Politecnico di Torino

DEPARTMENT OF MECHANICAL AND AEROSPACE ENGINEERING

Master's degree in Biomedical Engineering

SYSTOLIC BLOOD PRESSURE EVALUATION FROM PPG SIGNAL USING ARTIFICIAL NEURAL NETWORKS

Supervisor
Prof. Gabriella OLMO

Co-supervisor
Eng. Alessandro GUMIERO

Candidate
Filippo AUGUSTI

Academic year
2019/2020

List of Acronyms

ABP Arterial Blood Pressure

ANN Artificial Neural Network

BP Blood Pressure

CP Cardiac Period

CV Cross Validation

DP Diastolic Blood Pressure

DT Diastolic Time

ECG Electrocardiogram

Hb Deoxygenated haemoglobin

HbO_2 Oxygenated haemoglobin

HR Heart Rate

IBI Inter Beat Interval

ICT Information Computer Technology

LED Light Emitting Diodes

macc Multiply -Accumulate

MBP Mean Blood Pressure

MCU MicroController Unit

MLP Multi Layer Perceptron

PPG PhotoPlethysmography

PTT Pulse Transit Time

PWB Pulse Wave Beginning

PWE Pulse Wave End

PWSP Pulse Wave Systolic peak

PWV Pulse Wave Velocity

RAM Random Access Memory

ROM Read Only Memory

SP Systolic Blood Pressure

SUT Systolic Upstroke Time

WBAN Wireless Body Area Network

WLAN Wireless Local Area Network

WMAN Wireless metropolitan area network

WPAN Wireless Personal Area Network

Abstract

Nowadays cardiovascular diseases are the major cause of death. Among all of the origin of this problem, the commonest is hypertension. For this reason, continuous monitoring of the blood pressure is necessary for early death risk prevention. The optimum method for prevention and diagnosis is the creation of wearable devices within a Body Area Network for continuous blood pressure monitoring. Recent studies exploit the use of Electrocardiography and photoplethysmography devices combined in order to obtain the Pulse Wave Velocity or the Pulse Transit Time, i.e. measures which appear to be proportional with the blood pressure. The drawback of these techniques is that simultaneous recording from different regions of the body is requested. In such approach, the use of Artificial Neural Networks has been proved to be beneficial since it increases its accuracy. The newest technologies aim at further improving these techniques by the use of only the photoplethysmography (PPG) signal to perform the blood pressure measure, that is combined with the computational power of Artificial Neural Networks for further improvements. As input data some features representing the morphology of the photoplethysmography signal are chosen, because are considered correlated to the blood pressure. The aim of this project is to create a systolic pressure classifier based on Artificial Neural Networks. The classifier should be able to correctly predict the relative systolic pressure range for each PPG period. A dataset containing 124616 PPG periods features and relative systolic pressure values has been created from freely accessible MIMIC database. The features chosen represent each single PPG period morphology. The dataset has been balanced and the features normalized. Moreover, the systolic pressure values have been discretized basing on 7 preselected systolic pressure (SP) range. The chosen neural network was a Multilayer feed-forward back-propagation Neural Network, with 15 input neurons, two hidden layers of 120 and 600 neurons respectively, and 7 output neurons that represented the 7 SP classes. The hyperparameters optimization has been performed both manually and with a cross validation.

The results show an accuracy of 79% over the test set. Moreover, the chosen neural network tuned with the chosen parameters shows a good generalization (no overfitting). Furthermore, the precision and recall metrics show much higher performances over the external ranges classes that refer to either the lowest and highest pressures.

Contents

Chapter 1

Introduction

Hypertension is the main cause of cerebrovascular disease and of ischemic heart disease deaths. Furthermore, it is the commonest death factor throughout the world and causes millions of deaths per year [28]. Awareness, prevention, treatment and control of this epidemic is a public health duty, resulting in huge expenses and efforts. For this reason, nowadays low-cost method for continuous and remote hypertension control is highly requested, along with development of the primary prevention[12]. The new developments of Telemedicine monitoring systems respond to this problem. This monitoring can be achieved with wearable devices that allow the specialist to always keep the patients under control from a web-based link, with the support of low-cost monitoring systems.

1.1 Aim of the project

The aim of this project is to implement a non-invasive system for continuous systolic blood pressure (SP) monitoring. Contrarily to the current oscillometric blood pressure monitoring devices, the system must be cuff-less and suitable for continuous monitoring even for days or months.
This purpose can be achieved by implementing a Telemedicine monitoring system, that includes a Wireless Body area sensor. The embedded sensor would be a Photopletysmography one. In fact, a proportionality between the PPG signal morphology and the pressure value exist, but it is non-linear.
The Artificial Neural Networks (ANNs) are an efficient tool for investigating the non-linear relationship of data. Nevertheless, the ANNs use could be a more efficient approach than the signal processing and analysis.
Hence, the primary intent of the project is to create an Artificial Neural Network able to correctly and accurately detect systolic blood pressure from small PPG segments corrispondent to a single cardiac cycle period. In this way, the ANN needs only the morfology parameters of one PPG period, corresponding to roughly 1 second, resulting suitable for online beat-to-beat pressure monitoring on a wearable device. The implemented ANN is inspired to those already tested in literature [29, 40]. However, while these studies predict a numerical value, this project aims at implementing a multiclass classificator for various SP ranges. That is, having divided the possible systolic pressure values into the ranges shown in Table 1.1, the Neural Network duty is being able to assign every new PPG period to a SP range.

80 - 100
100 - 109
110 - 119
120 - 129
130 - 139
140 - 149
150 - 169

Table 1.1: Systolic pressure ranges division

If reasonably accurate and computationally light, the produced Neural Network would be then implemented within the microcontreoller unit (MCU)

of a ST Microelectronics wearable device for continuous beat-to-beat pressure detection. In fact, once trained, the artificial Neural Network reduces to simple aritmetics operations that, if they are not too numerous, can be performed on a MCU. The MCU should be embedded into a Wireless Body area sensor device that would be able to continuously acquire the data, analyze them though the implemented ANN and then send the predicted SP value to a storage center through the help of a Body Control Unit (a gateway).

1.1.1 Telemedicine

Preface

Within the next 10 years, 20% of the world population will be constituted by over 65 years old people. For this reason, a method for healthcare cost reduction and efficient prevention is a necessity for the modern societies. These are experiencing common problems such as: growing number of chronic patients due to life expectancy increasing; the demand of healthcare services much higher than the offer; governmental healthcare expenditures growing faster than economic growth. Telemedicine respond to this necessity [26]. By definition, telemedicine is an healthcare system that, exploiting the ICT transfer of biomedical data, offers the possibility of diagnosis, education or treatment from *distance* [17]. This results in low - cost and more efficient services. Of course, these systems do not replace the traditional healthcare structure, but improve their effectiveness and efficiency, also in doctor-patient relationship.

Telecommunication networks

Telemedicine relies on the connection links developed from Telecommunication networks technologies, that allow communication in a short, medium and long range [62]:

- The short range network (up to 30 meters) is called a *Wireless Personal Area Network* or WPAN: it exploits technologies such as Bluetooth, RFID, IrDA or ZigBee;
- The medium range networks (30 -100 meters) are called *Wireless Local Area Network* or WLAN and exploit the 802.11a, 802.11b, and 802.11g Wi-Fi protocols;

- The long range (more than 100 meters) *Wireless metropolitan area networks* or WMAN, exploit the IEEE802.16 and IEEE802.20 protocols. They cover longer distances with better quality-of-service (QoS) support than Wi-Fi [62].

In the late 1990's several techniques for compressing videos, images and biomedical data emerged, facilitating their transmission [20]. The bandwidth refers to the possible datarate of a trasmission channel: if it is large huge files can be sent (such videos or similar), else if it is small, a larger time is needed to upload the data acquired. Nowadays, the telecommunication aim is the trasmission bandwidth enlarging, meaning that more and heavier files can be sent in the unit time[20]. With the recent implementation of the 5GHz bandwidth, telemedicine is thought to further improve its efficiency.

Telemedicine services

Telemedicine services can be grouped into a few categories [17]:

- *Teleconsultation*: consists in the commuication between
  - two pair careers, for opinion exchange about a patient's case;
  - patient and a career , aiming at the creation of real - time feedback (consultation), in order to facilitate the physician decision making.

  This telemedicine modality is used in the 35% of cases.

- *Tele-education*: it is represented by everything that concerns clinical education on the internet or from teleconsulation, puclic education or academic study through the web.

- *Telemonitoring*: it consists in the monitoring of a patient vital signs from a remote location, with the aid of a device connected to the telecommuication networks. Each patient is controlled with monitoring systems, that gather the data and upload them on the web, and a clinician, which checks the data and takes actions in response. The majority of Telemonitoring systems are composed of five main parts [37]:
  - *data acquisition* system, that consists of an electronic system with an embedded sensor and usually a battery;

  - system for the *transfer of the patient data* to the clinician, based on telecomunication technologies;
  - system for the *aggregation of received data* from the acquisition system. In this way the patient history is virtually recreated in order to be able to correctly assess the patient status.
  - *possibility to take an action* in correspondence of an abnormal status of the patient.
  - *storage of data*, that can either be on the cloud or in a local machine.

- *Telesurgery*: it can either be:
  - *telepresence-surgery*, which is the practice by which the surgeon controls a robotic arm that actually perform the surgery in a remote location;
  - *telementoring*, which is a simple real-time video mentoring by an experienced surgeon, to another surgeon which is performing the surgical practice.

1.1.2 Body area wearable sensor

Telemedicine sensor devices used for continuous monitoring are usually wearable, but can sometimes be implanted into a patient [26]. The recent electronic improvements, especially regarding the MEMS introduction and the reduction of chips dimension and consumption, allow for portable and low power applications. The device application depends on the sensor type, which among all can be electrochemical (for example in ECG applications), mechanical (such as accelerometers) or optical (like in Photoplethysmography or PPG applications). Nowadays the wearable devices usually incorporate short range radio systems (WPAN), such as Bluetooth. The rest of data transfer is done by the Body Control Unit (BCU), that in conjunction with the WPAN, takes part in a WLAN as gateway. The BCU can also transfer the data over the Internet with a GSM connection and perform other actions, such as store data, run some pre-processing algorithms or send any clinical alarms. This is the principle of Wireless Body Area Network (WBAN)[60] (Figure 1.1).

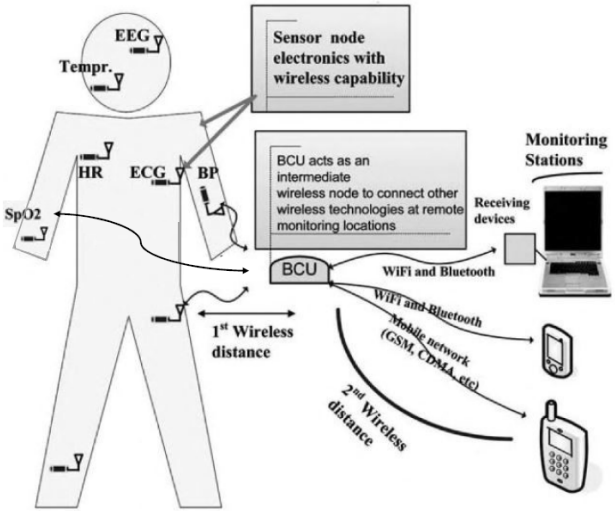

Figure 1.1: An example of WBAN diagram [60]

1.2 Theoretical framework

Blood pressure is an indicator of the cardiovascular system status, which can be compromised by many diseases, among which one is hypertension. The latter is a parameter that indicates the presence of other cardiovascular diseases and can be measured in many ways. The most recent methods exploit the use of either electrocardiography and photoplethysmography techniques combined or even just photoplethysmography. Moreover, in all cases the use of artificial neural networks helps in obtaining better results.

1.2.1 Cardiovascular system

The cardiovascular system distribute oxygen, nutrients and hormones throughout the whole human body and it removes the waste products. It is composed of the heart and blood vessels.

The Heart is composed of a particular muscle cells tissue, the *myocardium* and a specialized conduction system. The combination of these two elements produce a physiological pump for the blood inside the body. The human heart is composed of four chambers with different functions, two atria and two ventricles: while the atria receive the blood, the ventricles pump it by mean of a strong myocardium contraction.

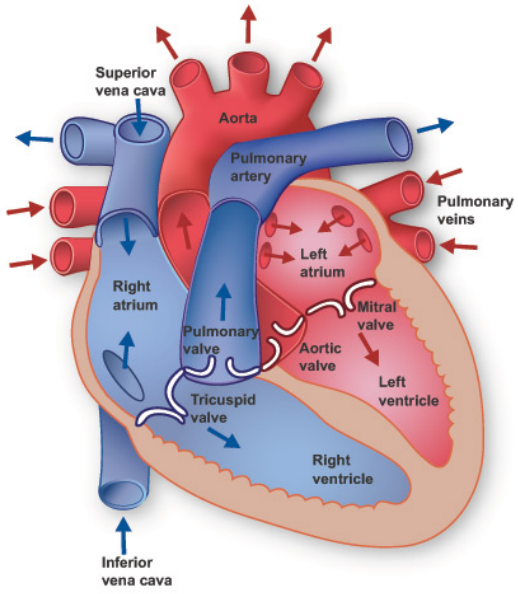

Figure 1.2: The human heart anatomy [55]

The four chambers are separated vertically by a thick wall called *septum* that avoids the left and right heart side blood to be mixed together. The reason of this left to right separation is the presence of two circulation circuits [25]:

- **pulmonary circulation**: a circle that passes through the heart and lungs, needed for the blood and heart tissues oxygenation.
- **systemic circulation**: the loop that delivers the oxygenated blood to the body periphery.

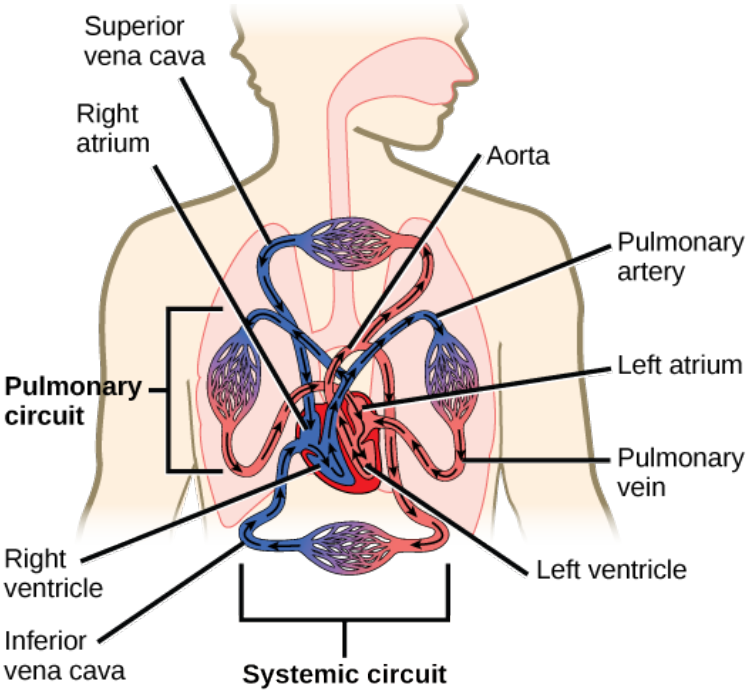

Figure 1.3: Diagram of the circulatory system [25]

In order to generate the correct sequence of the four heart chambers contraction, a **conduction system** generates electrical impulses and propagates them through a specifical pathway into the myocardium 1.4. The electrical pulse begins in the Natural pacemaker or *sinoatrial node* (SA), which is made of particular cells capable to automatically generate action potentials with a defined rhythm [55]. Hence, the electrical pulse depolarizes the atria first and then travels toward the atrioventricular (AV) node, located between the right atria and ventricle. Here, the signal is delayed and then spread through the bundle of His, located in the inter-ventricular septum. The bundle of His is divided into two branches, left and right, which continue in the Purkinje fibers. These two last conduction elements guarantee a coordinated contraction of the left and right ventricles, with a delay respect to the atria activation [56, 55].

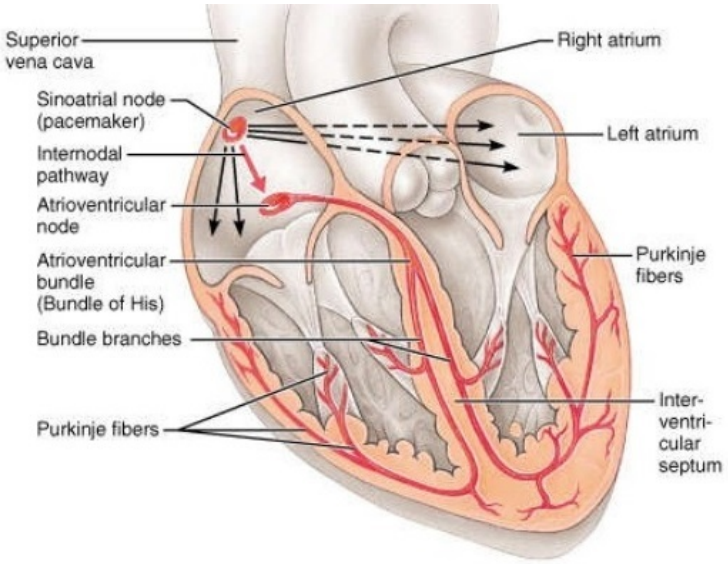

Figure 1.4: Diagram of the heart conduction system

Cardiac cycle The alternance of myocardium contraction and myocardium relaxation determines a periodical pumping mechanism. The period occuring from the beginning of a systole until the beginning of the next one is called the cardiac cycle [16]. It is constituted of two phases: diastole and systole [31].

- During the **diastole phase**, the blood is led into the right atrium through the inferior and superior vena cava. On the opposite side, the oxygenated blood enters into the left atrium, increasing its pressure. The tricuspid and mitral valves open when the atrium pressure exceeds the ventricle one. In this way the ventricles filling is started.

- During the **systole phase**, the blood is first injected into the ventricles by mean of an early atria contraction due to SA node electric impulse. Then follows the beginning of the ventricular contraction, called isometric contraction: in this phase the electrical pulse has reached the ventricles, causing their early contraction, which is still not strong enough to open the pulmonary and aortic valves. As soon as the ventricles contraction generates a pressure higher than the arterial tree, then the semilunar valves open and the blood is finally ejected. The diastole ends with the whole myocardium relaxation.

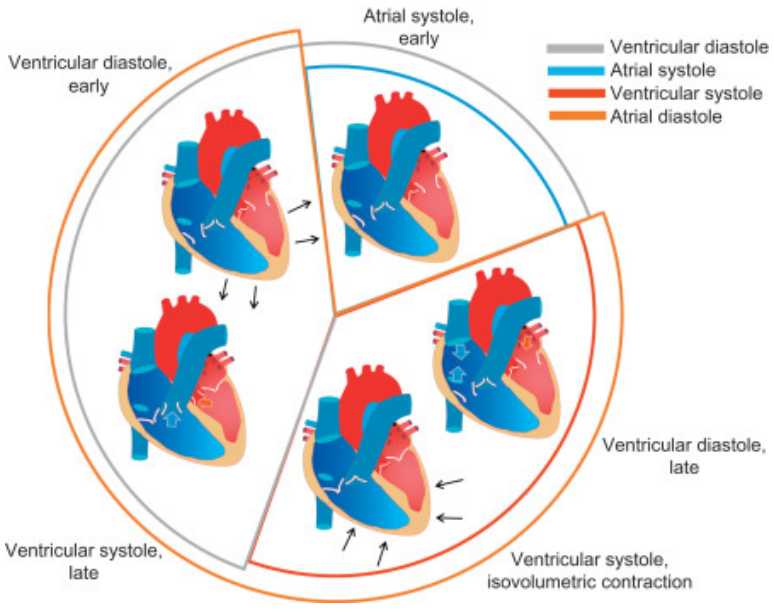

Figure 1.5: Cardiac cycle diagram [31]

Describing parameters and available measuring techniques

Many parameters that describe the circulatory system status exist, such as Heart rate, blood pressure, Cardiac output and many others. Each of them has a different clinical importance:

- Heart rate HR is the indicator of the number of heart contraction per minute and it is a strong health indicator. For example, an accelerate HR at rest could be an indicator of a cardiovascular disease. Furthermore, a rapid change of HR value can indicate a heart arrhythmia, which indicates a fail in the heart conduction system and can be a death risk factor. The HR at rest in healthy subjects is between 60 and 100 beats per minute.

- Blood pressure BP refers to the pressure exerted on the vessels walls by the blood. A high pressure or hypertension is an indicator of a cardiovascular disease and it is often correlated with an abnormal HR value. The BP value should stay within a physiological range because if it decrease too much, it means that the cardiac output is not enough to reach the peripheral capillaries, while if increases over healthy values it can be dangerous for the vessels integrity.

Electrocardiograpy The most common and reliable technique used in the field of cardiovascular system status assess is the **Electrocardiography** (ECG), that is a technique by which graphically represent the electrical activity of the heart. It is measured at the body surface level and requires the measure of the voltage difference between two or more sites of the body, though its optimal configuration requires 12 derivations over the limbs and the chest. Its waveform represent the depolarization and repolarization of the myocardium during the cardiac cycle.

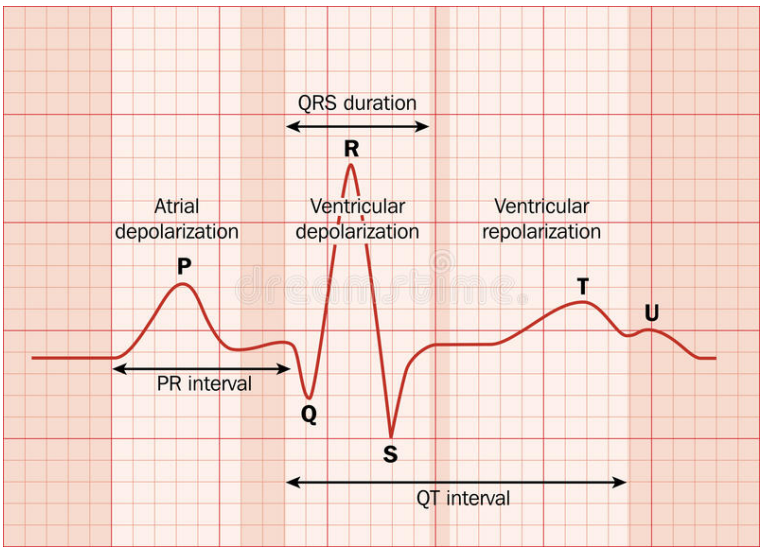

Figure 1.6: ECG typical waveform representation

Hence an abnormality in the ECG is an indicator of a cardiovascular disease. By means of ECG can be calculated:

- HR
- Heart rate variability
- R-R wave (each sample represents the distance between two subsequent R peak)
- Various forms of arrhythmias indicators
- other parameters useful for vascular diseases diagnosis

Photoplethysmography A less classical approach to the cardiovascular parameter monitoring is the **Photoplethysmography** (PPG), which is a simple and low cost optical technique. The PPG signal reflects the blood volume fluctuation into the superficial capillaries of the skin, that reflects the cardiac cycle behaviour.

