physical conditions and provide tailored health recommendations. These insights could be used to design preventive healthcare programs targeted at specific age groups or populations in need. Additionally, the data could aid public health management by identifying areas with higher health risks. This integration of PPG technology would foster cities that actively prioritize the well-being and longevity of their residents.

This paper presents an introduction to the estimation of characteristic parameters of PPG signals with the help of wavelet transformation. In chapter 2, estimation methods are assumed, which can be considered as an introduction to further work in this area. Assumptions are tested in chapter 3 on the model of neural networks and predictions based on the characteristic parameter of AI. Rough testing was conducted. The results are shown in graphs (Figures 27-30) and can be considered as guidelines for the development of the methodology.

In Chapter 2 and the description of the wavelet transform on PPG signals, the built-in functions in Python "modwt" and "imodwt" for the inverse transform were used as an example. In chapter 3 and generating the wavelet transform on the PPG signal file, the built-in functions "wavedec" and "waverec" were used for the inverse transform. In further works, the function "modwt" and "imodwt" can be used in testing in combination with the generation of neural network models.

In the following works, methods for selecting wavelets that are adequate for the estimation of other characteristic parameters described in chapter 1.3 can be assumed.

It does not necessarily mean that if a wavelet is suitable for extracting a characteristic parameter, it is automatically suitable for the next characteristic parameter (7 characteristic parameters are listed in chapter 1.3). Testing on neural networks can be very useful for confirming assumptions. Generation of neural network models based on other possibilities of wavelet transformation is also a topic for further work in this area.

REFERENCES

[1] John Allen, Photoplethysmography and its application in clinical physiological measurement, Article in Physiological Measurement · April 2007.

[2] Avolio A., The finger volume pulse and assessment of arterial properties, Journal of Hypertension, 2002.

[3] Webster J.G., Medical Instrumentation: application and design, Wiley 2010.

[4] Campolo D., New Developments in Biomedical Engineering, InTech, ISBN 978-953-7619-57-2, 2010.

[5] Hertzman A., The blood supply of various skin areas as estimated by the photoelectric plethysmograph. Amer J Physiol. 1938;1924:329-40.

[6] Reisner A, Shaltis P, McCombie D, Asada H., Utility of the photoplethysmogram in circulatory monitoring. Anesthesiology, 2008;108:950-8.

[7] Nilsson L, Johansson A, Kalman S., Respiration can be monitored by photoplethysmography with high sensitivity and specificity regardless of anaesthesia and ventilatory mode, Acta Anasthesiologica Scandinavica, 2005;49:1157-62.

[8] Chua E., Redmond S., McDarby G., Heneghan C., Towards using photo-plethysmogram amplitude to measure blood pressure during sleep, Annals of Biomedical Engineering, 2010;38(3):945-54.

[9] Awad A., Haddadin A., Tantawy H., Badr T., Stout R., Silverman D., The relationship between the photoplethysmographic waveform and systemic vascular resistance, Journal of Clinical Monitoring and Computing, 2007;21(6):365-72.

[10] Wang L., Pickwell-MacPherson E., Liang Y.P., Zhang Y.T., Noninvasive cardiac output estimation using a novel photoplethysmogram index, Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2009.

[11] Jubadi W.M., Mohd Sahak S.F.A., Heartbeat monitoring alert via SMS, IEEE Symposium on Industrial Electronics & Applications, 2009.

[12] Lu S., Zaho H., Ju K., Shin K., Lee M., Shelly K., Can photoplethysmography variability serve as an alternative approach

to obtain heart rate variability information?, Journal Clinical Monitoring and Computing, 2008;22(1):23-9.
[13] Takazawa K. T.N., Fujita M., Matsuoka O., Saiki T., Aikawa M., Tamura S., Ibukiyama C., Assessment of vasocative agents and vascular aging by the second derivative of photoplethysmogram waveform, Hypertension, 1998:32:365-70.
[14] Rubins U, Grabovskis A, Grube J, Kukulis I., Photoplethysmography Analysis of Artery Properties in Patients with Cardiovascular Diseases, Springer Berlin Heidelberg; 2008.
[15] Millasseau S, Kelly R, Ritter J, Chowienczyk P, Determination of age-related increases in large artery stiffness by digital pulse contour analysis, Clinical Science, 2002;103:371-7
[16] Allen, J., & Kyriacou, P. (2021), Photoplethysmography - Technology, Signal Analysis and Applications, eBook ISBN: 9780128235256.
[17] M. Misiti, Y. Misiti, G. Oppenheim, J.M. Poggi, Wavelet Toolbox For Use with MATLAB, MathWorks, 1997.
[18] N. Teofanov, Predavanja iz primenjene analize, Prirodno-matematički fakultet, Univerzitet u Novom Sadu, Zavod za udžbenike, Beograd, 2011
[19] S. Mallat, A wavelet Tour of Signal Processing, Academic Press, New York, 1998.
[20] D.P. Radunović, Talasići, Akademska misao, Beograd, 2005.
[21] H.G. Stark, Wavelets and Signal Processing, University of Applied Sciences, Germany, Springer-Verlag Berlin Heidelberg 2005.
[22] MathWorks, modwt, https://de.mathworks.com/help/wavelet/ref/modwt.html
[23] http://ataspinar.com/posts/machine-learning-with-signal-processing-techniques/
[24] https://www.deeplearningbook.org/contents/ml.html
[25] https://github.com/PyWavelets/
[26] https://pywavelets.readthedocs.io/
[27] https://scikit-learn.org/
[28] https://play.google.com/store/apps/details?.ctb.pulse.heartrate.camera.ecg4everybody
(the application through which the PPG signals used in this work were collected and generated in a file)

Contact information:

[1] **Zlatko RADOVANOVIĆ**, M.Sc.
Faculty of Information Technology, Alfa BK University, Belgrade
zlajo@pobox.sk
ORCID iD: 0009-0002-3829-2128
[2] **Stevan JOKIĆ**, PhD
professor, Faculty of Information Technology, Alfa BK University, Belgrade
ORCID iD: 0000-0003-4432-0172
[3] **Ivan JOKIĆ**, PhD
assistant professor, Faculty of Economics and Engineering Management in Novi Sad
ORCID iD: 0009-0008-0083-7675
[4] **Branislav GERAZOV**, PhD
professor, FEEIT, UCMS, Skopje, Macedonia
ORCID iD: 0000-0003-2498-6831
[5] **Ana KOVAČEVIĆ**, PhD
Zentrix Lab OÜ, Harju maakond, Tallinn, Kesklinna linnaosa,
Narva mnt 7-652, 10117, Estonia
ORCID iD: 0000-0003-4928-9848
[6] **Nenad GLIGORIĆ**, PhD
Zentrix Lab OÜ, Harju maakond, Tallinn, Kesklinna linnaosa,
Narva mnt 7-652, 10117, Estonia
ORCID iD: 0000-0002-9054-2799