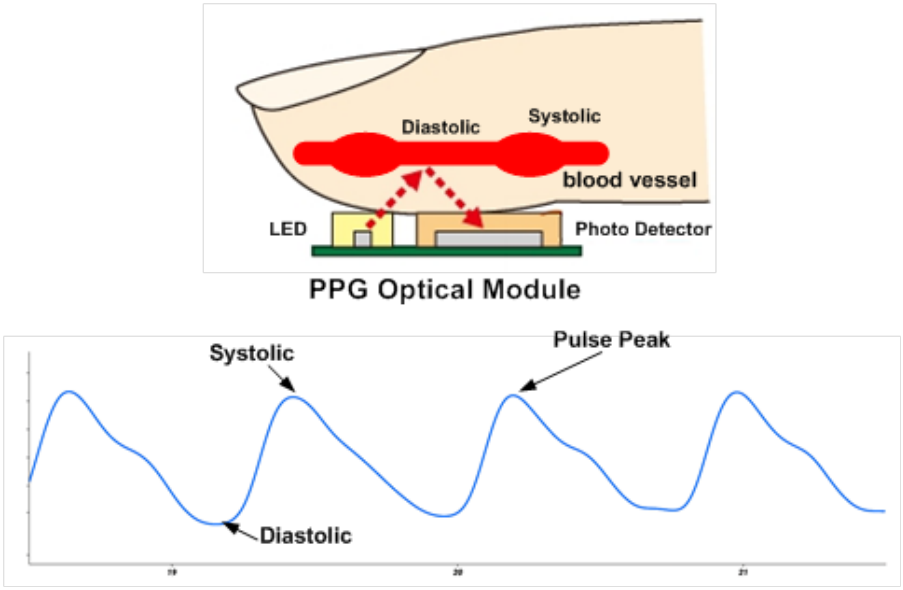

Figure 1.7: PPG typical waveform representation

By PPG is possible to obtain informations about:

- Heart rate
- Respiration
- SpO_2
- Heart rate variability
- Blood pressure

The blood pressure

Depending on the vessel in which the BP is measured, it is referred as venous or arterial pressure. The venous pressure is much lower than the atrial one,

because the veins do not receive the direct heart ejection thrust. For this reason, venous pressure is not a common pressure parameter used for the health status description. Yet, the Arterial blood pressure (ABP) is used as a health status descriptor, since its abnormal values are strong indicators of circulatory system diseases [16].
The first vessel that the blood encounter is the aorta, a thick-wall artery with a lot of elasticity and a larger diameter than the average vessels [55]. The aorta and other arteries elasticity is very important to maintain the ABP at similar values: if they were stiff, the ABP would increase at very high values and would be dangerous.

Arterial blood pressure waveform The ABP waveform follows the blood volume profile during the cardiac cycle: the more is the blood volume inside the artery, the more it is stretched and, hence, the higher is the ABP. The ABP increases during the systole, reaching a peak of intensity at the complete contraction, and then decreases during the diastole until it reaches its minimum, in correspondence of the complete relaxation of the heart. Hence, ABP waveform is a periodic series of peaks and troughs, in which the maximum peak represents the systolic pressure SP and the minimum through is called the diastolic pressure DP [1, 16].

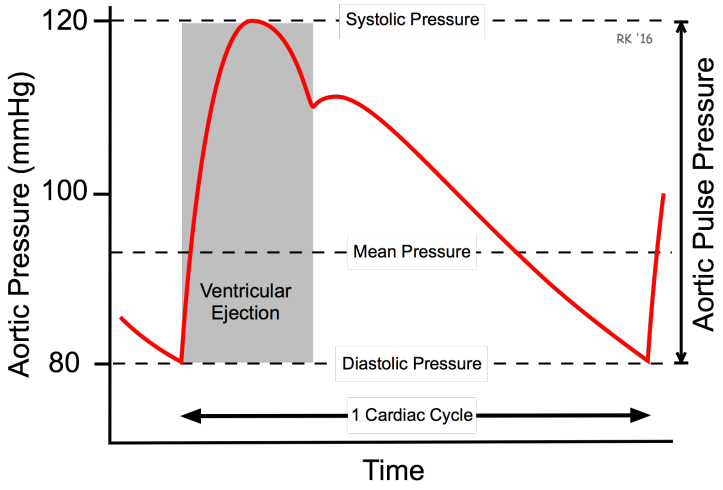

Figure 1.8: Aortic pressure variation during the cardiac cycle [1]

The blood is forced from the left ventricles into the aorta, thus creating a pressure wave that propagates along the whole cardiovascular system. The ABP is composed of a stationary and pulsating component. **The stationary component** is called *Mean Pressure value* (MBP) and represents the effective organs perfusion pressure. Moreover, this steady state component is correlated only to the cardiac output and total peripheral resistance. **The pulsating components** are much more complex than the steady one. The SP is determined by haemodynamic factors such as arterial stiffness, stroke volume, and left ventricular ejection fraction. The DP, on the other hand, is due to total peripheral resistance, heart rate, arterial stiffness, and systolic blood pressure [53, 16]. Furthermore, the ABP shows an increasing trend for older age [53].

Importance of pressure monitoring SP and DP values are used to understand if the pressure status is within specific healthy ranges. In fact, a too low SP means that the peripheral body regions are not perfused enough with nutrients and oxygen, while a too high SP is a risk for the vessels and organs integrity. Indeed, although the SP and DP values can change during time due to autoregulation, a long time lasting hypertension could be fatal. Nevertheless, altered values of SP and DP can be indicators of atherosclerosis or another cardiovascular disease and can be used to reduce the death risk [28].

Arterial blood pressure ranges (mmHg)			
Category	Systolic pressure	and	Diastolic pressure
Optimal	< 120	and	< 80
Normal	120-129	and/or	80-84
High normal	130 - 139	and/or	85 - 89
Grade 1 hypertension	140-159	and/or	90-99
Grade 2 hypertension	160-179	and/or	100-109
Grade 3 hypertension	>180	and/or	>110
Isolated systolic hypertension	≥ 140	and	<90

Table 1.2: Table reporting the ABP ranges classification of American Heart Association [32]

For all the reasons discussed, continuous pressure monitoring is very important for the medical specialist in order to adjust its patient therapy. Nev-

ertheless, an accurately chosen pressure monitoring system is fundamental for early detection and complications prevention. This type of monitoring system can be created by the implementation of a PPG sensor tele-monitoring device.

1.2.2 Photoplethysmography

The term Photoplethysmography (1930) refers to a non -invasive technique for measuring the volume of blood flowing within the vessels[49]. The pulsating behaviour of arterial blood volume has such a clinical importance, that it can be used for pressure monitoring. Nevertheless, it overcomes some of the limits of the classical methods for detecting cardiovascular diseases, such as ECG. In fact, ECG devices need to acquire the signal from at least two different regions of the body, hence needing more devices connected wirelessly or a single device with more sensor connected by cables. This aspect makes ECG recording bulky, especially when the sensor-device connection is not wireless. On the contrary, the PPG devices require only a sensor that is usually integrated into the device case, resulting in an easier and more comfortable set up for heart monitoring[15].

Light-matter interaction principles

There are three ways by which an incident light interacts with the tissues: reflection, refraction and absorption. The transmitted light is the portion of the incident light that have not been refracted, scattered or absorbed [38] by the tissues.

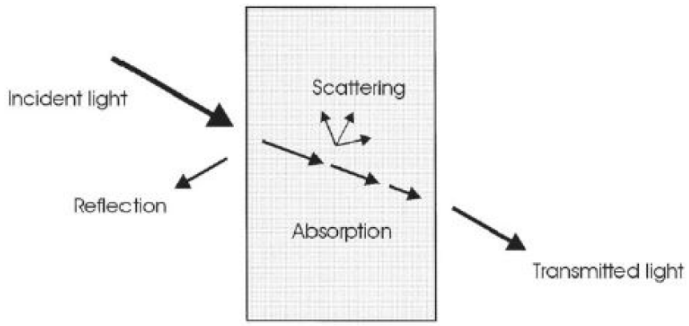

Figure 1.9: Light-matter interaction principles [38]

Reflection is defined as the incident light wave return back. When the wavelength of the radiation is smaller than the discontinuities of the surface, it is called *specular reflection*. In this case the reflected angle is equal to the incident one $\theta' = \theta$. Whenever the incident wavelength is larger than these irregularities, the *diffuse scattering* occurs, by which the beam is broken down and re-emitted in several directions.

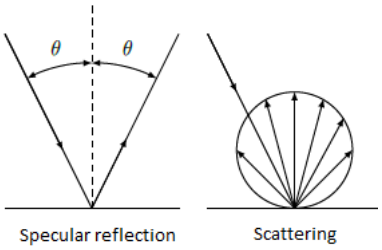

Figure 1.10: Reflection versus diffuse scattering [38]

Refraction is represented by a change in the incident light wave speed along its propagation direction. When this happen, the light direction is changed according to the Snell's Law:

$$\frac{sin(\theta)}{sin(\theta'')} = \frac{v}{v'} \tag{1.1}$$

where θ'' is the angle of refraction and v and v' are the velocities of light before and after the reflective surface respectively.

Absorption is the phenomenon by which a portion of light is retained by the tissue. When travelling through the biological tissues, the light is attenuated in a proportional way to the tissue absorption coefficient α. If the hypothesis that the tissue is composed of only arteries (homogeneous tissue) can be made, the Lambert-Beer Law describes this behaviour. This law states that in a homogeneous medium, light intensity decays exponentially as a function of path length (l) and light absorption coefficient (α), corresponding to medium properties at a specific wavelength:

$$I = I_0 e^{-\alpha l} \tag{1.2}$$

where I is the intensity of the transmitted light through the medium and I_0 is the emitted light intensity.

The PPG sensor

The PPG sensor works in contact with the skin and can be placed in various regions of the body such as nose, earlobe, finger tip , wrist and so on. The PPG sensors are very tiny and are composed of two elements: light source and photodetector:

- **light source**: for this purpose semiconductor technologies such as LEDs are exploited. The LED intensity and emission band have to be carefully chosen in order to not ionize the cells and the organic tissues[3]. Furthermore, the signal characteristics change together with the bandwidth of the light emitted (for example there is a slight difference between red and green light wavelengths), hence the choice have to be calibrated on this as well.

- **photodetector**: it is usually a photodiode that is able to capture the light that travels through the irradiated tissues. This light is then converted into an electrical output signal. Because the photodetector can not capture all bandwidth radiations, it has to be chosen coherently with the emitting LED wavelength choice[3].

The sensor configurations

Basing on the light source and photodetector positioning it is possible to define two different configurations [18].

- **transmission mode**: light source and photodetector are placed in diametrically opposite sides, facing each other. The photodetector catches the light not absorbed by the tissues. Only a few quite thin body regions are suitable for this technique, such as earlobes, fingertips, because they allow enough light to pass through them. However, this technique allows to isolate in a better way the sensor from the environment light, that can produce artefacts [3].

- **reflection**: light source and photodetector are placed on the same side. An optical shield is needed between the light diode and the photodetector for artefact avoiding. In this case the scattered emitted light is detected from the photodetector. Moreover, because this technique does not require the measured body region to be thin, it can be used

in many other regions, such as wrist, forehead, limbs or chest. However, this method is more sensible to motion and environmental light artefacts. [49, 35].

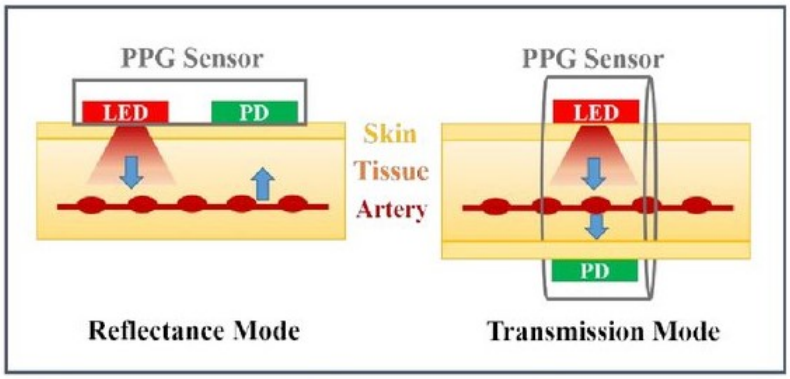

Figure 1.11: An example of transmission and reflectance mode [6]

Skin influences on the trasmitted light

Being the PPG sensors placed on the skin, its properties are very important for the outcome of the sensing. Human skin can be divided into *epidermis, dermis* and *hypodermis* [23].

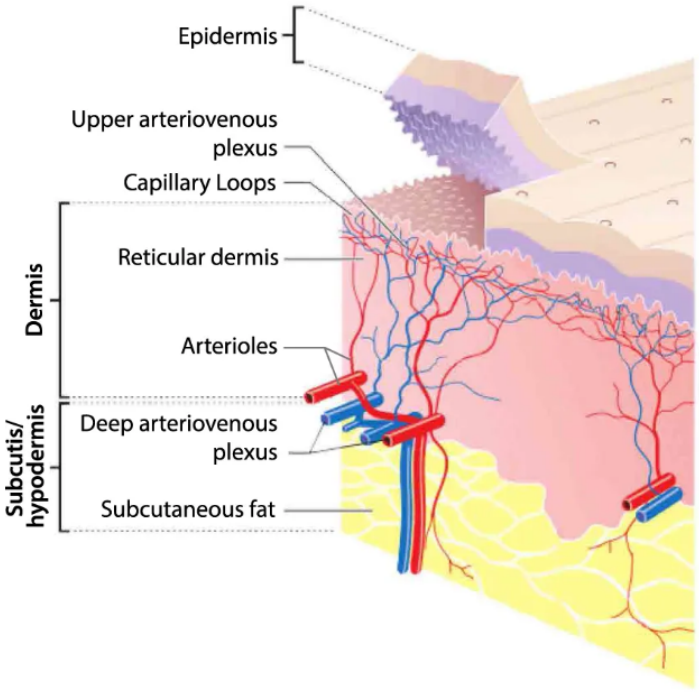

Figure 1.12: Skin layers schematic representation [34]

The *epidermis* is 0.027 - 0.15 mm thick and does not have blood supply, hence it represent an element of light obstacle. The 90% of its cells are keratinocytes and continuously shed and replaced. Some other cells, called melanocytes, contain the melanin, that is a substance responsible of some dangerous wavelength absorption for skin protection[5]. The *dermis* layer is 0.6 - 3 mm thick, is the one that contains the smallest skin vessels [14]. Finally, the *hypodermis or subcutis*is much thicker (1-6 mm) and contains the skin largest vessels together with connective tissues [22].

Skin layers response to light incidence Due to their different thickness and being composed of different tissues and components, these three layer respond differently to incident light. For further understanding, the absorption spectrum of the different skin component is reported.

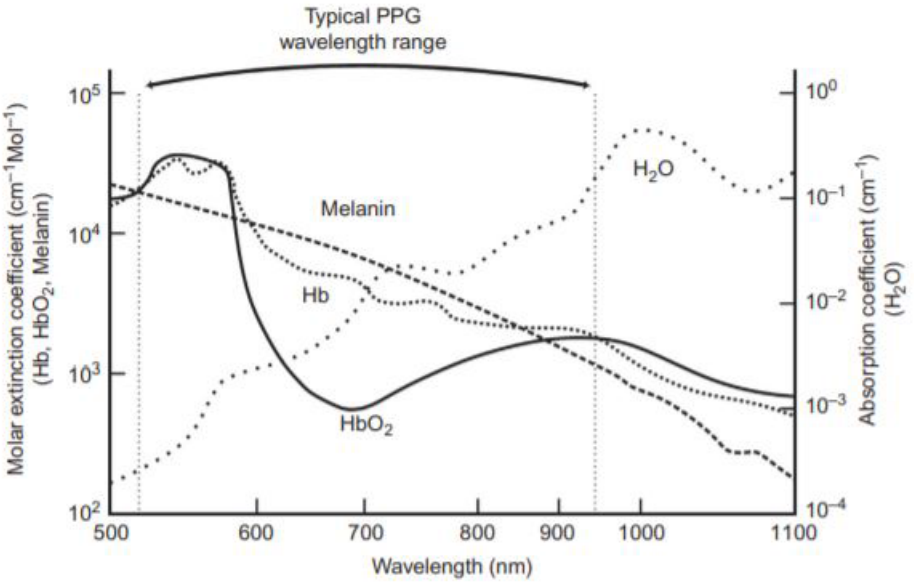

Figure 1.13: Absorption spectrum of *water, melanin, Haemoglobin, Oxygenated haemoglobin* and *deoxygenated haemoglobin* [30]

Water composes the majority of each tissue and allows only wavelengths shorter than 950 nm: if higher, they are strongly absorbed by it and do not penetrate much. *Melanin* spectrum present a very high peak in correspondence of 510-600 nm. However, because it is restricted to a very thin layer, even with a high absorption coefficient its effect on light propagation is very low. *Haemoglobin* is the main component of our blood and can be found in three configurations: dysfunctional haemoglobin (which does not bind with oxygen), oxygenated haemoglobin (HbO_2) and deoxygenated haemoglobin (Hb). Moreover, different wavelengths reach a different depth into the skin (Figure 1.14). Because of the skin spectrum properties described above, the wavelength chosen in PPG applications ranges from 510 nm (green) to 920 nm (red). The choice is very important and depends on the desired application and on which configuration (reflection or transmission) is used.

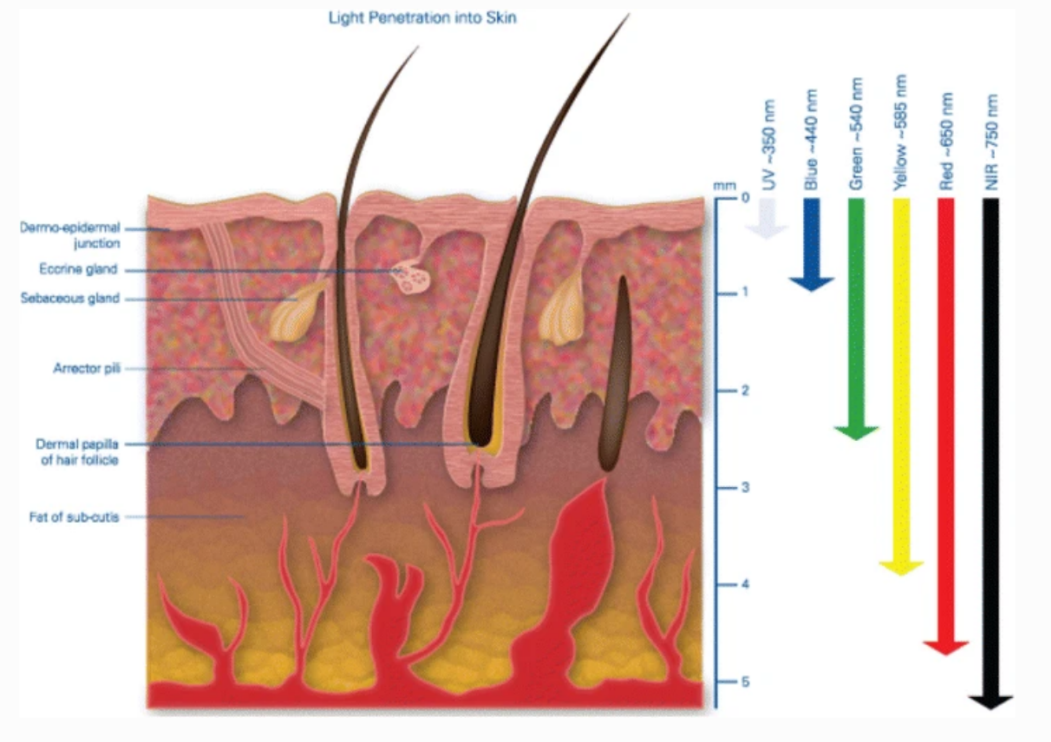

Figure 1.14: Depth reached from different wavelength that incide to the skin [4]

Because **red wavelength** can reach up to 5 mm depth into the skin, the PPG signal oscillatory shape is associated to the pulsating nature of the arteries.

The **green wavelength** results more suitable for wearable devices applications. The nature of the signal in this case is still associated to the pulsating nature of the arteries, but in an indirect way. Indeed, the green wavelength can reach only roughly 3 mm in depth, that is a skin region in which we find only capillaries. Because capillaries are not characterized by a pulsating flow, the nature of the PPG signal is in this case associated to a capillarity density increase as a consequence of deeper layer pulsating volume change. As to say, we still measure the arterial pulsating flow, but from the consequences it implies in the more superficial layers [24].

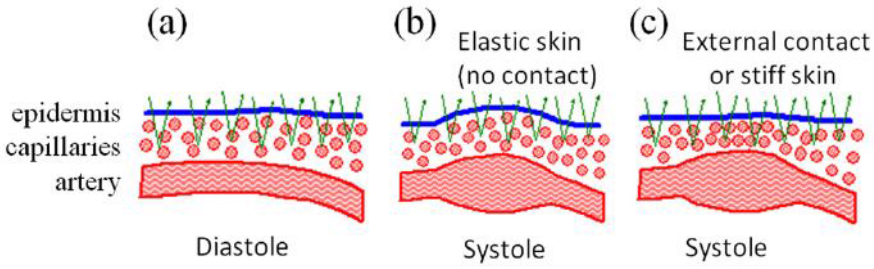

Figure 1.15: Capillary density change in epidermis layer due to deeper layers arterial pulsating flow [24]

.

PPG signal waveform

Some experimental studies show that the PPG signal intensity is inversely proportional to the blood volume in the tissue[19, 42] (Figure **??**). Although it applies to both configurations, for an easier understanding it is better to consider the transmission mode first. The key concept is that the tissues are less opaque than the blood, that is, the blood absorbs a higher amount of light than the tissues. Thus, being the diastole characterized by a less blood volume into the vessels, during this phase there is a high amount of transmitted light and, hence, low absorption. Contrarily, the increasing blood amount during the systole results in a low transmitted light measure, indicating high absorption.

PPG signal waveform is composed of [52]:

- a **DC component**, due to respiration, autoregulation an sympathetic nervous system activity;
- an **AC component** that reflect the cardiac cycle periodical activity. This is the most informative component between the two.

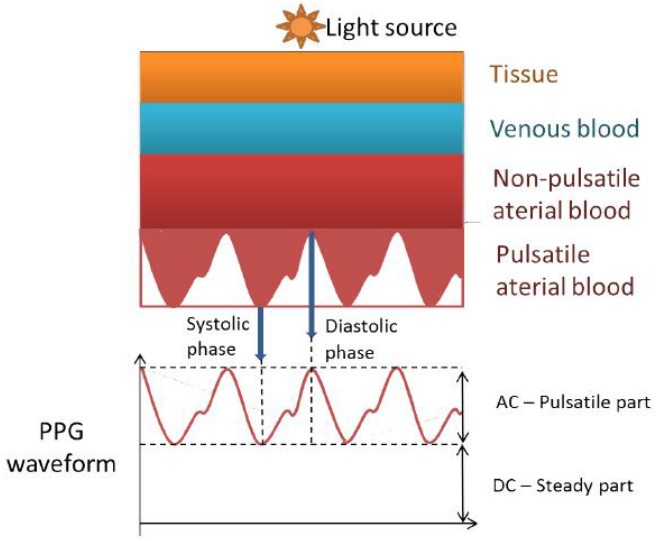

Figure 1.16: PPG waveform [52]

The AC component has a characteristic periodic shape (see Figure 1.16), which is composed of a so called *catacrota* (descending phase) and a *anacrota* phase (rising phase). The *catacrota* is due to the blood vessels stretch for the blood volume increase in correspondence with the systole. On the other hand, the increasing intensity during the *anacrota* is due to the progressive decreasing amount of blood during the diastole. Usually, for a more natural comprehension of the signal, the PPG signal is inverted. In this way the intensity increase represents an increase in blood volume [2].

The *anacrota* can vary significantly from subject to subject, because it is affected by vascular conditions such as arteries stiffness. Usually it is composed of a first *pre-dicrotic dip*, followed by a *dicrotic notch* and a final dip at the end [46] (Figure 1.17). The dicrotic notch is due to a reflexed wave, caused from arterial elasticity. This aspect of the *anacrota* is lost when the monitored patient suffers from vascular diseases that increase the vascular resistance. If the vascular resistance increase, the dicrotic notch can be invisible if the signal is acquired in periphery (for example fingertip).

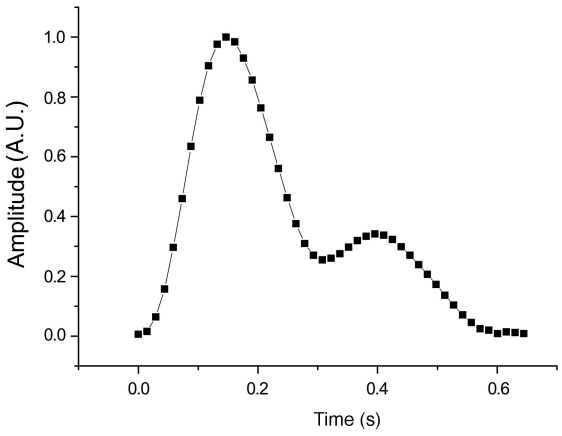

Figure 1.17: Inverted PPG waveform of an healthy adult [46]

In general, the characteristic points of the PPG waveform are named as illustrated in (Figure 1.18). The pulse wave begin (PWB) represents the start of the systolic phase, that ends at the pulse wave systolic peak (PWSP). The pulse wave end (PWE) indicates the end of the diastolic phase and the time between PWB and PWE is called pulse wave duration. The time elapsing from a PWSP to the consecutive, usually expressed in ms, is called inter-beat interval (IBI) and has high correlation with the inter-beat R-R interval of the ECG signal. (Figure 1.18).

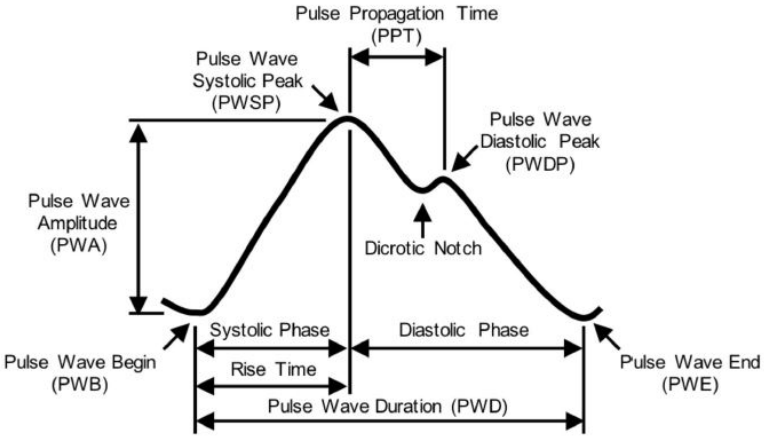

Figure 1.18: PPG descriptors diagram [46]

1.2.3 Artificial Neural Networks

Artificial intelligence has been defined in many ways, *machines with minds*[59], or *the study of creating machines that perform functions which require intelligence when performed by people* [41]. It comes from the human will to make computers *think* like the human being do, in order to increase their potentiality and contemporarily to understand the *thinking* functioning [44]. Neural networks are considered part of Artificial intelligence and their primary peculiarity is that they can *learn* from the data, through a process called *training*. Of course, in order to imitate the human thinking, the Artificial Intelligence is inspired to the brain and the physiological neural network functioning.

Brain physiology It is well known that the neuron is the fundamental unit of the brain and the rest of nervous system. It is composed of:

- several *dendrites*, by which it receives signals from other neurons;
- a *soma* (or cell body), where the nucleus is located, that elaborates the received input signals and produces an output signal;
- an *axon*, a much longer fiber that, acting as an isolated conductor, serves as output signal propagator;
- the *Myelin sheath*, that is the axon insulating material. It is composed of Schwann cells separated from many gaps called Node of Ranvier.

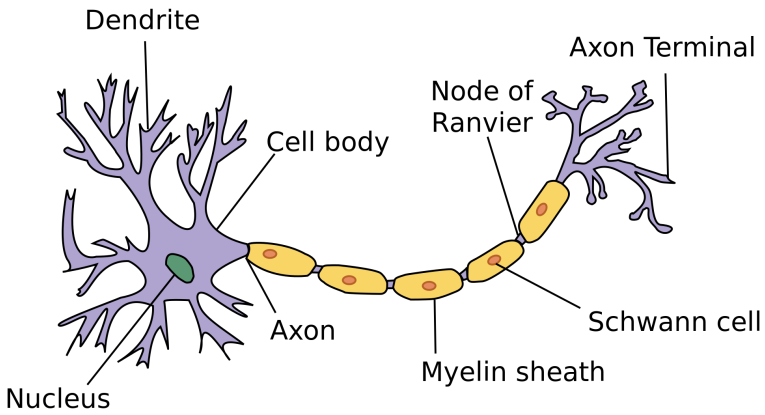

Figure 1.19: Diagram of the human brain neuron [57]

At the end of each branch, the synapses manage the communication between different neurons: a specific connection between two neurons is strong or weak depending on the frequency of excitation: if a connection is never used, becomes weak.
When a neuron cell body receives many impulses from many different stimulating neurons, these stimulus are **summed** spatially and temporally. Within the input stimulation summation a different importance is given to each stimulating neuron, basing on the specifical connection strength. If the intensity of the summation overcomes a **physiological threshold**, then the neuron is *fired*. In this way, the neuron produce an action potential and transmits it as output signal through the axon. Despite the exact way the brain works is not really known, it has been nowadays established that the brain *thinking* is the result of specific path firing among the whole set of neurons connections [44].

Following this pathway, the ANNs attempt to emulate this working principles by the recreation of the physiological structure of the brain:

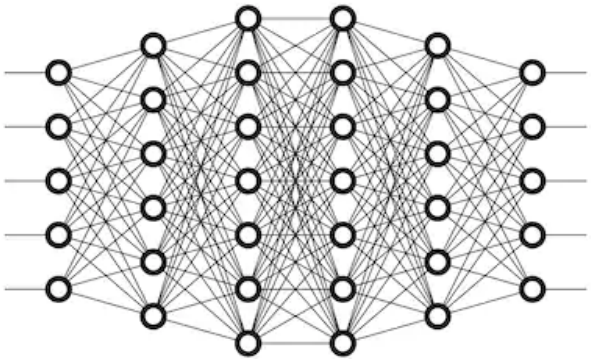

Figure 1.20: A simple schematic of ANN

- neurons are represented by single computational units or *perceptrons* that receive inputs and create outputs;
- axons, dendrites and synapses are summarized into the *links* among perceptrons;
- each neuron-to-neuron interaction is characterized by a different strength, defined as a *weight*.

Input data The input data can be given either as **raw data**, such as images or temporal series, or as **features**, that are a list of preset parameters characterizing each input element. The data type depends on the need of neural network architecture that is intended to be used. Moreover, the dataset can be either **labelled** or **unlabelled**. Labelling means assigning to each sample the correct output label (target).

input	feature1	feature2	...	feature i	...	feature n
1						
2						
.						
.						
.						
N						

Table 1.3: Unlabelled dataset

input	feature1	feature2	...	feature n	target1	target2	...	target n
1								
2								
.								
.								
.								
N								

Table 1.4: Labelled dataset

The single perceptron

In the field of ANNs, the single unit is called **perceptron** and was introduced by McCulloch and Pitts in 1943 [33]. As a parallel to the biology, it first performs a weighted summation of the inputs, applies a threshold to its result and produces an output.

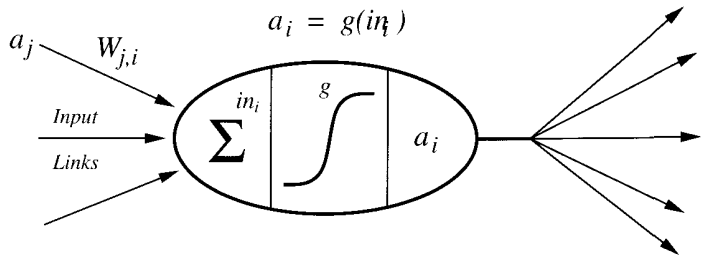

Figure 1.21: Perceptron: the single unit of Artificial neural networks [44]

Taking into consideration a single perceptron i, every input a_j is multiplied by a weight w_j. Then, all inputs are summed, producing the result:

$$in_i = x_1 + \sum w_{j,i} * a_j \tag{1.3}$$

where x_1 is called the *bias* and defines the threshold of activation of the perceptron. This input is then passed into a non-linear function or *activation function* $g(in_i)$. This function determines an activation level a_i, that is propagated as output:

$$a_i(x) = g(in_i) \tag{1.4}$$

Being a simple sum of weighted inputs, the single perceptron is just a different representation of a linear equation, that varies basing on the bias and weights values.

Activation functions The activation function is the non-linear part of the perceptron computation and its choice determines a different behaviour.

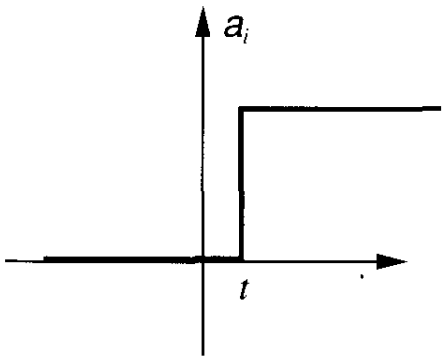

Figure 1.22: Step activation function [44]

The easiest function that represent the actual neurons behaviour is the step function, which output is set to one only when a certain threshold has been reached from the input. The step function decides if the neuron either fires or not. By the way, the step function introduces a very rigid threshold that often results unusable. The *sigmoid* is the most common and fulfilling activation function is the , that defines an activation level between 0 and 1 basing on its non-linearity.

$$g(x) = sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{1.5}$$

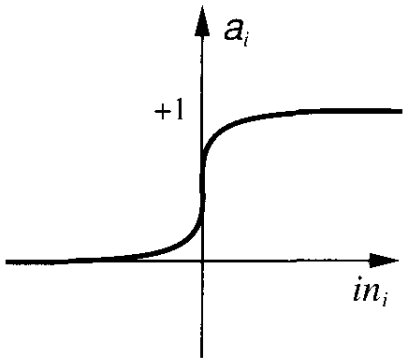

Figure 1.23: Sigmoid activation function [44]

A lot of other activation function are used in the practice: the choice among all possible activation functions is made basing on the ANN architecture and application.

ANN architectures

Many different Architectures of ANNs exist [27] and are used for different purposes. However, the basic principles are common to any of them [51]. Usually single units are grouped in *layers*. By linking different layers neurons it is possible to create infinite architectures. Usually, by increasing the complexity of the architecture, that is the number of layers (depth) and their neurons number (width), it is possible to create more complex non-linear functions.

Basing on the direction of the information within the ANN, these can be defined as **feed-forward** or **recurrent**. Within the recurrent ANNs the information can spread forward and backwards thanks to their bidirectional links and capability to form loops. On the other hand, within the feed-forward ANNs the information can spread only in one way, from inputs to the output.

An ANN is composed of:

- **input layer**: it is not considered a layer of the ANN because it only represent the inputs given to it. Its width is equal to the number of *features*;
- an optional number of **hidden layers**, each one having an optional width;
- **output layer**: it represents the output of the ANN. In case of regression problems, each output layer neuron returns a numerical prediction. Otherwise, in case of classification problems, each output layer neuron correspond to a different class. In the latter case, the fired neurons (the green neuron in Figure 1.24) are the indicators of the predicted classes.

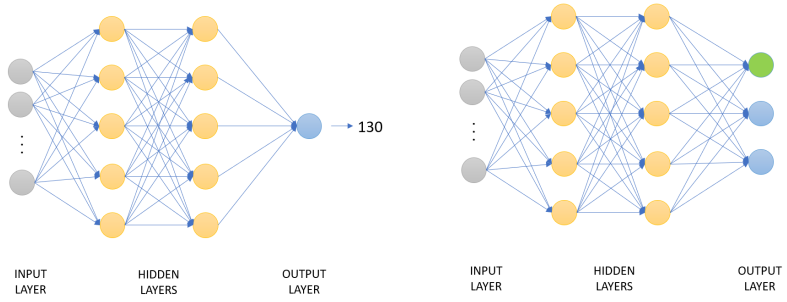

Figure 1.24: A schematic difference between Regression and Classification problem related ANN architectures

An ANN with more than one hidden layer is called Multilayer perceptron (MLP). In the MLP every neuron is fully connected with all the neurons of the previous and subsequent layers and is not connected to the same layer neurons [44]. As it was said before, the MLP can have an optional depth and hidden layers width. A more complex MLP can detect some data characteristics that are not easy to be detected by the human user. However, finding the optimal architecture is a difficult task because a too small ANN causes *underfitting* and a too big ANN generates *overfitting*. Underfitting is the inability of the ANN to find the correct peculiarities of the data, while overfitting refers to the fact that the ANN learns too well to recognize the training set inputs, but lacks of generalization to new unseen samples [51].

The learning process

The learning process consists in an algorithm that iteratively evaluates the input data and update the ANN weights values basing on the result obtained. It is also called **training algorithm** and it can be:

- **supervised**: the ANN knows the correct output and modifies its internal parameters in order to achieve a prediction as close to the ground truth as possible. The intent of this modality is building an approximator. In this case the training set must be **labelled**.
- **unsupervised**: the ANN is provided with only the input data and does not know the output. It is duty of the ANN to find relationships and patterns that describe the data [51]. The aim of the unsupervised

learning is to perform a clusterization of the input data. In unsupervised learning the training set is **unlabelled**.

A complete iteration over the training dataset is called *epoch*. The weights can be updated after each epoch, or after the iteration over a training set portion, of a desired dimension also called *batch size*. Although the typical batch size is between 32 and 512, it can vary from 1 to the entire training set dimension and it is usually a power of 2. Dividing the training set into batches is used in order to insert some variability into the dataset. In fact it is known that the use of the whole training set, if big, can produce lack of generalization: thus, the introduction of variability becomes a very important factor in overfitting risk reduction. In the **Supervised training**, it is possible for the neural network to estimate the loss at each iteration. The loss is the penalty for a poor prediction, that is expressed by a number indicating the error on a single prediction. The loss varies as a function of the weights combination: it is useful to describe it with the aid of the **cost function** $J(w_{1,1}, w_{2,1}, ..., w_{p,q})$ (or $J(W)$, where W represent the weight space). The cost function assigns a loss value to each input weight combination.
If a problem consists of an only input, the loss function is a single curve assigning a loss value to every weight assigned to the input value.

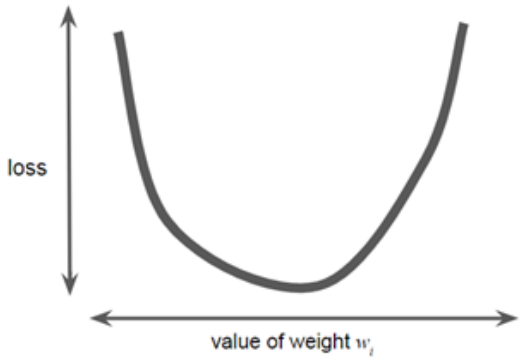

Figure 1.25: One-dimensional cost function as a function of the weight w_1 [8]

If an ANN has two inputs, then the cost function is represented on a plane and a different error value will be associated to each couple of weights w_1 and w_2.

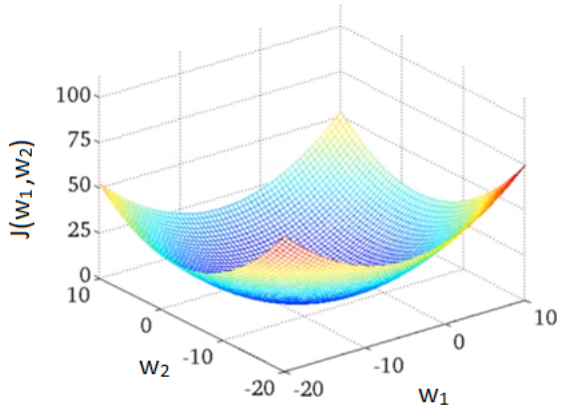

Figure 1.26: Two-dimensional cost function representation [8]

As the weights space W size increases, the cost function spreads over more dimensions and becomes computationally very heavy. The loss function is fundamental in the learning process, which intent consists in finding the optimum combination of weights ($w_{i,j}$) and biases x_{1j} that minimize the loss. However, instead of the cost function computation and subsequent minimization, lighter or faster techniques have to be used: these are called the **optimization algorithm**. The optimization algorithms exploit the cost function for the loss defining, but bypass its computation over the whole W space, by finding alternative strategies and calculating the loss only for a reduced portion of W. The most used optimization algorithm is the *gradient descent* and most of the rest of optimizers are inspired to its working principles or even a slightly modified copy of it.

Gradient descent algorithm Starting from a random initial point, the loss partial derivative is calculated over all the W dimensional directions and inserted into the gradient function, which indicates the direction of greatest increase of the function. If its negative value is taken, -∇f, then it is possible to move towards the minimum of the loss function. Then, the calculated gradient value is inserted into the weight updating equation:

$$w_{i,j} = w_{i,j} - \nabla \frac{dJ(W)}{dw_{i,j}} \tag{1.6}$$

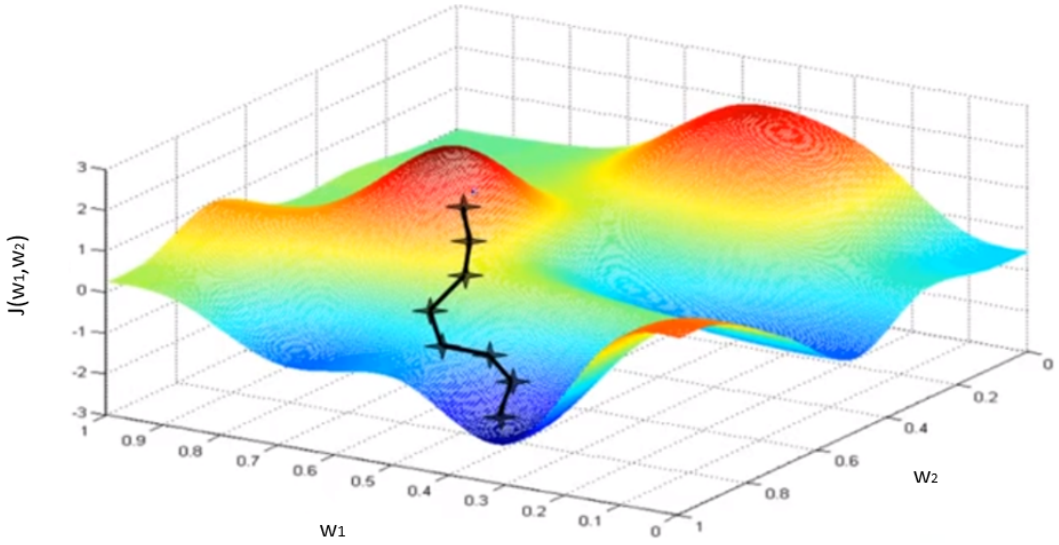

Figure 1.27: Gradient descent working principle [8]

For the weight update, the MLP exploits a back-propagation algorithm, that basing on the calculated loss, back propagates it backward in order to modify the weights according to it [44].

back-propagation algorithm In order to perform this algorithm, it is needed to calculate the *J(w)* first. In fact, in this way the algorithm can proceed towards its aim, that is the calculation of the partial derivatives $\frac{\partial J}{\partial w}$ and $\frac{\partial J}{\partial b}$ of the cost function J over the weights space and the biases, for each layer.
Let us consider the number m of input training examples, the number of network layers L, a loss vector $\delta(l)$ composed of a loss $\delta_j^{(l)}$calculated for each node j in the layer l and a matrix $\Delta_{i,j}^{(l)}$ containing all $\delta(l)$ calculated for each training example (i). Thus, the back-propagation algorithm consists in the following: That, in simple words, means that for every training set example, the loss calculated at the output layer is back propagated once layer at a time until the first hidden layer [**?**].
It is possible to mathematically demonstrate that $a_j^{(l)}\delta_i^{(l+1)}$ correspond to the partial derivatives calculated for the i-th training example. Hence, the term $\Delta_{i,j}^{(l)}$ can be considered as an accumulator of these partial derivatives. Then,

Algorithm back-propagation algorithm

1: **Training set** $\leftarrow (x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)})$
2: **Set** $\Delta_{i,j}^{(l)} = 0$ (for all l,i,j)
3: **for** $i = 1$ `to m` **do**
4: **Set** $a^{(1)} = x^{(i)}$
5: **Perform forward propagation to compute** $a^{(l)}$(for l = 2,3,..,L)
6: **Using** $y^{(i)}$, **compute** $\delta^{(L)} = a^{(L)} - y^{(i)}$
7: **Compute** $\delta^{(L-1)}, \delta^{(L-2)}, \ldots, \delta^{(2)}$
8: $\Delta_{i,j}^{(l)} = \Delta_{i,j}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$
9: **end for**

the partial derivatives calculated are:

$$\frac{\partial J}{\partial w} = D_{i,j}^{(l)} = \frac{1}{m} \Delta_{i,j}^{(l)} + \lambda W_{i,j}^{l} \tag{1.7}$$

(where λ is the regularization term) for j $\neq$ 0, and:

$$\frac{\partial J}{\partial b} = D_{i,j}^{(l)} = \frac{1}{m} \Delta_{i,j}^{(l)} \tag{1.8}$$

for j = 0. These partial derivatives are then used by the optimization algorithm chosen, for the cost function optimization.

The MLP feed-forward back-propagation Neural Network

Multi-layer feed-forward neural networks (MLP) are used for non-linear problems solving, that can not be solved with Linear Regression algorithms.

The MLP can be used in training and in prediction mode [51]: **training** is the process by which the MLP learns, by modifying its parameters, in order to reduce the errors obtained at each iteration over the **training set** data; **prediction** mode is the process of evaluating the resulting MLP performances over another dataset called **test set**. The prediction mode allows one to assess the generalization ability of the network: if the performances over the test set are similar to those obtained on the training set, then the generalization over new unseen data is good. On the other hand, if the performances on the test set are much lower than the training set ones, the MLP lacks of generalization: it suffers from **Overfitting**.

Avoiding overfitting

Overfitting means lack of generalization, and can occur for several reasons. However, the cause of overfitting is always the same: the hypothesis function becomes such detailed in describing the input data, that cannot describe a new set of unseen data.

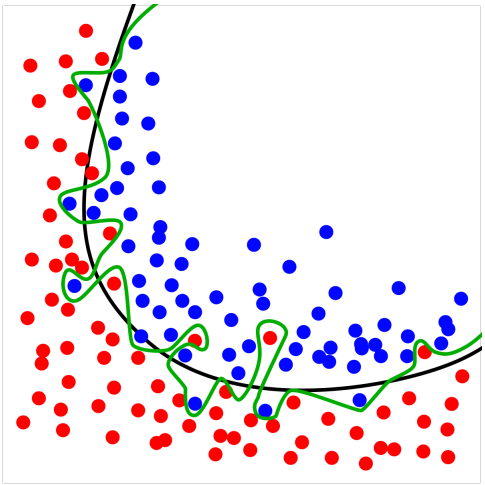

Figure 1.28: Overfitting example: the black curve represent a good generalization of data, while the green one represents an Overfitting scenario [58]

A very useful method to assess overfitting is the initial dataset division into three partitions: Training set, validation set and test set. While training set comprehend the majority portion of the dataset and is used for the actual training of the network, the remaining two portions are a much smaller and are used for overfitting verification. The substantial difference between the validation and test set, is that:

- overfitting is firstly evaluated *during the training* over the validation set. Epoch after epoch, the performances difference between validation and training set are evaluated A big difference is an indicator of overfitting: the learning process in this way can be stopped before this difference becomes too big.

- Afterwards, overfitting is evaluated also on the test set. Let us say that after the training process, the generalization over the validation has been proved good. In this case, there is the need to demonstrate

that the neural network has not overfitted over the validation set as well [8]. In order to verify this, a new unseen dataset, that is the test set, is introduced for the last prediction test. If the performances over the test set are comparable to the training set ones, then the generalization to new data has been proved to be good.

The most common partitioning of the original dataset are either 80% training set, 10% validation set and 10% test set or 70% training set, 15% validation set and 15% test set.

1.3 State of the Art

Preface: invasive and non invasive methods

The technique that offer the most reliable pressure readings requires an invasive intra-arterial set up. This approach is considered the reference technique. However, because an invasive set up is not suitable for clinical practice, non invasive methods are the most common in both ambulatory measurements and for continuous monitoring [36]. The classical approach for non-invasive pressure measuring before the 21st century has been the *auscultation or Korotkoff technique*. Auscultation technique consist in an observer that listens to the stethoscope while watching a sphygmomanometer. The sphygmomanometer is a pressure measurement device composed of a cuff that is wrapped around the patient's arm causing the brachial artery to occlude. The cuff is then gradually deflated so that the blood can flow again and start to produce the Korotkoff sounds. With a stethoscope placed over the brachial artery is possible to hear these sounds and associate them with the concurrent pressure value seen in the sphygmomanometer. However, this approach accuracy strongly relies on clinical staff preparation (it is user-dependent) and is easily compromised by many factors, thus resulting not accurate [36, 39]. For this reason, automated or semi-automated devices that are based on the *oscillometry* technique, have been introduced in clinical practice during the last two decades. Oscillometry technique is widely used in clinic, ambulatory and at home by the patient in holter mode [39]. The readings rely on the oscillation amplitude measured on the lateral walls of upper arm. The Mean arterial blood pressure is identified as the cuff pressure value when the oscillation amplitude is at its maximum. Thus, SP and DP are calculated starting from the MLP value, applying some fixed ratios [10]. Semi-automated devices acquire only one pressure value for each single activation, while automated devices are able to acquire several pressure values separated by a rest period with a single activation [36]. The techniques cited before have been validated and commercialized devices exploit them, but they suffer of many limits. The main limit is that the monitoring is not continuous, but requires a recovery time from 3 (sphygmomanometer) to 20 minutes (oscillometric devices) [39]. Moreover, although the accurate measures, when the cuff inflation results very uncomfortable, especially during the night time monitoring [39].

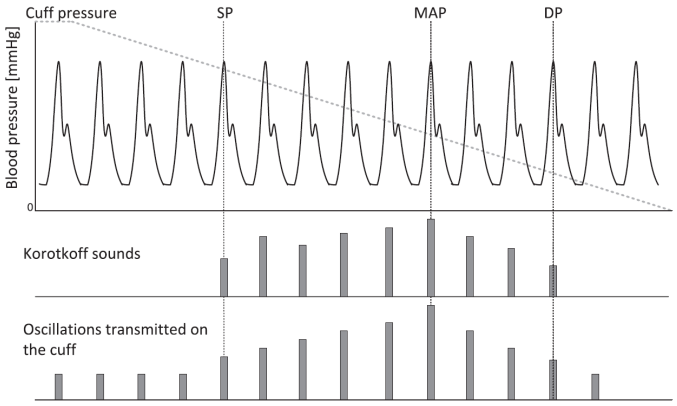

Figure 1.29: Diagram that show the methods of pressure values defining via auscultatory and oscillometric methods [39]

1.3.1 Continuous pressure monitoring systems: Cuff based methods

During the last two decades many new methods have been proposed for non invasive continuous monitoring of blood pressure, that hypothetically overcome the previous techniques limits.

Tonometry

is a more adapt technique to continuous pressure monitoring, because does not occlude the arteries and offer beat-to-beat pressure measurements. Although it does not use a cuff, a device that push a superficial artery towards a bone is needed. The pushing strength should be low, because the artery should be not occluded. In this way the device can be hold constantly without ischaemic damage. Meanwhile an embedded force sensor measures the pressure at contact. Because the partial occlusion is maintained during the entire cardiac cycle, the blood pressure profile is obtained 1.30. The accuracy of this technique is high only if its placement is continuously verified by an expert, because misplacement of millimetres produce high errors. Moreover, it is suitable only at rest conditions because it suffers significantly from motion artefacts [39].

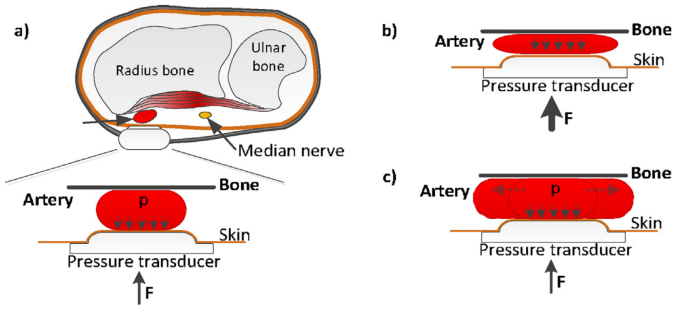

Figure 1.30: Diagram that show the methodology of tonometry technique [39]

Volume clamp method

Some modern optical based technologies such as *Finapres* (1990s) have been developed basing on the volume clamp method or Penáz technique, first described in 1973. Penáz technique offers beat-to-beat pressure measurement, in which the finger peripheral vessels are *unloaded* through a small finger cuff. Unloading the vessels means keeping the blood volume constant by the use of a Photoplethysmography (contained into the cuff) for blood volume estimation, within a feedback loop. Finapres refine the technique considering that if the volume under the finger is constant, then the arterial pressure equals the cuff pressure. Hence, by knowing arterial pressure, it is possible to reconstruct the brachial artery pressure through an algorithm [7]. Thus, Finapres monitoring is suitable for continuous pressure monitoring, but it still result uncomfortable.

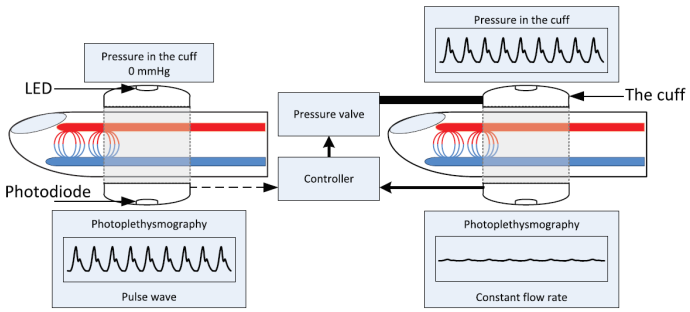

Figure 1.31: Diagram that show the methodology of volume clamp technique [39]

1.3.2 The modern trend: cuff-less methods

The modern trend is, however, the creation of cuff-less pressure continuous monitoring systems.

Pulse wave velocity It is possible to calculate accurate pressure values starting from the pressure wave velocity (PWV) throughout the arterial tree. In fact, the arterial blood pressure increase with the PWV increase. However, this method is suitable only for central elastic arteries, while for other arteries the accuracy is lowered. A way to make the measurement process easier is to measure the PWV at two substitute sites along the same peripheral artery as close to the aorta as possible. The best sites for non-invasive measuring are the first arteries, which begin at the aorta, that is the carotid and the femoral artery [13, 39].

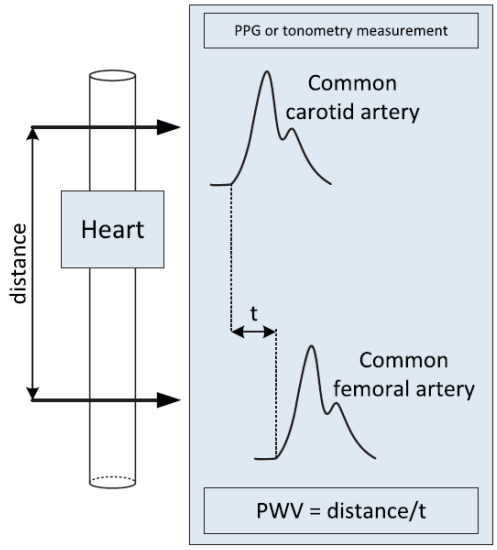

Figure 1.32: Graphical explanation of the PWV measuring technique [39]

By knowing the exact distance between the aorta and the measure site, it is possible to calculate the pulse transit time (PTT) [39]. The PWV is then calculated as:

$$PWV = distance/PTT \tag{1.9}$$

The PTT can be defined as the time difference between the occurrence of the ECG R-wave and the pulse appearance at the artery detection site. The pulse detection at periphery can be done through a Photoplethysmograph PTT is inversely proportional to SP, while DP and MBP are not easily deducible [7]. This method required an experienced technician and is user dependent. It is hence subject to many errors [39].

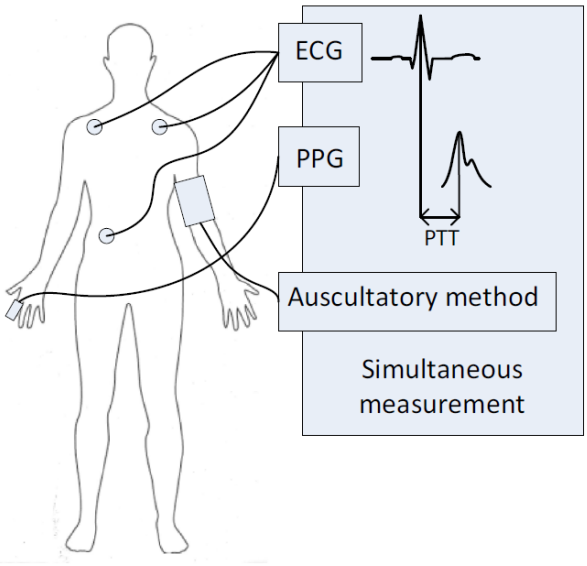

Figure 1.33: Graphical explanation of the PTT derivation technique [39]

These two techniques exploit the use of cuff-less sensors, but more than sensors are needed, making the whole device bulky and not comfortable. The last developments are moving towards the use of the Photoplethysmographic sensor alone. The reason is that the use of only one tiny sensor such as the PPG one, allow for may wearable applications.

Wearable devices

Photoplethysmographic sensor wearable devices are being investigated lately for non-invasive continuous blood pressure monitoring. Some studies have tried the development of some equations in order to calculate BP values from PPG waveform analysis [54, 45, 43, 50]. It is known that the ABP and PPG waveform are similar and that the physiological principles of the signal source are similar. However, although this is a perceptible relationship, it is very hard to characterize. In some studies, has been shown a linear relation between BP with cardiac cycle duration detected through PPG. It seems that a higher BP correspond to a quicker cardiac period [54]. However, parameters such as Systolic upstroke time, Diastolic and width of 2/3 and 1/2 pulse amplitude were considered separately for deeper studies. It has been shown that the Diastolic time is the most correlated PPG morphology parameter

with the BP, but that this correlation is not always linear. Indeed, people with same Diastolic time can have different BP. Different studies provide their methods and coefficients for BP detection from PPG wave analysis, but they lack of generalization, resulting accurate only for the studied dataset. [29, 54, 11]. In order to look for the BP-PPG relationship, many newest studies exploit the computational power of the Artificial Neural Networks (ANN) nowadays, which appears to be the most efficient method for the purpose. The first study [29]in which ANNs were introduced used a multilayer feed-forward back-propagation ANN, with two output neurons for SP and DP estimation. After the architecture investigation trials, it was found that 2 hidden layers offered the best performance, with 35 neurons within the first and 20 neurons in the second.

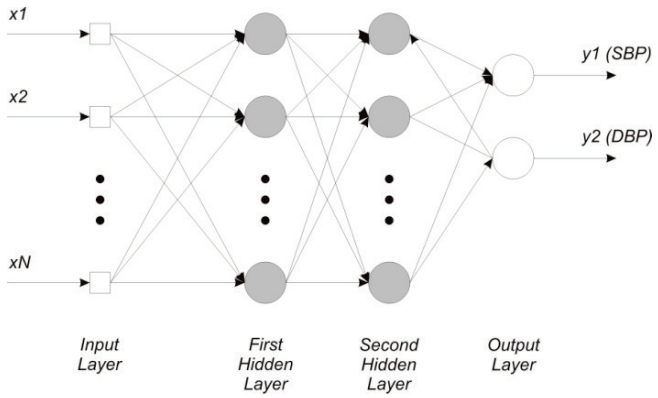

Figure 1.34: Multilayer perceptron for SP and DP estimation [29]

It has been seen that PPG amplitude is too compromised by motion artefact to be exploited as a feature for the ANNs inputs. Thus, such features were at first Systolic upstroke time (SUT), Diastolic time (DT), Cardiac period (CP), and the width of the PPG signal at 10%, 25%, 33%, 50%, 66% and 75% of the signal height. Moreover, other cross-features were extrapolated by combination of some of the previous (Figure 1.35).

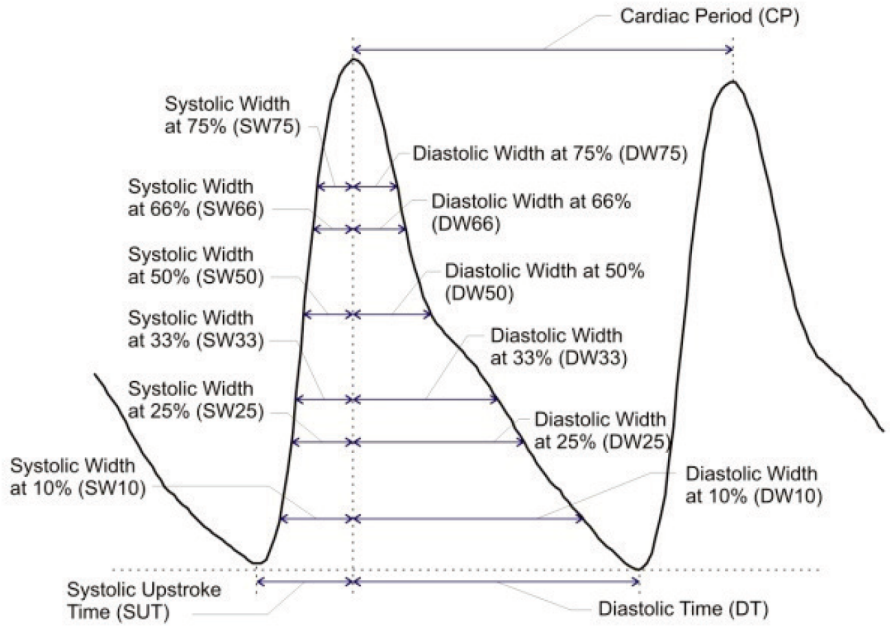

Figure 1.35: Calculated features [29]

In the study of Zhang and Wang [61], the previous work has been developed more in deep by performing a feature reduction on the 21 features previously selected, among which only 16 have been confirmed to be relevant for BP estimation. Moreover, because of the low prediction accuracy due to the random initialization of the NN, a genetic algorithm (GA) has been implemented to optimize the initial coefficients of the NN and hence obtain more accurate results.

Recently, very encouraging results were obtained [40] by using varied (modified) temporal periods of PPG waveforms as features for ANN training. Indeed the parameters used (SUT, DT, CP, R-PTT) (Figure 1.36) are averaged over time in order to create the new features that will be given as inputs to the NN (only mean values are used).

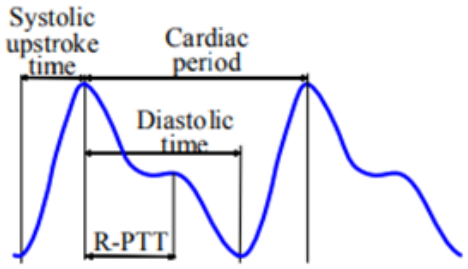

Figure 1.36: Features used within the study [40]

A newer study is presented starting from the consideration that the previous works suffer from long time accuracy decay since they do not take into consideration BP modelling over time [48]. This work considered different temporal acquisitions (1st day, 2nd day, 3rd day, 4th day and 6th months after the first recording) and estimated BP using a deep recurrent neural network consisting of multilayered Long Short-Term Memory networks. This method is shown to be the most effective in literature so far, surpassing the accuracy of all the previous BP prediction ANNs methods.

Chapter 2

Materials and methods

The study has been carried on at the St Microelectronics s.r.l., within the *Remote monitoring* group, belonging to the ST Reasearch and Development division. The group deals with the design of telemedicine wearable devices for biomedical purposes and it is specialized in Heart monitoring wearable devices such as *Bio2Bit NewMove*. These devices are tiny and portable and have been designed in order to continuously monitor the patient health status. The *Bio2Bit NewMove* has not been used inside this thesis project, but it is involved since the future applications include the created ANN otimization in order to be implemented into this device.
The chosen ANN to be implanted was a Multilayer feedforward backpropagation perceptron (MLP). This MLP is intended to perform a supervised multiclass classification task. The classes representing the different pressure ranges chosen were 7: [80 -100], [100-109], [110-119], [120-129], [130-139],[140-149] and [150-170]. Those classes are referred to with the central value of the range, i.e. with the labels 90, 105, 115, 125, 135, 145, 160 repectively.
The training dataset needed for the supervised learning task was a labeled dataset. Each row of the dataset described 15 morphological features of the single PPG period and a SP value.
The dataset was created from online freely available data containing ABP and PPG signals. The ABP signal was filtered and preprocessed in order to obtain a precise SP value for each PPG period. The PPG signal was preprocessed and segmented into periods in order to create a dataset containing an only PPG period in each row. From each segmented period were subsequently extracted the 15 features that represented the morphology of each PPG period and placed into the final training dataset together with the correspondent SP value.
The ANN was created with Keras, a python environment library for easy ANNs prototyping [8]. Moreover, for a faster convergence of the algorithm, the ANN was trained on Google Colab online available notebooks. These are remote hardware notebooks available from a web based platform, which provide free use of GPUs for AI reasarch.

2.1 Dataset creation

2.1.1 Data collection

The MIT Lab for Computational Physiology *Physionet MIMIC III Dataset* [21] has been the source of Photopletysmography and Arterial blood pressure data. The physionet website offers the possibility to get access to the anonimous data in a free way. Several types of signals are acquired simultaneously from intensive care unit patients, such as PPG, ECG, ABP, annotations about diseases and pathologies or events (e.g. arrhytmias or apnea). Each patient recording contains several hours or days of acquisition. Furthermore this dataset ha been chosen as data source because of the huge amount of available data and because, discarded the motion artefacts, many segments of acquisition show a very good quality.

Data extraction through Physionet wfdb tool The Physionet *wfdb* is a tool that allows the user to set some research filters in order to get a list containing only records with the desired characteristics. The desired characteristics can be the signal of interest, the desired anomalies/pathologies on the signal, specifical event-realted annotations and many more. In our case, a list of the patient records containing at least PPG and ABP was needed.

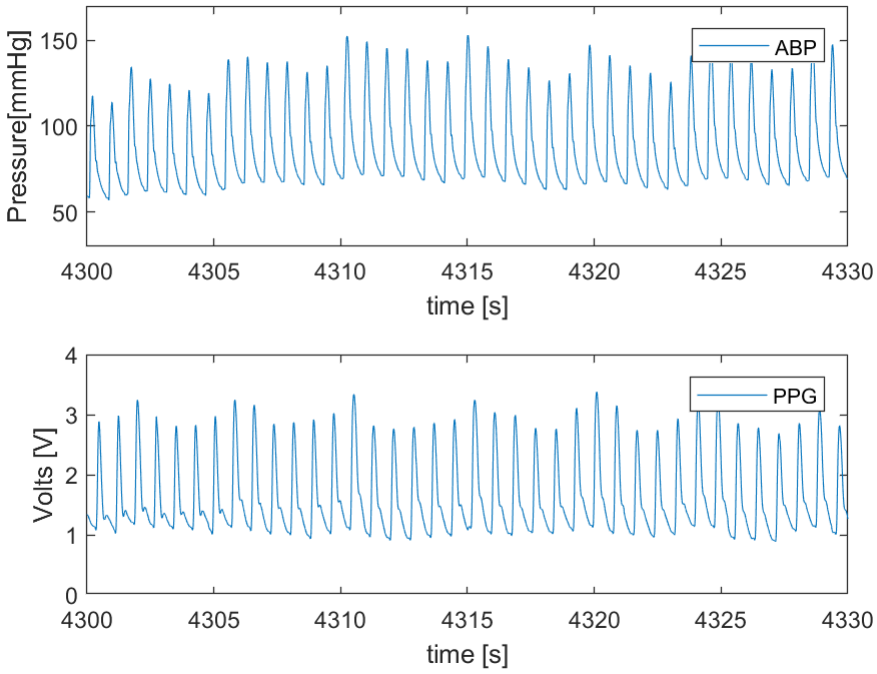

Figure 2.1: Example of raw concurrent PPG and ABP signals extracted

Another intention was to select only the records without artefacts, which required the visualization of the signals before downloading. The visualization of these signals was a long process because, given the long duration of the signals (even 6 days of cointinuous recording), the same record could contain several bad quality segments, that should have been discarded before downloading in order to avoid storage problems. Thanks to *wfdb* tool, these signals could be plotted and analyzed before downloading. The data extraction algorithm was performed on Google Colaboratory notebooks. For each record/patient, a new folder was created and the record informations were saved into it.

Algorithm Data extraction algorithm

```
Import selected Patient list
for patient in Patient list do:
    divide patient record in p Segments
    for segment in Segments do:
        visualize 10 random windows of 1 minute duration
        assign a good/bad label to the segment
    end for
    Dowload segments labelized as good
    Save information about record and segment intervals downloaded
end for
```

Dataset Eight patients simultaneous PPG and ABP records have been extracted. The aim was to create a Dataset containing a set of given features calculated for each PPG signal and to associate them with a unique SP value as shown in Figure 2.2.

Input #	Features															Targets
	CP	SUT	DT	SW10	SW25	SW33	SW50	SW66	SW75	DW10	DW25	DW33	DW50	DW66	DW75	SP
1																
2																
...																
n																

Figure 2.2: Template of the desired final Dataset

The number of inputs n correspond to the total number of PPG periods extracted from the recordings stored: in this project case n it was 124616. In Figure 2.2, the green columns names correspond to the features calculted over each PPG period. The features are inspired from the work [29] (graphically represented in the Figure 1.35) and are:

- **CP**: Cardiac Period
- **SUT**: Systolic Upstroke time
- **DT**: Diastolic time
- **SW**(x): width Δt (in seconds) between the pulse wave systolic peak PWSP and the time by which the x% of the systole is reached (e.g. SW10: Δt between the end of the systole and the 10% of the systole amplitude)
- **DW**(x): width Δt (in seconds) between the PWSP and the time by which the x% amplitude of the diastole is reached (e.g. DW75: Δt between the beginning of the diastole and the 75% of the diastole)

The target **SP** is the systolic pressure corresponding to the same row PPG period.

2.2 Implemented algorithm

2.2.1 Signal preprocessing

The signal preprocessing has been carried out on *Matlab 2017R*.
The first part of the preprocessing consisted in the downloaded signals visualization. The matlab visualization was a lot quicker and fluider than the previous wfdb one, because the plot window could be browsed and zoomed in order to identify compromized portions of the signals. If either ABP or PPG contained some compromized time segments, they were cut and deleted from both PPG and ABP signals, for consistency. In this way some information was lost, but the loss was not significant given the huge quantity of PPG periods available within each record. Nevertheless, the temporal subsequence of PPG periods was not an important factor for our Neural Network algorithm since the algorithm analyzes the single PPG period and not the temporal series.

ABP signal preprocessing

The aim of the ABP signal preprocessing was to remove the high frequency noise from the signal and to calculate the *Systolic wave*. The *Systolic wave* is a signal obtained from the envelope of the systolic peaks detected on the ABP signal.
First, the ABP was low-pass filtered at 6.6 Hz in order to remove the high frequency noise, while the low frequency components were kept in order to not alterate the SP and DP values.

Systolic pressure calculation algorithm

the algorithm for the SP calculation consisted in:

- a peak detection algorithm, which returned the SP value and the index of each peak: their sequence represent the *systolic pressure wave* or *SP wave* 2.5;
- a *resampling* of the *SP wave* at 125 Hz;
- a *smoothening* of the *SP wave*, based on a moving average FIR filter. This was done in order to avoid a too high beat-to-beat pressure variability.

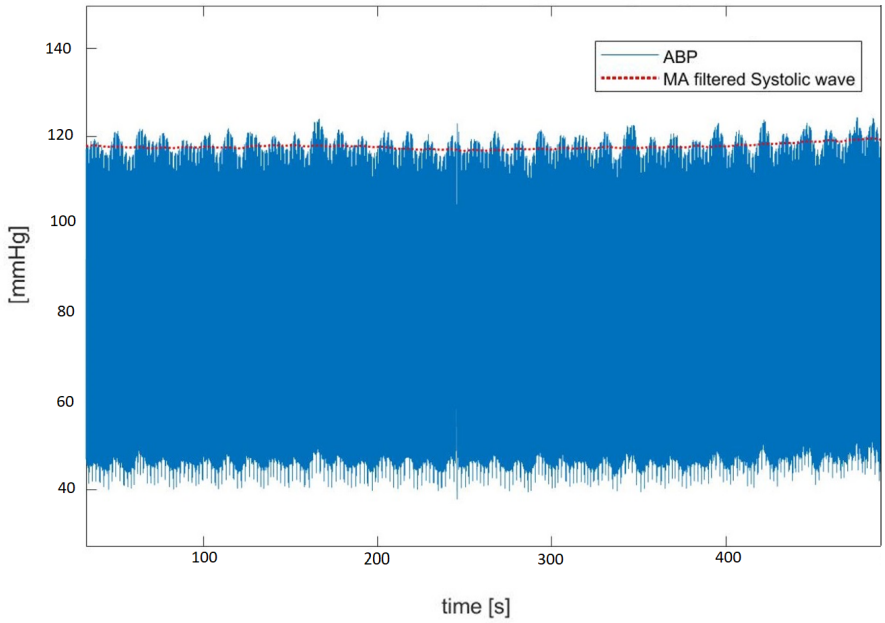

Figure 2.3: *SP wave* plot over the ABP waveform

Then, in a final step after the PPG preprocessing, the SP wave values were averaged over the correspondent PPG-period temporal window in order to obtain only one SP value. In fact, for the classification purpose, only one SP value per PPG period was required, that would have been its classification target.

PPG preprocessing

The aim of the PPG signal processing was to clean and to segment the PPG signal into periods. The next step was in fact the creation of a matrix containing a PPG period per row. The advantage of the **segmentation** is that the PPG periods could undergo a preliminary step of segmentation quality verification before the calculation of the features. The period segmentation started from each PWB until the PWE. The PPG matrix had dimension (n, $maxLen$), where n was the total number of PPG periods extracted from the recordings and $maxLen$ was a fixed length at $2*$(*average PPG period length*) in order to avoid problems due to the length variability of the PPG periods

segmented. In order to fit into the matrix row, a zero padding tail was added to each PPG period shorter than *maxLen*.

Filtering The PPG signal was high-pass IIR filtered at 0.6 Hz for trend remotion and low-pass IIR filtered at 6.6 Hz for the high frequency noise remotion.

Segmentation Then, an algorithm of PPG segmenting was performed (see algorithm), which was divided into three parts:

1. PPG diastolic valleys and systolic peaks detection through the algorithm *double threshold peak detection with minmax alternation* (see 2.4);

2. detecting of PPG periods outliers by period length. All periods shorter than *threshold1* and all those longer than *threshold2*, were labeled as outliers and not considered in the next segmentation step. In fact, the former were usually portions of PPG periods, while the latter were usually composed of two PPG periods by mistake. The values of *threshold1* and *threshold2* were defined experimentally.

3. The PPG periods were cut basing on the remaining diastolic trough indexes. Furthermore, there was the need of keeping track of the corresponding ABP values to each PPG period. Hence, th systolic wave was cut into slices basing on the same diastolic trough indexes. Each SP slice values were averaged in order to obtain only one SP value for each PPG period. This SP value was saved into the *SP vector*, which was used to map the PPG period-to-SP value corrispondence.

Algorithm Dataset preprocessing

PPG and ABP signal load
PPG and ABP coherent compromised segment remotion

PPG preprocessing

High – pass IIR filter, cut – off frequency = 0.6 Hz ▷ detrending
Low – pass IIR filter, cut – off frequency = 6.6 Hz ▷ high frequency noise remotion

ABP preprocessing

Low – pass IIR filter, cut – off frequency = 6.6 Hz ▷ high frequency noise remotion

Systolic wave calculation

ABP peaks detection ▷ see block diagram in Figure 2.5
SP filtering FIR MA filter, order = 200 ▷ Systolic wave smoothening
SP transient and tail first order polyfit interpolation

PPG segmentation

PPG peaks detection ▷ see block diagram in Figure 2.4
PPG periods length analysis and labeling into *too short, average* or *too long*
PPG periods segmentation and allocating into a *PPG_periods matrix*
Creation of *SP_vector*, containing an SP average for each PPG period saved

The algorithm used for PPG peaks detection was designed in order to concurrently perform:

- the detection of minimum peaks (PWB/PWE) and maximum peaks (PWSP);
- the check of their alternance: in order to avoid the subsequent detection of two peaks of the same type (two minimum or two maximum peaks), a minimum could be detected only if the last peak detected was a maximum and viceversa. For this purpose were used the two flags Flag_max and Flag_min, e.g.when Flag_max was true, the Flag_min was automatically set to False and only a minimum could be detected.
- a check on the distance between two consecutive peaks of the same type, that should always be higher than the MaxMax_dist threshold;

- a check on the distance between two consecutive peaks of different type, that should always be higher than the MinMax_dist threshold;

For this reason the algorithm is called *Double threshold peak detection with min-max alternation block diagram* 2.4.

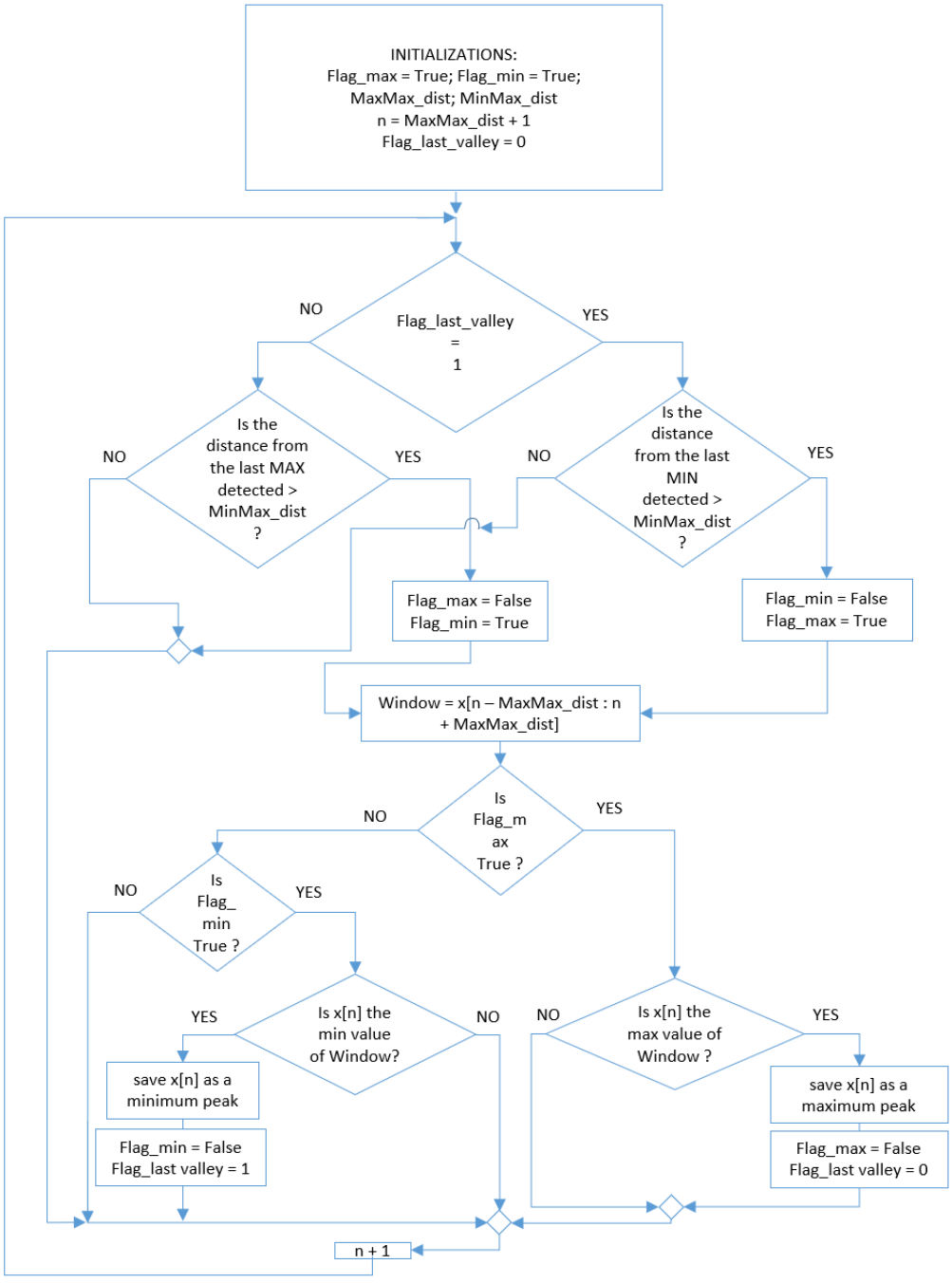

Figure 2.4: Double threshold peak detection with minmax alternation block diagram

The result of this algorithm was the detection of peaks quite far from each other. When there was the necessity to detect closer peaks or when the accuracy of peak detection was not as important as the PPG systolic peaks detection, the algorithm 2.5 was used. Hence, it was for:

- the correct detection of the PWB for the correction of the segmented PPG periods with an initial fluctuation;
- the ABP peaks detection: in fact, if a systolic peak was not detected on the ABP signal it was not as dangerous as not detecting a correct systolic peak on the PPG signal. This is due to the fact that the ABp systolic peaks compose the systolic wave, that is then filtered with a 200 order moving average FIR filter. For this reason, the missed detection of a peak in ABP signal is not a problem, in fact the error is then repaired with the FIR filtering.

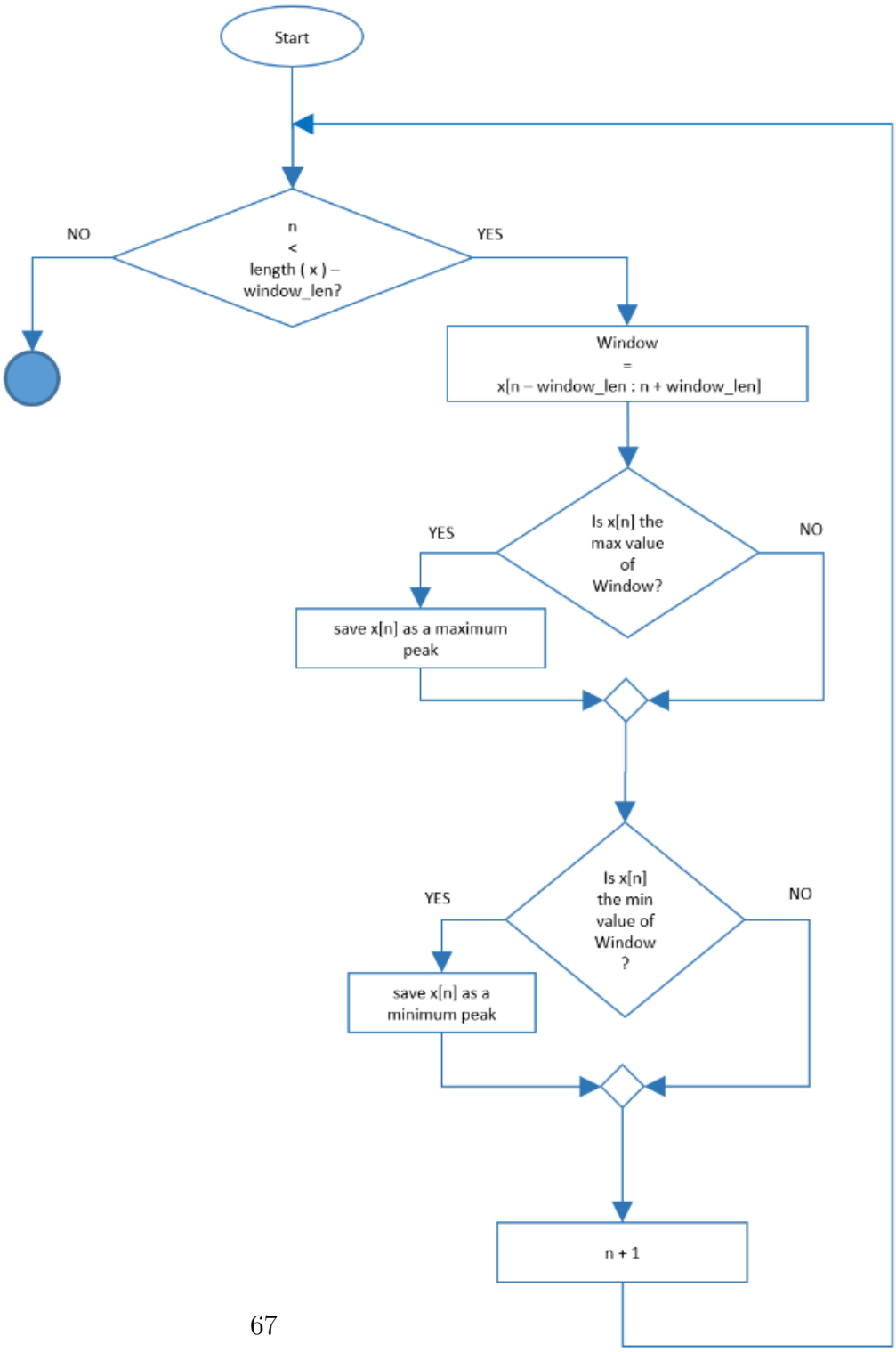

Figure 2.5: Peaks detection 2 algorithm block diagram

In order to check the status of the segmentation, all periods of the PPG periods matrix were plotted in the same figure. As we can see from the figure 2.6, there are many periods in which the peaks have been wrongly detected.

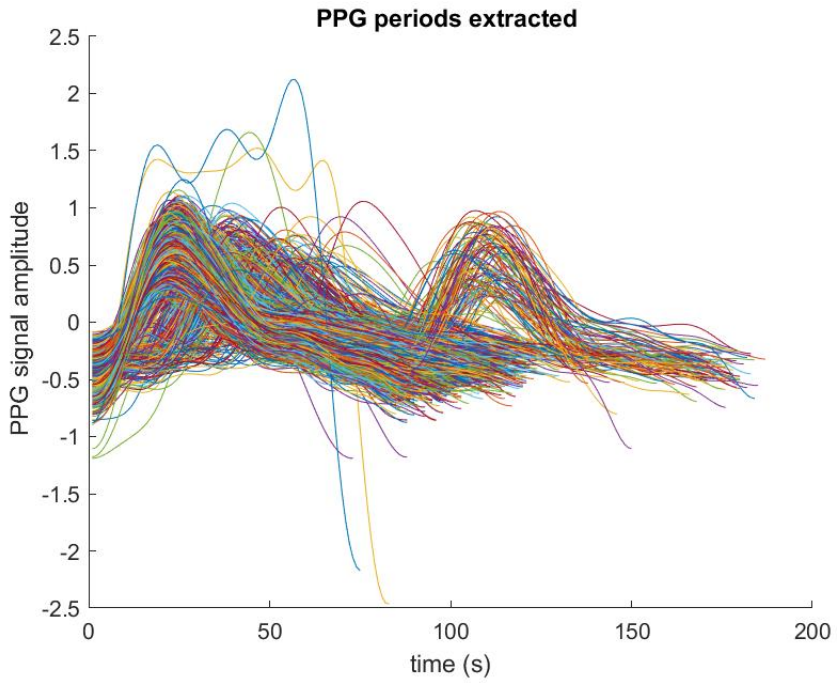

Figure 2.6: Plot of the PPG periods segmentation from one patient record

The most evident aspect is the presence of many segment mistakenly containing two PPG periods (the longest ones). This can be easily solved by deleting all the segments containing two periods. The second important thng is a short signal fluctuation at the beginning of some PPG periods. This fluctuation represents the diastolic tail of the previous PPG segment wrongly inserted before the beginning of the systole. This error can be deleted by detecting the real PWB right before the systolic upstroke start. After these correction, the plot of the same record PPG periods appears much cleaner as in figure 2.7.

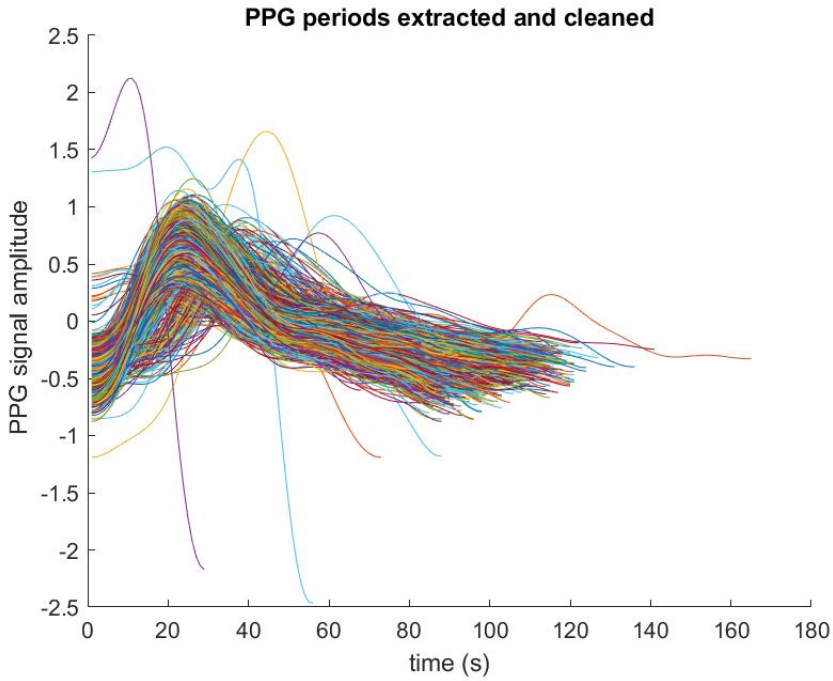

Figure 2.7: Plot of the PPG periods segmentation from one patient record after correction

Summarizing, the result of the segmentation is a matrix containing all these different PPG periods.

2.2.2 Feature extraction

Feature extraction is the process by which are calculated the parameters that will be the neural network inputs.

The features The algorithm used for the purpose is shown in the 2.8.

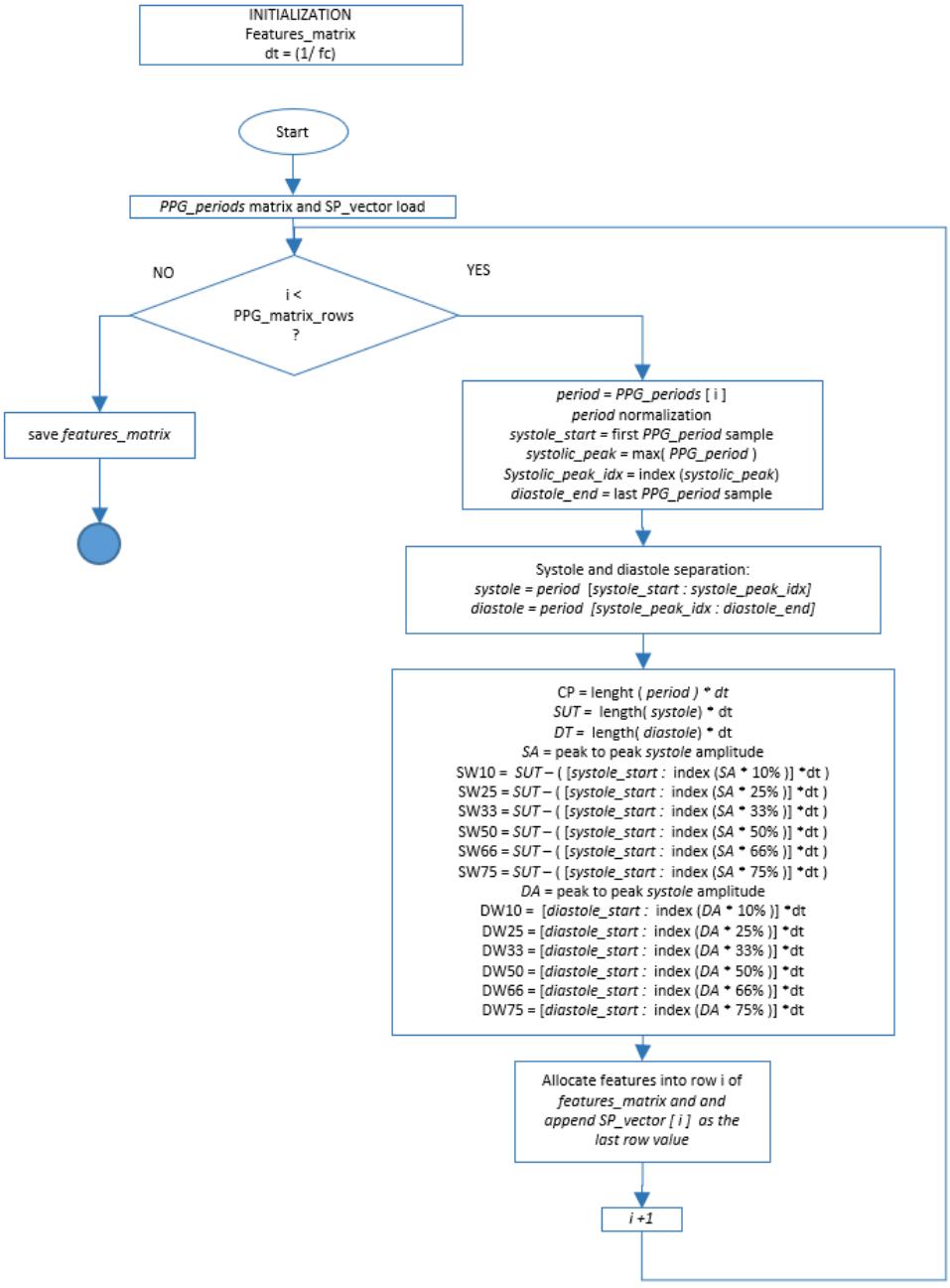

Figure 2.8: Features extraction algorithm block diagram

Features validation In order to verify the correctness of the features calculation, a graphical visualization has been done over the 10% of the dataset elements. In fact, the features represented temporal values, that in a plot, have to perfectly fit to the shape of the waveform. If they do not fit perfectly, they have been wongly calculated. The Figure 2.9 shows an example of visualization of the calculated features.

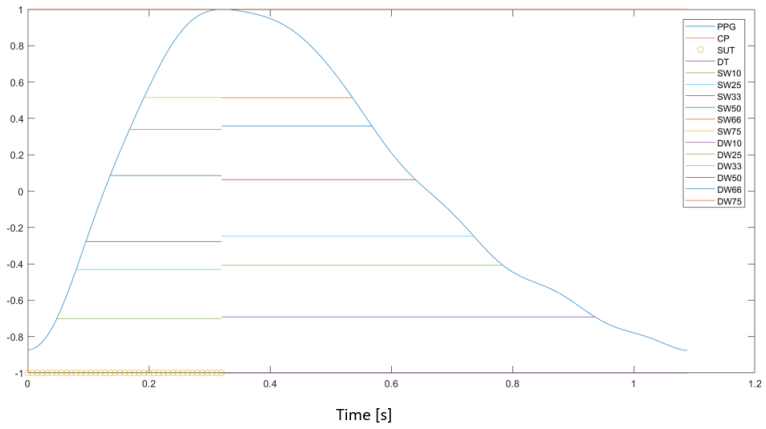

Figure 2.9: An example of visualization of the calculated features

2.2.3 Feature engineering

The feature engineering was carried on into the *Google Colab notebooks* in order to convert the training dataset into a python compatible format. This was necessary because the ANN algorithm would have been performed on *Keras*. For this purpose, the *features matrix* and the SP_vector values are integrated into a pandas DataFrame(see algorithm). In these type of python structures, each column is named after the feature name and the last column was named 'SP' and the contained the SP_vector values in it (an example is shown in Figure 2.10). In this way, a dataset as described in 2.2 was obtained. The SP_vector into the algorithm is called $targets$.

	CP	SUT	DT	SW10	SW25	SW33	SW50	SW66	SW75	DW10	DW25	DW33	DW50	DW66	DW75	SP
115715	0.155986	-0.358235	0.391261	-0.308635	-0.139064	-0.309932	-0.243140	-0.195040	-0.066859	0.367919	0.039355	-0.209536	-0.122339	-0.127661	0.029555	125.285509
65483	-0.013730	-0.213050	0.091716	-0.126715	-0.139064	-0.089069	-0.243140	-0.195040	-0.066859	-0.792035	-0.650773	-0.610692	-0.690708	-0.710467	-0.669661	121.375028
120835	-0.296592	-0.213050	-0.282716	-0.126715	-0.139064	-0.089069	-0.243140	-0.195040	-0.066859	0.875399	0.902014	0.993932	0.019754	0.066607	0.262627	123.486104
40572	1.230859	0.948432	1.140125	0.782886	0.689945	0.573519	0.533476	0.760223	0.679813	1.310382	1.592141	1.695956	1.582769	1.426486	1.427986	103.058307
37848	2.588593	3.706953	1.514557	3.693609	3.798727	3.665599	3.898813	3.944434	4.039839	1.817862	2.196003	2.397979	2.719507	2.592097	2.593346	125.358925
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
104003	-0.975459	-0.648606	-0.956694	-0.672475	-0.553568	-0.530795	-0.502012	-0.513461	-0.440196	-0.937029	-0.737039	-0.610692	-0.406523	-0.321930	-0.203517	120.604961
10742	-0.013730	-0.648606	0.316375	-0.490555	-0.553568	-0.530795	-0.502012	-0.513461	-0.440196	0.367919	-0.219443	-0.811270	-0.832800	-0.904735	-0.902733	131.666898
49689	1.004570	0.512876	1.065239	0.600966	0.482693	0.573519	0.533476	0.441802	0.306477	0.875399	0.815748	0.793354	0.446030	0.260876	0.262627	96.719482
58564	-1.145175	-0.358235	-1.331126	-0.308635	-0.346316	-0.309932	-0.243140	-0.195040	-0.440196	-0.284555	-0.133177	-0.209536	-0.264431	-0.127661	0.029555	137.892447
61615	-0.918886	-0.213050	-1.106467	-0.308635	-0.346316	-0.309932	-0.243140	-0.195040	-0.440196	-0.864532	-0.737039	-0.610692	-0.548616	-0.516198	-0.436589	120.090872

124616 rows × 16 columns

Figure 2.10: shuffled DataFrame example visualization

Normalization and Standardization

The obtained features are never given to the Neural Network in raw format, but they have to be either normalized or standardized. Because all of the features represent temporal values, for logical reason the initial idea was to normalize them would e within the range [0 - 1]. However, different Scalers have been tried in order to find the most suitable option in terms of accuracy performances. The *Scikit Learn* scalers were compared:

- **StandardScaler**: removes the mean and scales the standard deviation to the unit;
- **MinMaxScaler**: scales the features to the given range (various ranges were tried, such as [-1,1],[-3,3],[0,1]);

The best results (in terms of performances) were obtained with the use of StandardScaler, that was then chosen.

2.2.4 Dataset preprocessing

Dataset targets discretization The Dataset targets (SP values) initially contained a numeric value from 80 to 170. In order to implement a multiclass classification algorithm, these values had to be discretized, i.e. condensating all values within a range with a unique label. The choice of the ranges was difficult because the narrower the range, the more precise is the prediction, but it is also more difficult for the ANN to classify an input within the correct SP range. In this work, the ranges were chosen to be 10 mmHg and 20 mmHg

wide. The chosen ranges are described into Table 2.1, where each range is also associated to the relative class name. In this way each *targets* column

SP range	Class label
80 - 100	90
100 - 109	105
110 - 119	115
120 - 129	125
130 - 139	135
140 - 149	145
150 - 169	160

Table 2.1: Systolic pressure ranges division

value is converted into a label. In order to do this, two variables have to be created: the *bins* variable described the SP ranges borders in one vector (i.e. $bins$ = [80, 100, 110, ..., 150, 170]) and the *labels* variable contained the class labels as indicated in 2.1. For semplicity, a number respresenting the aritmetic average of the two range limits is taken as label for each range. The discretization algorithm discretized the target column values basing on the *bins* values and associated to each of them the correspondent label. Then the label was converted into a categorical value. The categorical values range from zero to Z, where Z is the number of classes ($classNum$) -1. Each categorical value is then converted into a binary vector (one-hot-encoded vector) containing Z -1 zeroes and a *one* in corrispondence of the belonging label (see table 2.2).

SP value	*label*	Categorical	One-hot-encoded
89,2	90	0	1000000
101,7	105	1	0100000
119,9	115	2	0010000
...			
165	160	6	0000001

Table 2.2: Example of targets discretization and one-hot-encoding

Dataset balancing Moreover, an analysis over the classes distribution was done for assessing the dataset balancing (Figure 2.11).

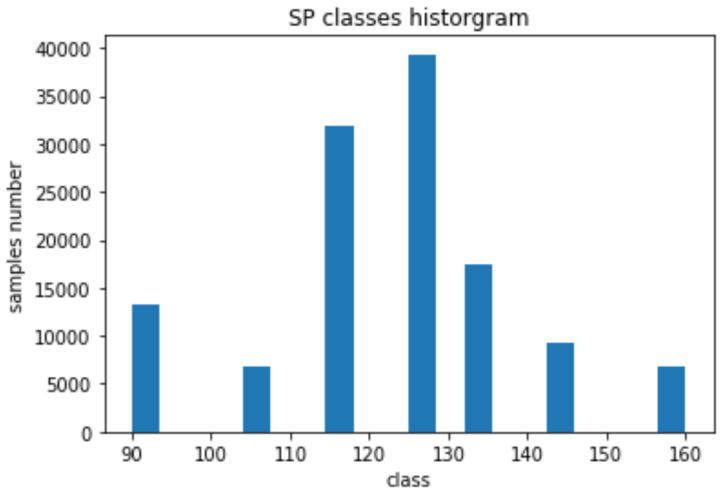

Figure 2.11: Dataset balancing analysis

Because of the evident unbalancing, an algorithm for Dataset balancing was performed. First of all, the desired number of examples per class was expressed through the $samplesNum$ variable. Then, the Dataset balancing algorithm performed an **Upsampling** of the classes which elements numerosity was less than the $samplesNum$ and a **Downsampling** of the classes with more than $samplesNum$ elements.

Different trials were made for the number of examples per class to be considered. The best choice resulted to be 20000 samples per class.

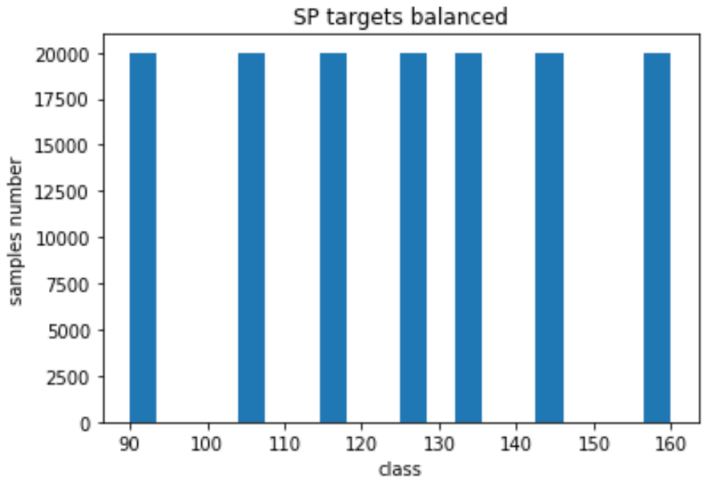

Figure 2.12: Result of dataset balancing

Dataset partitioning

In order to avoid overfitting, in order to avoid overfitting, the dataset was divided in training set (70%), validation set (15%) and test set (15%), resulting:

- 97966 training examples and targets;
- 20999 validation examples and targets;
- 21000 test examples an targets.

Algorithm Dataset preprocessing

Initialization

random *seed* declaring ▷ for reproducibility
import *features matrix* and *SP_vector*
targets ← *SP_vector*
bins ← [80 , 100 , 110 , 120 , 130 , 140 , 150 , 170] ▷ bins borders
labels ← [90 , 105 , 115 , 125 , 135 , 145 , 160] ▷ Classification labels
categorical ← $[0, 1, 2, .., 6]$

Create and shuffle dataset

procedure CREATE AND SHUFFLE DATASET($features, targets$)
df ← *features matrix* and *targets* ▷ df is a pandas DataFrame
shuffle *df* rows
end procedure

Discretization of target column

binned_targets ← *targets* discretized and labelized ▷ Each target value contains a *label*
procedure FROM LABEL TO ONE-HOT-ENCODING(*sets*)
categorical_targets ← converted *binned_targets* into categorical values
one_hot_encoded_targets ← converted *categorical_targets* into one-hot-encoded vector
df ['targets'] ← *one_hot_encoded_targets*
end procedure

Dataset balancing

samplesNum ← 20000 ▷ desired number of elements per each class
Balanced_ds ▷ New balanced DataFrame initialization
for *label* **in** *labels* **do** :
temp_df_indexes ← indexes of *df* for which $targets == label$
label_count ← *temp_df_indexes* rows number
if $label\_count < samplesNum$ **then**:
num ← *samplesNum* - *label_count*
procedure UPSAMPLING($num, temp\_df\_indexes$)
newElems ← choose *num* random rows into *temp_df_indexes*
temp_ds ← append *newElems*
end procedure

else if *label_count* > *samplesNum* **then**
num ← *samplesNum*
procedure DOWNSAMPLING(*num*, *temp_df_indexes*)
temp_ds ← choose *num* random rows into *temp_df_indexes*
end procedure
end if
Balanced_ds ← append *temp_ds*
end for
Shuffle *Balanced_ds*

Dataset division into Training, Validation and Test set

$N \leftarrow$ number of *Balanced_ds* rows
$train\_perc \leftarrow 70\%$ N
$valid\_perc \leftarrow 15\%$ N
$test\_perc \leftarrow 15\%$ N
shuffle *Balanced_ds*
$valid\_examples \leftarrow$ first $(train\_perc * N9$ elements of *Balanced_ds*
$test\_examples \leftarrow$ elements between $(train\_perc * N)$ and $((train\_perc + valid\_perc) * N)Balanced\_ds$ elements
$training\_examples \leftarrow$ last $(test\_perc * NBalanced\_ds)$ elements

2.2.5 Artificial Neural Network algorithm

The artificial Neural Network algorithm has been created on **Keras**, which is a Deeplearning API created primarily by Google for reducing the cognitive load [9]. Furthermore, for a faster ANN training, the algorithm was carried on exploiting the GPU computational power of **Google Colab** Notebooks. In this way, all the heavy computational load was traslated in a web based python environment, avoiding the overloading of local hardware resources. Moreover, the process could be parallelized in several notebooks in order to achieve a much faster training.

2.2.6 The model creation

The model is created as a Keras Sequential, because each layer can be added in a sequential way by the *add* function. The standard guidelines for the creation of a MLP for multiclass-classification were followed. The model was initialized as follows:

- the input layer had input dimension equal to the input features, that is 15;
- all the hidden layers were **Dense** layers, which are fully connected with the previous and following layers;
- all neurons of the hidden layers were initially characterized by a **sigmoid** activation function;
- the output layer was a **Dense** layer and was composed by a number of neurons correspondent to the number of classes; its activation function is called **softmax** and is necessary in multiclass classification problems because it associates a probability to each output layer neuron. The winner neuron can be only one and is the one with the highest probability.

To this model were added:

- a Dropout layer for Regularization after each hidden layer, initialized with *0.0 dropout rate*;
- a *normal* weights inizialization for each layer;

The initial neural network was named the *model zero* and was used for the parameters optimization: the hidden layer were two, of 120 and 240 neurons respectively.

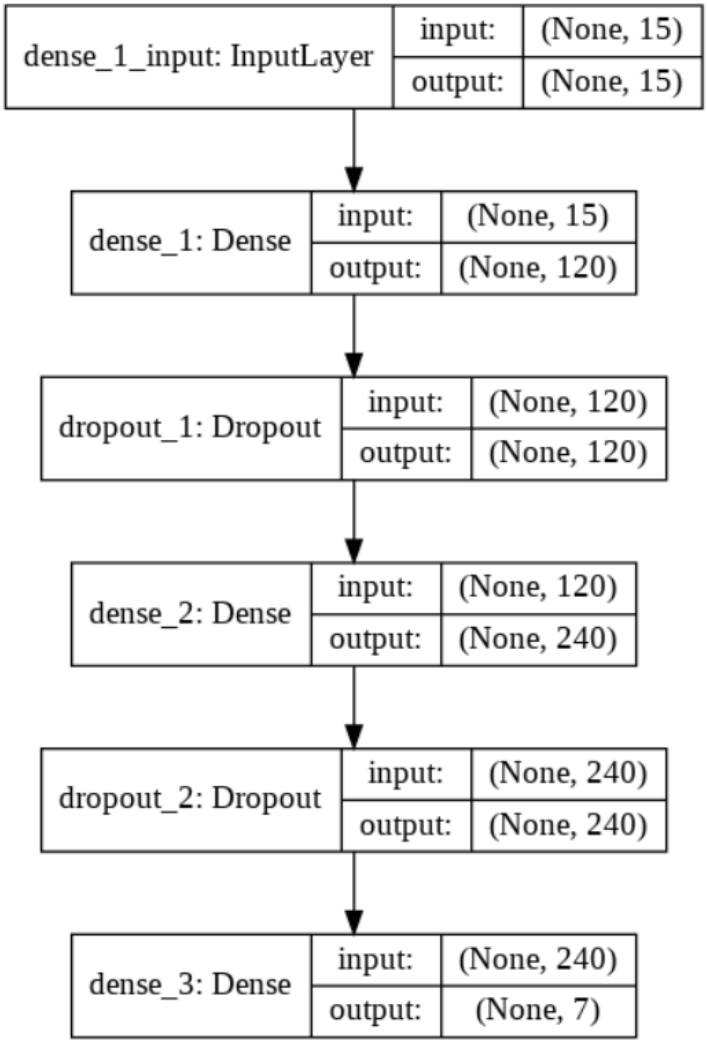

Figure 2.13: Keras MLP *model zero* schematic

The training was performed using the parameters:

- **Adam** algorithm as optimizer

- **categorical crossentropy** as loss function
- **categorical accuracy** as metric
- **Model Checkpoint** callback, that saved the model weights corresponding to the epoch of early stopping as *best model* of the training.

Other than the insertion of the Dropout layers, two **Early stopping** callbacks were inserted for avoiding overfitting that stopped the algorithm either when the validation data loss reached its minimum and when the validation data accuracy reached its maximum. Both the early stopping function had a toleration of 60 epochs.

2.2.7 Optimization of ANN parameters

For ANN optimization, many parameters have to be chosen by the programmer, that is:

- architecture: *depth* and *width*;
- hyperparameters: *epochs number, batch size, activation function of the single units, weight initialization mode, optimizer, learning rate* and *dropout rate*.

The parameters optimization consisted of two phases: the initial phase consisted in a **cross validation** over the hyperparameters, that was followed by a **manual optimization** of the architecture.

Cross validation

The cross validation served both as validation of the previous manual tuning and for investigating a set of combinations of new parameters that would have requested much more time if performed manually. In fact, the cross validation has been performed with the *GridSearchCv* function built in **SciKitlearn** library. This function performs a grid cross validation over the parameters that the user chooses. In this thesis were performed the following grid searches:

Parameters	Values
Activation function	relu, linear, tanh, sigmoid, hard sigmoid, softmax, elu, selu, softplus, softsign
Optimizer	sgd, rmsprop, Adagrad, Adadelta, Adam, Adamax, Nadam
Learning rate Optimizer	0.01, 0.03, 0.06, 0.1, 0.2, 0.3, 0.5 Adam, Nadam
Learning rate Optimizer	0.001, 0.002, 0.003, 0.004, 0.005, 0.006, 0.007, 0.008 Adam, Nadam
SGD learning rate SGD momentum	0.005, 0.01, 0.03, 0.1, 0.2, 0.3, 0.5 0.0, 0.2, 0.4
batch size	32, 64, 128, 256, 512, **1028**, 2056, 4112, 8224, 16448
Epochs	500, 750, 1000, 1250, 1500, 2000, 2500
Dropout rate	0.0, 0.1, 0.2, 0.4, 0.5, 0.6, 0.8, 0.9 0.02, 0.04, 0.06, 0.08
Initialization mode	uniform, normal, zero, lecun_uniform glorot_normal, glorot_uniform, he_normal, he_uniform
Activation function Initialization mode	sigmoid, hard sigmoid, softsign lecun_uniform, glorot normal, he uniform
Activation function Initialization mode	relu, linear, tanh, sigmoid uniform, normal, zero

Manual tuning

The manual tuning was performed by fixing all parameters except the one that was investigated. First of all, the architecture of the neural network was investigated, by varying the number of layers and their numerosity (see Figure 3.1, 3.2). Afterwards, batch size was optimized for the architecture defined.

One Hidden layer depth The use of only one layer allows the MLP to learn the linear relations between features and targets. This is a very efficient way to study those linear relations, but the use of only one layer does not suit for the project purpose, which aims at investigating the non - linear relation between the PPG morphology and the relative SP value. However, one layer optimization is needed to extract as many linear relations as possible because

they help to improve the next layers performances.

Two Hidden layer depth It can be demonstrated that implementing a MLP with two layers of appropriate width, every non-linear relation can be approximated. The first layer width that guarantees the best performances has been fixed in order to proceed with the second layer width optimization. It can be seen that by incresing the second layer width, the MLP accuracy continues to rise, while the loss continues to be reduced.

Higher depths The cases where the performances improve with a number of layer larger than 2 are sporadic. The exploration of higher depths is done through the same method mentioned above: firstly, when the optimum width of a layer is found, it is fixed and the following layer width can be explored. As it can be seen from the results, increasing the depth to more than 2 layers does not correspond to a significant increase of accuracy.

Algorithm Artificial Neural Network algorithm for training, saving and evaluation over *test set*

Model zero Initialization

$Learning\_rate = 0.01$
$batch\_size = 1024$
$epoch\_num = 1500$
$inputNum = 15$ ▷ number of input features
$classNum = 7$ ▷ number of output classes
$hidden\_layers = [120, 240]$
$sets = training\_set, validation\_set, test\_set$
$myOpt \leftarrow$ Adam ▷ optimizer choice
$myLoss \leftarrow$ Categorical crossentropy ▷ Loss function choice
$myMetric \leftarrow$ Categorical accuracy ▷ metric choice
$IL\_Activ\_Func \leftarrow$ sigmoid function ▷ Input layer activation function
$HL\_Activ\_Func \leftarrow$ sigmoid function ▷ Hidden layers activation function
$O\_Activ\_Func \leftarrow$ softmax function ▷ Output layer activation function
$Drop\_rate = 0.0$
$Init\_mode \leftarrow$ normal

Model creation

$Model \leftarrow Sequential$ ▷ Keras Sequential model
define *input_layer* with $inputNum$ neurons and **sigmoid** activation function
for *hidden_layer* **in** *hidden_layers* **do**
$Model \leftarrow$ **add Dense** *hidden_layer* with **HL_Activ_Func** activation function and $Init\_mode$ initializer
$Model \leftarrow$ **add Dropout** with Dropout rate $Drop\_rate$ ▷ Regularization
end for
define *output_layer* with **softmax** activation function $outputNum$
$Model$ **compilation** $(myOpt, myLoss, myMetric)$

Training

$Model$ **training for** *epoch_num*
$performances\_dict \leftarrow$ training set $(min(loss), max(accuracy))$
$performances\_dict \leftarrow$ validation set $(min(loss), max(accuracy))$
save $Modelweights$

Overfitting validation

prediction$(test\_examples, test\_targets))$
$performances\_dict \leftarrow$ test set $(loss, accuracy)$
$predicted\_classes \leftarrow Model$ predictions over the $test\_examples$

2.2.8 The performances calculation

At the end of the 2.2.7 algorithm, the final model is considered as the model saved by the Model checkpoint and named *best_model*. The performances are calculated over the *best_model*. The evaluated performances metrics, beyond the categorical accuracy already estimated on the test set, are the Precision, the Recall and the F1-score (see 2.2.8).

Algorithm Performance metrics calculation

$CM \leftarrow$ **confusion_matrix**$(predicted\_classes)$
$TP \leftarrow\ diag(CM)$ ▷ True positive
$FP \leftarrow\ sum(columns) - TP$ ▷ False positive
$FN \leftarrow\ sum(rows) - TP$ ▷ False negative

$Precision = \frac{TP}{TP+FP}$

$Recall = \frac{TP}{TP+FN}$

$F1 - score = 2* \frac{Precision*Recall}{Precision+Recall}$

First of all, it is necessary introduce the concept of true positive, false positive and false negative.

The **True positive** (TP)are highlighted in blue within the confusion matrix Figure 2.14 and correspond to the *test_examples* being correctly classified. The **False Poisitives** (FP), for each class, are the elements wrongly predicted as belonging to the current class: they are represented by the sum of the column elements, except the diagonal ones. The **False Negatives** (FN) of each class are the elements belonging to the currently analyzed class, that were predicted as belonging to another class. They are represented by the sum of the row elements, except the diagonal ones.

The **Precision** is inversely proportional to the false positive elements classified by the model. In fact, a low precision for a specific class x (let us say 10%) indicates that the model often classifies the elements within the x class even if they do not belong to it. On the other hand, a 95% precision ratio illustrates the great model attitude to predict the x class element correctly. A high precision reduces the number of false positives.

The **Recall** is inversely proportional to the elements classified as false negative by the model. A low Recall means that the number of false negatives

is high, while a low Recall means that the number of false negatives is low.

The **F1-score** is a weighted harmonic sum between precision and recall. It is used for a first understanding of the model performances: indeed it assigns the same importance to both Precision and Recall. On the contrary, the importance given to precision and recall should be problem-related. In this project,it was used to compare different architectures of MLP, in order to find the optimum.

For a better understanding, let us focus on the class '90'(see Figure 2.14). The TP are 2368. The FP are $416 + 54 + 23 + 29 + 20 + 8 = 550$. The FN are $469 + 48 + 19 + 41 + 32 + 25 = 634$. The precision would be $\frac{2368}{2368+550} = 0,811$ and the Recall $\frac{2368}{2368+634} = 0,788$.

		Predicted classes						
		90	**105**	**115**	**125**	**135**	**145**	**160**
Ground truth classes	**90**	2368	469	48	19	41	32	25
	105	416	2371	110	20	85	23	12
	115	54	210	1900	532	210	102	12
	125	23	43	660	1609	460	219	11
	135	29	62	165	337	1911	340	28
	145	20	24	93	82	262	2164	321
	160	8	14	10	4	9	237	2795

Figure 2.14: Example of confusion matrix

Importance given to Precision and Recall It is known that the weight given to Precision and Recall in the performances evaluation is problem related. In medical and clinical applications, a low FN ratio is often much more important than the FP ratio, especially for the classes that indicate a patology. In this project, 135, 145 and 160 are the SP classes that represent a pathological condition (see table 1.2). For this reason it is very important to have a very high Recall for those classes, i.e. low number of FN . On the

other hand, too many false alarms would lead to a lack of reliability in the monitoring device: for this reason, a high Precision, that is a low number of FP, for these three classes is needed as well. Summarizing, the most important value to be evaluated in terms of performances of a clinical device is the Recall, that should be high enough for the classes indicating hypertension. However, high Precision is requested too in order to lower the number of false alarms. Precision and high recall together are the best solution for the classes indicating high pressure or hypertension.

For this reason, the whole optimization of the MLP parameters was done firstly by assessing the accuracy, in order to have an idea of the general quality of the classification averaged over all classes. Then Recall and Precision were evalued separately for all classes, but the decisions about the final parameters choice were done in order to maximize the hypertension classes Recall.

2.2.9 Computational cost and memory load of the model

In order to choose the final model, it is very important to take under control the computational cost of the model in terms of memory storage needed. In fact, in order to execute the inference of an ANN algorithm on a MCU, it is necessary to store the *weights, activation,* the *input data* and the*output data*, usually in *32 bit floating point* format that requires 4 Bytes. For the computational cost analysis of the final model, the **STM32CUBE.AI** toolkit from STMicroelectronics was used. This toolkit is capable of interoperating with the commonest Artificial Intelligence libraries (such as Keras, Tensorflow, Caffe, Lasagne and ConvNetJs) in order to convert any pre-trained ANN in a C-language format, ready to be written onto the STMicroelectronics (MCUs)[47].

This tool is able to.

- provide informations about the CPU load (multiply and accumulate macc operations) and memory (RAM and ROM) required in order to embed the ANN into a MCU;
- show a list of STMicroelectronics MCUs containing the necessary memory to both store the necessary parameters and support the calculus;
- perform a layer-per-layer analysis of the CPU load and memory requirements;

- Compress the weights from a 32-bit floating point format to 8-bit quantized: this allows both saving of flash memory and in some cases the reduction of the computational cost, since operating with 8-bit format requires less operations;
- compare the compiled model with the original one in terms of accuracy and time required for the single inference;

The flash memory (ROM) of the STMicroelectronics MCUs vary from a few dozens of kB to 1024 or 2048 kB, while the ram varies from a few dozens of kB to hundreds of kB. The MCU used within the *Remote monitoring group* devices is the **STM32L4R7** and contains:

- a **ROM** of **2048 kB**
- a **RAM** of **640 kB**

In order to say that the ANN is embeddable, the flash and RAM memory it requires must be lower than the STM32L4R7 available ones.

The weights, which are the most memory requiring part of the ANN, are stored into the ROM, while the input values, the output values and the activations are stored into the RAM. After loading, the Keras model was compiled in C and different compression of the weights were evaluated: they can be compressed by 4 or 8 times, basing on the memory saving necessities. In case the model is compressed, a validation cross-correlation analysis is performed between the keras and the C-compiled model performances, to check if the accuracy of the latter was compromised. Both models inferences are done on random data and the output classification of the original model is taken as the ground truth for the C-compiled one. If the Validation results in terms of accuracy are low, it does not always mean that the implemented model performances have decreased, yet a further analysis on the network is required to check the accuracy.

The **inference time** is very important to understand if it can be done in real-time. In fact, being the purpose of the project to create a real-time and beat-to-beat pressure monitoring system, the inference time should be inferior to the physiological duration of a cardiac cycle. The Heart rate at rest can vary from 60 to 100 bpm. This means that the inference time should be lower than 0.6 seconds in order to evaluate each PPG period. If the inference time were higher, it could still be done in real-time, but it would not be a beat-to-beat evaluation because some PPG periods would be lost.

Chapter 3

Results

3.1 Parameters optimization

For the parameters optimization the *Model zero* was used: the accuracy of this model before the parameters and hyperparameters optimization was 73% and the loss was 0.71 over the test set.

3.1.1 Cross validation

The first parameters to be investigated were the activation function and the Optimizer.

Activation function	Accuracy
relu	0.743
linear	0.501
tanh	0.750
sigmoid	**0.757**
hard sigmoid	**0.754**
softmax	0.729
elu	0.734
selu	0.718
softplus	0.749
softsign	**0.756**

Optimizer	Accuracy
sgd	0.679
rmsprop	0.727
Adagrad	0.683
Adadelta	0.71
Adam	**0.74**
Adamax	0.734
Nadam	**0.742**

As it can be seen the sigmoid activation function clearly reaches the highest values. Contrarily, the optimizer choice has been trickier because both Adam and Nadam return very similar accuracies. Because of this reason, Adam and Nadam optimizers were further investigated with another cross validation perfromed by varying the learning rate.

Learning rate	Adam accuracy	Nadam accuracy
0.001	0.748	0.750
0.002	0.754	0.750
0.003	0.760	0.761
0.004	0.761	0.755
0.005	0.761	0.761
0.006	0.762	0.762
0.007	**0.763**	0.757
0.008	0.759	0.751
0.01	0.759	0.750
0.03	0.745	0.669
0.06	0.661	0.254
0.1	0.351	0.252
0.2	0.163	0.141
0.3	0.199	0.141
0.5	0.141	0.141

Table 3.1: Adam and Nadam optimization algorithm CV

The Adam optimizer showed better performances, hence the initial choiche of using it was confirmed. Moreover, the Stocastic Gradient Descent (SGD) is investigated because is well known that if used in a MLP with the proper learning rate and momentum values it can lead to better performances than the others optimization algorithms. Unfortunately, the SGD performances were much lower than Adam optimizers.

SGD learning rate	SGD momentum	Accuracy
0.005	0.0	0.577
0.005	0.2	0.579
0.005	0.4	0.579
0.01	0.0	0.625
0.01	0.2	0.623
0.01	0.4	0.624
0.03	0.0	0.673
0.03	0.2	0.674
0.03	0.4	0.675
0.1	0.0	0.673
0.1	0.2	0.675
0.1	0.4	0.677
0.2	0.0	0.469
0.2	0.2	0.471
0.2	0.4	0.390
0.3	0.0	0.16
0.3	0.2	0.532
0.3	0.4	0.337
0.5	0.0	0.49
0.5	0.2	0.166
0.5	0.4	0.143

Table 3.2: SGD CV

Epochs and batch size were explored after the optimization algorithm grid search.

Epochs	Accuracy
500	0.755
750	0.757
1000	0.759
1250	0.758
1500	**0.761**
1750	0.756
2000	0.755
2500	0.754

Batch size	Accuracy
32	0.753
64	0.756
128	0.758
256	0.758
512	0.760
1024	**0.762**
2048	0.757
4096	0.749

Table 3.3: Epochs number and Batch size CV

The regularization was assessed by changing the dropout rate first by 0.1 step into the range [0-1]. Because a descending trend was observed with the increase of dropout rate, the range was reduced to [0-0.1] and the step to 0.02. Because the best result was given by Dropout 0.02, a third search was given for values close to 0.2. The best result was achieved with a 0.13 dropout rate. The initialization mode was tested, with a good result shown by *lecun uniform* and *glorot normal*.

Dropout	Accuracy
0.0	**0.757**
0.1	0.749
0.2	0.737
0.4	0.721
0.5	0.711
0.6	0.698
0.8	0.642
0.9	0.551
0.02	**0.761**
0.04	0.759
0.06	0.753
0.08	0.751
0.013	**0.764**
0.017	0.763
0.023	0.761
0.026	0.760

Initialization mode	Accuracy
uniform	0.753
normal	0.757
zero	0.609
lecun uniform	**0.762**
glorot normal	**0.761**
glorot uniform	0.759
he normal	0.760
he uniform	**0.761**

Table 3.4: Dropout rate and Initialization mode CV

Finally a wide search over the activation functions and Initialization mode was done in order to find the best combination, that resulted to be the use of *sigmoid function* together with *glorot uniform*.

Activation function	Initialization mode	Accuracy
relu	uniform	0.740
relu	normal	0.742
relu	zero	0.142
linear	uniform	0.503
linear	normal	0.501
linear	zero	0.145
tanh	uniform	0.747
tanh	normal	0.750
tanh	zero	0.140
sigmoid	uniform	0.754
sigmoid	normal	0.757
sigmoid	zero	0.609
sigmoid	lecun uniform	0.761
sigmoid	glorot uniform	**0.763**
sigmoid	glorot normal	0.759
sigmoid	he normal	0.760
sigmoid	he uniform	0.759
hard sigmoid	lecun uniform	0.755
hard sigmoid	glorot uniform	0.757
hard sigmoid	glorot normal	0.754
hard sigmoid	he normal	0.756
hard sigmoid	he uniform	0.754
softsign	lecun uniform	0.756
softsign	glorot uniform	0.756
softsign	glorot normal	0.756
softsign	he normal	0.756
softsign	he uniform	0.755

Table 3.5: Activation function and initialization mode combined CV

3.1.2 Manual tuning

Architecture tuning

The first task of *architecture tuning* is to understand what are the architectures that represent the best compromise between low complexity and high

performances.

As it can be seen from the following figures, increasing the architecture size, the performances over the test set in terms of accuracy 3.1, loss 3.2, F1-score 3.4, Precision 3.6 and Recall 3.5 firstly improve and subsequently reach a plateau, that represents the best achievable performances. The *plateau average* value is calculated for each of these metrics, by averaging the values of the 3, 4 and 5 layer architectures for each curve. Then, a *plateau threshold* was calculated as

$$plateau\_threshold = plateau\_average - ((2)\%plateau\_average) \tag{3.1}$$

for accuracy, Recall, Precision and F1-score and as

$$plateau\_threshold = plateau\_average + ((3)\%plateau\_average) \tag{3.2}$$

for the loss metric. The plateau threshold is represented in the graphs by a dotted line and make easy to understand which are the architectures with performance values comparable to the plateau average, i.e. the ones eligible as *final model architecture*.

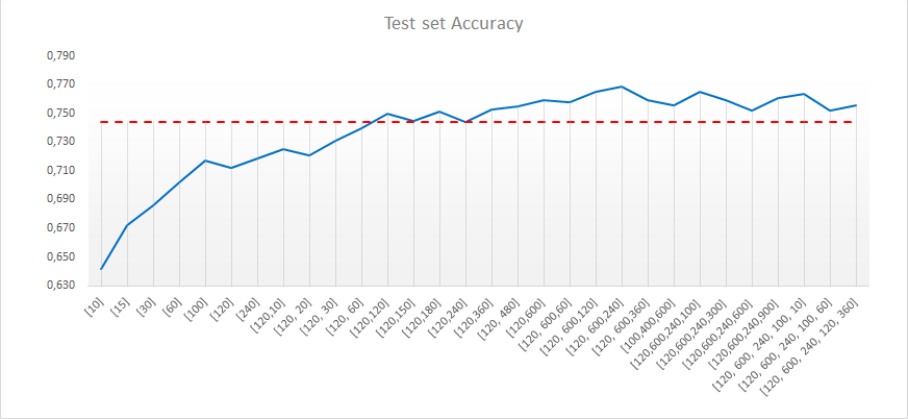

Figure 3.1: Accuracy variation as a function of width and depth. The red line represent the *plateau threshold*.

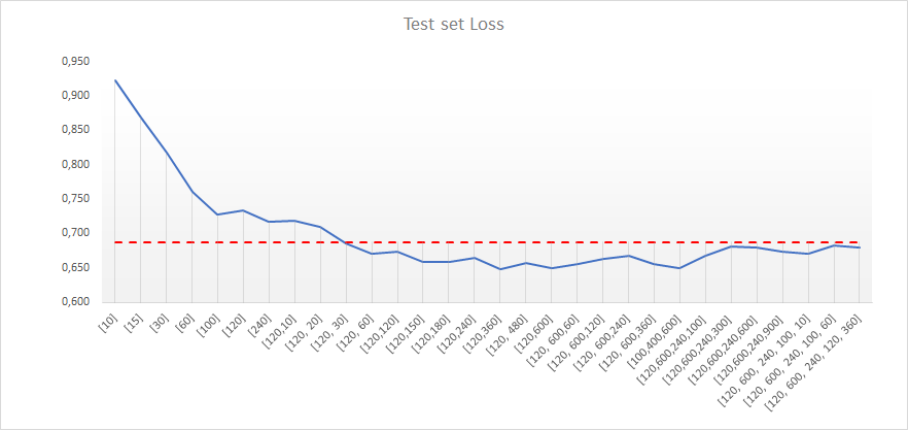

Figure 3.2: Loss variation as a function of width and depth. The red line represent the *plateau threshold*.

Let us call these *the plateau architectures*. Although *the plateau architectures* have similar performances in terms of loss and accuracy, their computational complexity, which is described by the number of *total parameters*, varies a lot when the width and the depth increase (see Figure 3.3).

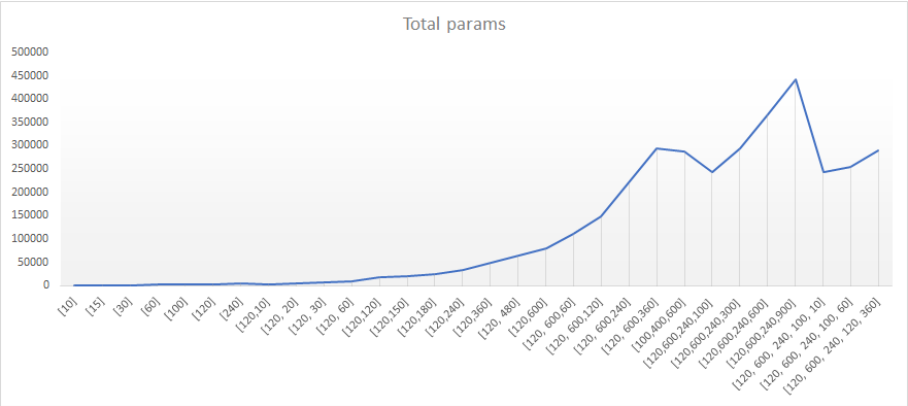

Figure 3.3: Keras model number of total parameters variation of the different architectures tested

The preferred ANNs for the MCU implementation are those with a little number of total parameter, i.e. the ones on the left side of the Figure 3.3.

For example, among all of the architecture tests, [120,360] is considered a good compromise because it shows one of the lowest complexities, with

48007 total parameters, and loss and accuracy values within the plateau threshold. Contrarily, the choice of the architecture [120, 600, 240, 120, 360] would not add significant information respect to the [120,360] model, though it contains much more neurons, because it shows similar values of accuracy and loss, resulting only heavier.

The F1-score metric clarifies what are the performances for each different class. All of them show the characteristic plateau. The plateau average and *plateau threshold* have been calculated separately for each class.

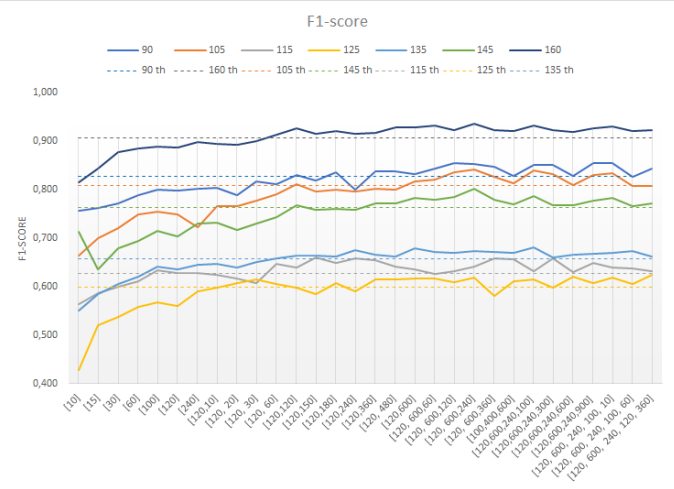

Figure 3.4: F1-score variation of all SP classes as a function of ANN architecture. The *plateau thresholds* are represented by a dot line of the same color of the correspondent class.

From the plot it is possible to see that [120,120] architecture is the minimum complexity architecture by which the plateau threshold has been reached by all classes curves.

Moreover, by F1-score plateau average values of Table 3.6 it is introduced an interesting aspect of the performances over the different classes: the external classes show much better performances than the central ones.

Class	**90**	**105**	**115**	**125**	**135**	**145**	**160**
F1-score	0.84	0.82	0.64	0.61	0.67	0.78	0.93

Table 3.6: Plateau average values for F1-score metric

This fact is remarked by the Recall and Precision analysis over the classes, which plateau average values are shown in the table 3.7:

Class	**90**	**105**	**115**	**125**	**135**	**145**	**160**
Recall	0.83	0.86	0.60	0.61	0.67	0.81	0.94
Precision	0.86	0.80	0.70	0.61	0.68	0.75	0.92

Table 3.7: Plateau average values for Recall and Precision

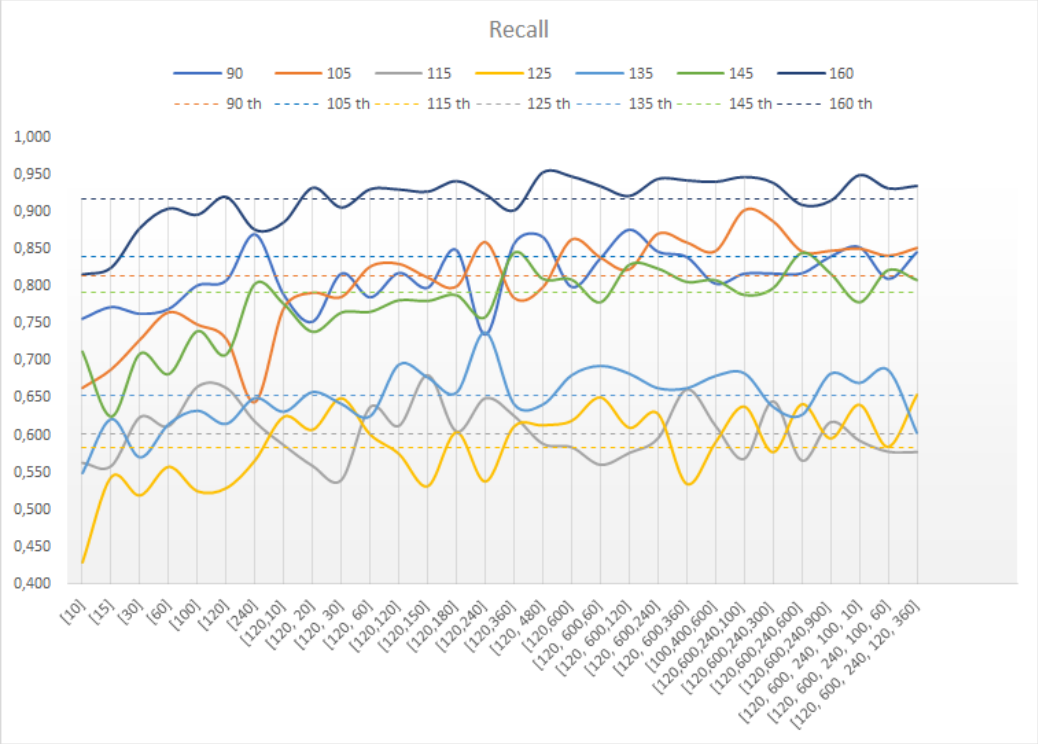

Figure 3.5: Classes Recall variation as a function of ANN architecture

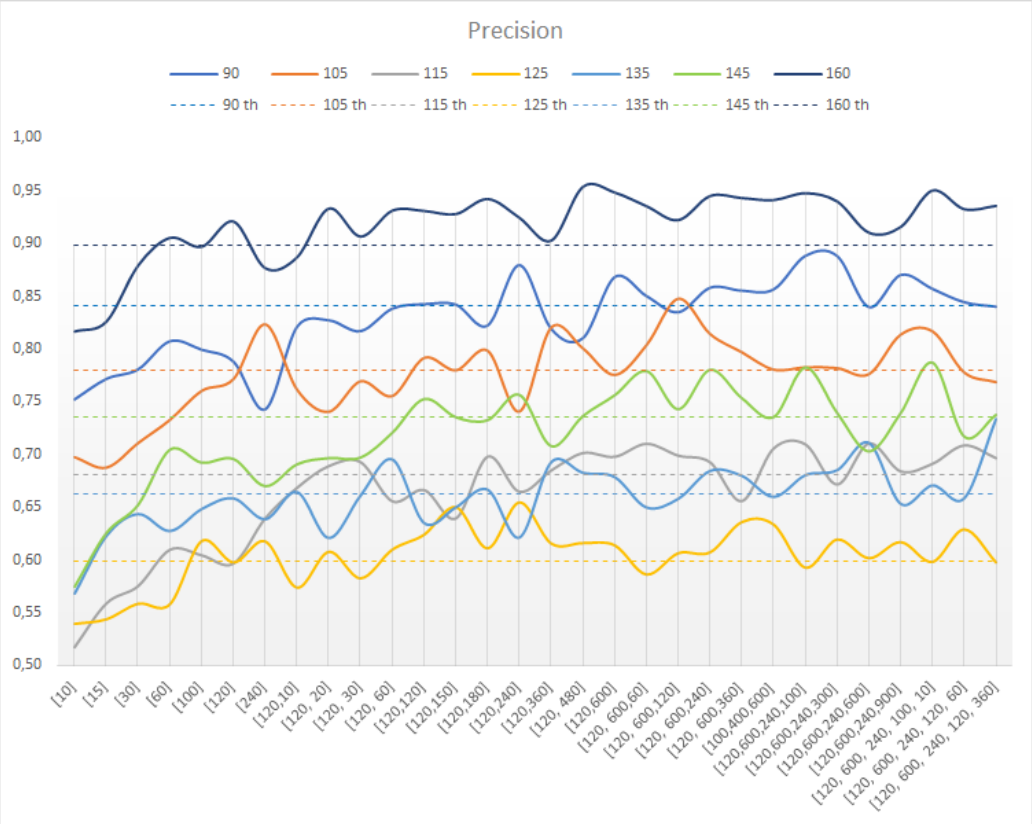

Figure 3.6: Classes Precision variation as a function of ANN architecture

In Table 3.8 the classes were ordered basing on Recall performances:

Class	**SP range** (mmHg)	**Precision**	**Recall**
160	[150 - 170]	0.92	0.94
105	[100 - 109]	0.80	0.86
90	[80 -99]	0.86	0.83
145	[140-149]	0.75	0.81
135	[130 - 139]	0.68	0.67
115	[110 - 119]	0.7	0.60
125	[120 - 129]	0.61	0.61

Table 3.8: Summary of precision and recall average values after reaching the plateau in order of Recall performances

The outcome of the Recall plot analysis is that the most performant architectures for Recall oltimization are [120,120], [120,240], [120, 600], [120, 600, 60], [120, 600, 120] and [120, 600, 240, 100, 60]. Of course, the first two are the preferred because of the lowest computational cost.

	Recall			**Precision**		
Class	**135**	**145**	**160**	**135**	**145**	**160**
[120,120]	0,70	0,78	0,93	0,63	0,75	0,92
[120,240]	0,74	0,76	0,92	0,62	0,76	0,91
[120, 600]	0,68	0,81	0,95	0,68	0,76	0,91
[120, 600,60]	0,69	0,78	0,94	0,65	0,78	0,93
[120, 600,120]	0,68	0,83	0,92	0,66	0,74	0,92
[120, 600, 240, 100, 60]	0,69	0,82	0,93	0,66	0,72	0,91

From this table it is clear the trade-off between high recall of the 145 class (above 0.8) and good enough recall for 135 class (above 0.7). A good choice would be to maximize the recall for the most critical classes, i.e. 145 and 160, and in the meanwhile look for a good Precision for the less critical class, i.e. 135. This compromise is obtained with the [120, 600] architecture, which raises 160 class Recall to 0.95, its highest value, and 145 class Recall to 0.81, which is almost the maximum value reached by this class over the whole trials. Furthermore, it can be seen as this architecture choice maximizes the Precision value of the 135 class, that reaches 0.68 value, without penalizing the precision of 145 and 160 classes that stay high for both cases. Hence the architecture **[120, 600]** was chosen as the **final model architecture**. The number total parameters of this architecture is 78727.

3.2 Final model

After the manual tuning and the cross validations performed, the final model was defined as:

- architecture: [120, 600];
- optimization algorithm: Adam
- learning rate: 0.007;
- initialization mode: glorot_uniform;
- activation function (hidden layers): sigmoid ;
- Dropout rate: 0.013;
- Batch size: 1024;
- Epochs number:1500;

Class	**Precision**	**Recall**	**F1-score**
90	0,91	0,88	0,89
105	0,85	0,92	0,88
115	0,67	0,60	0,63
125	0,60	0,61	0,61
135	0,71	0,69	0,70
145	0,83	0,88	0,85
160	0,95	0,97	0,96
accuracy	0.79		

Table 3.9: Adam and Nadam optimization algorithm CV

Ground truth classes \ Predicted classes	90	105	115	125	135	145	160
90	2654	290	25	17	21	15	3
105	151	2761	51	9	18	4	3
115	56	153	1812	722	204	67	8
125	28	20	588	1863	415	140	6
135	26	30	179	423	2024	221	10
145	3	3	33	61	147	2635	120
160	4	3	1	1	2	81	2888

Figure 3.7: Final model confusion matrix

Ground truth classes \ Predicted classes	90	105	115	125	135	145	160
90	88,5	9,7	0,8	0,6	0,7	0,5	0,1
105	5,0	92,0	1,7	0,3	0,6	0,1	0,1
115	1,9	5,1	60,4	24,1	6,8	2,2	0,3
125	0,9	0,7	19,6	62,1	13,8	4,7	0,2
135	0,9	1,0	6,0	14,1	67,5	7,4	0,3
145	0,1	0,1	1,1	2,0	4,9	87,8	4,0
160	0,1	0,1	0,0	0,0	0,1	2,7	96,3

Figure 3.8: Final model confusion matrix expressed as percentage of the total class elements.

3.3 Computational cost of the final ANN model

The results obtained from the **STM32Cube.AI** toolkit are presented below.

3.3.1 ROM and RAM required

The following results should be compared with the STM32L4R7 MCU memories:

- ROM: 2048 kB;
- RAM: 640 kB;

The input and output are stored in FLOAT32 format, i.e. each value requires 4 Bytes. Being the input and output sizes 15 and 7 respectively, the total memory needed for them can be obtained by multiplying these sizes by 4.

	no compr	**compr by 4**	**compr by 8**
input	60 B	60 B	60 B
output	28 B	28 B	28 B
macc	85305	85305	85305
weights	307.53 KB	86.29 KB	41.17 KB
activations	2.81 KB	2.81 KB	2.81 KB
ROM (total)	307.53 KB	86.29 KB (-71.94%)	41.17 KB (-86.61%)
RAM (total)	2.90 KB	2.90 KB	2.90 KB

Table 3.10: General details about the ANN *final model* requirements in terms of maccs, ROM and RAM. Comparison of no compression (no compr), compression by 4 (compr by 4) and compression by 8 (compr by 8) cases.

Considering that in C language each Keras Dense layer is splitted into two layers, a Dense layer (indicated as D) and a Non-linearity layer (indicated as NL), the layer by layer analysis are presented below.

Layer	out shape	param #	macc (#)
Input	(15,)		
1 D	(120,)	1920	1800
1 NL	(120,)		1200
2 D	(600,)	72600	72000
2 NL	(600,)		6000
out D	(7,)	4207	4200
out NL	(7,)		105

Table 3.11: macc required for each layer.

Layer	out shape	params #	ROM (kB)		
			no compr	compr by 4	compr by 8
Input	(15,)				
1 D	(120,)	1920	7.5	7.5	7.5
1 NL	(120,)				
2 D	(600,)	72600	283.59	3.66	37.56
2 NL	(600,)				
out D	(7,)	4207	16.43	5.13	2.14
out NL	(7,)				

	total (kB)		
ROM required	307,53	86,29	41,17
ROM required reduction		-71,94%	-86,61%

Table 3.12: Layer by layer ROM requirements comparison. The cases of no compression (no compr), compression by 4 (compr by 4) and compression by 8 are compared.

Layer	out shape	param #	macc (%)	rom (%)
Input	(15,)		0	0
1 D	(120,)	1920	2.1	3.6
1 NL	(120,)		1.4	0
2 D	(600,)	72600	84.4	91.2
2 NL	(600,)		7	0
out D	(7,)	4207	4.9	5.2
out NL	(7,)		0.1	0

Table 3.13: Layer by layer analysis over the layer load in terms of macc and ROM percentage

	no compr	compr by 4	compr by 8
cross correlation accuracy	100%	100%	50%

Table 3.14: Validation results of cross correlation between the reference and the C-compiled model ouput. The accuracy is not to be intended as the metric, but as *accuracy in reproducing original model performances*

elapsed time(s)	Original	no compr	compr by 4	compr by 8
10 inputs	0.202	6.15	6.14	6.15
single inference	0.0202	0.615	0.614	0.615

Table 3.15: Validation results of inference time. The elapsed time of the single inference time is the one that counts.

3.4 Results Discussion

The ANN classification performances As it can be seen from the final ANN model results, the classification performances are acceptable for continuous monitoring. The Recall over the most critical classes 145 and 160 indicating *grade 1* and *grade 2 hypertension* show a very high Recall and Precision. This means that a hypertension case of this type, that represents a high death risk, can be correctly detected by the monitoring device.

Furthermore, it can be seen from the 3.7, that the class 160, for the 93% of the misclassification is wrongly classified in the 145 class, which is still an hypertension class. Hence, the error over this class can be considered close to zero, considering that the misclassification would generate an hypertension alarm in 93% of cases.

The 145 class is classified for the 4.9% of times as 135 and 4.0% of times in the 160 class. Hence, the classification error of this class is not very dangerous because:

- if the classification is 135, it still generates a state of alert, because it represents high-normal SP values
- if the classification is 160, it is a False positive of critical hypertension a little bit more severe than the 145 class one, but it is less dangerous than a False negative.

The potentially dangerous misclassification of the 145 class is represented by 90, 105, 115, 125 and 135 classes, that represent the 0.67% of the total misclassification, that is the 8.17% of classifications.

The class 135, corresponding to *high normal* pressure, presents a 69% recall and 71% precision, which can be considered good enough for a class which represent a lower death risk than the previous ones. This is a class in which both a good Recall and a good Precision would be required because:

- a false positive in the 115 and 125 classes would result in absence of alarm, which is not correct, being 135 a value indicating not hypertension, but still a high pressure;
- each false positive that falls in the 145 class would generate an alarm that overestimates the risk.

Considering that the 135 represent a potentially risky condition over long-term, its classification are considered good enough for an hypertension monitoring system, but they should be increased at least to 80% for both Recall and Precision.

The classification over the classes 115 and 125, indicating *normal pressure*, are not very satisfactory, with a Recall of 0.6 for both classes and a precision of 0.6 for the 125 class and 0.67 for the 115 class. This is not a desirable performance in classification, but considering that these classes do not represent a pathological condition, this value can be considered acceptable. It can be seen from 3.7 and 3.8 that the ANN reaches its major misclassification errors for these two classes, that are often exchanged between them during the prediction.

The ANN computational cost The computational cost of the final model is low enough to allow the ANN embedding into the STML4R7 MCU. The original model, without compression, already shows satisfactory memory requirements: it would occupy only the 15 % of the ROM and the 0,4 % of the RAM. This is a very good computational cost, that allows the embedding of the ANN on a MCU that, other than the pressure monitoring, contains other functionalities and needs to reserve ROM and RAM space to other applications.

Although this very low computational load, if it is desired by the user, that for example wants to embed the same model on a less performant board, the ROM occupation can be lowered to 4.3% and 2.0% in case of compression by 4 and by 8 respectively. Moreover, it can be seen from Table 3.12 3.13 that the layer which requires more ROM memory usage is the second layer of the model, that contains 600 neurons. By adopting another architecture with a smaller second layer, such as [120, 120] or [120,240], the performances would be penalized by a few percentage points, but the load of the model could be further decreased.

From the cross-correlation results it can be seen as the not compressed and the compressed by 4 C-compiled model are perfectly suitable for the embedding, because they show an accuracy of 100% with respect to the reference (original model result). On the other hand, for the compressed by 8 model the cross correlation accuracy drops to 50%, meaning that before embedding it into a MCU, further analysis over the model performances should be done.

The time requested by an inference has been described in 3.15. The prediction time for the single input is slightly too high for a beat-to-beat monitoring. In fact, as it was said, for the beat-to-beat prediction the inference time should have been inferior to 0.6 seconds in healthy conditions. However, the time window required for a single prediction largely satisfies the necessity of pressure monitoring in real applications of systolic pressure continuous monitoring. Furthermore, in order to allow the pressure beat-to-beat monitoring on people with tachyarrhythmias, the inference time should be lowered to a lower value. If the inference time was 0.4 seconds, it would allow the inference of 150 beats per minutes, while a 0.33 inference time would allow the inference of 180 beats per minutes.

In fact, the classical techniques pressure measuring devices have a time resolution from 3 minutes (sphygmomanometer) to 20 minutes (oscillometric devices).

3.4.1 Future improvements

The optimization process led to an 8% improvement in accuracy over the test set. Future expedients could improve very much the ANN performances by improving the dataset:

- Enlarging of the dataset by number of patients records and by increasing the number of training examples of the classes that were represented by many less number of values i order to avoid the upsampling of some classes examples;
- Creation of a dataset specific for the device in which the ANN should be embedded. In this way the data would be device specific and the performances would be higher;
- moving average over a window of several predictions: in this way, false positives and false negatives value should be averaged out and a value of systolic pressure closer to the real one would be achieved.
- incrementation of features calculated over the morphology

Moreover, the classes representing the two ranges 110-119 and 120-129 mmHg could be merged in the classification process because both represent normal values of SP and a 20 mmHg interval class would not arise problems. Moreover, since the two classes distinguish is the major difficulty of the

model, probably the performances over these two classes would be increased by merging them.

Chapter 4

Conclusions and future applications

4.1 Conclusion

The aim of this project was to to create a MCU embeddable classifier based on neural networks able to correctly predict the systolic pressure range from the Photoplethysmography signal morphology. The work was not easy due to the non linear relationship between PPG morphology and SP. A large dataset containing 124616 PPG periods was created and 15 features were calculated for each of them. A supervised learning algorithm was chosen and, hence, each PPG period of the dataset was labelled with the correct SP value. Furthermore, the features were standardized, the targets discretized and one-hot-encoded and the dataset itself was balanced in order to present 20000 examples per class.

The ANN model chosen was a MLP with 15 input neurons, and 7 output classes representing 7 different SP ranges. The initial model had good values of classification performances, with an initial accuracy of around 73 % over the test set. After a cross validation of the Keras model parameters and a manual tuning of the MLP depth and width, the accuracy over the test set raised to 79%. The cross validation was performed on activation function, optimizer, learning rate, epochs number, batch size, Dropout, initialization mode and a combination of activation function and initialization mode. After the cross validation, the manual tuning was performed over 1,2,3,4 and 5 depth architectures and the best models were decided by assessing the Recall performances over the most critical classes that represent grade I and II hypertension, i.e. 145 and 160.

The final model achieved 97% and 88% recall and 95% and 83% precision values for 160 and 145 respectively. This is a very good result because a critical hypertension value can be detected in a very accurate way and with a very low percentage of error. For the very low SP classes such as 90 and 105 the results are still over 85% for both precision and recall. The only classes that the MLP does not classify accurately are 115, 125 and 135, meaning that it gets confused over them. By assessing the confusion matrix of the *final model*, 115 and 125 are the two classes that the ANN confuses the most. All of this can be confirmed with the co

The memory required from the final chosen model, if compiled in C and embedded into a MCU is of 307 kB of ROM and 2.9 kB of RAM, that is acceptable since the STM32 MCUs ROM and RAM can reach up to 20148 kB and 640 respectively. The computational cost of the ANN is acceptable

for be embedded into a STM32L4R7, because the time required for each inference is 614 ms.

Bibliography

[1] Arterial blood pressure.

[2] Pierre Agache and Philippe Humbert. *Measuring the skin*. 2004.

[3] John Allen. Photoplethysmography and its application in clinical physiological measurement, 2007.

[4] Caerwyn Ash, Michael Dubec, Kelvin Donne, and Tim Bashford. Effect of wavelength and beam width on penetration in light-tissue interaction using computational methods. *Lasers in Medical Science*, 2017.

[5] Gladimir V. G. Baranoski and Aravind Krishnaswamy. An Introduction to Light Interaction with Human Skin. *Revista de Informática Teórica e Aplicada*, 2004.

[6] Anubha Bilgaiyan, Ryutaro Sugawara, Fahed Elsamnah, Shim Chang-Hoon, Muhamad Affiq, and Reiji Hattori. Optimizing performance of reflectance-based organic Photoplethysmogram (PPG) sensor. 2018.

[7] Elena Chung, Guo Chen, Brenton Alexander, and Maxime Cannesson. Non-invasive continuous blood pressure monitoring: A review of current applications, 2013.

[8] Google Developers. Machine Learning crash course.

[9] Google Developers. Keras API, 2020.

[10] Mohamad Forouzanfar, Hilmi R. Dajani, Voicu Z. Groza, Miodrag Bolic, Sreeraman Rajan, and Izmail Batkin. Oscillometric blood pressure estimation: Past, present, and future. *IEEE Reviews in Biomedical Engineering*, 2015.

[11] Giancarlo Fortino and Valerio Giampà. PPG-based methods for non invasive and continuous blood pressure measurement: An overview and development issues in body sensor networks. In *2010 IEEE International Workshop on Medical Measurements and Applications, MeMeA 2010 - Proceedings*, 2010.

[12] Ricardo Fuentes, Nina Ilmaniemi, Eija Laurikainen, Jaakko Tuomilehto, and Aulikki Nissinen. Hypertension in developing economies: A review of population-based studies carried out from 1980 to 1998. *Journal of Hypertension*, 2000.

[13] L. A. Geddes, M. Voelz, S. James, and D. Reiner. Pulse arrival time as a method of obtaining systolic and diastolic blood pressure indirectly. *Medical & Biological Engineering & Computing*, 1981.

[14] Marion Geerligs. *Skin layer mechanics*. 2010.

[15] Mohammad Ghamari. A review on wearable photoplethysmography sensors and their potential future applications in health care. *International Journal of Biosensors & Bioelectronics*, 2018.

[16] D Girola. *Cardiologia e fitness - Prevenzione cardiologica applicata al fitness : valutazione funzionale - protocolli terapeutici e di allenamento - casi clinici*. 2009.

[17] Hamed Asadi Hamed Akhlaghi. Essentials of telemedicine and telecare. *researchgate*, 2002.

[18] Matthew J. Hayes and Peter R. Smith. Artifact reduction in photoplethysmography. *Applied Optics*, 1998.

[19] Alrick B. Hertzman. THE BLOOD SUPPLY OF VARIOUS SKIN AREAS AS ESTIMATED BY THE PHOTOELECTRIC PLETHYSMOGRAPH. *American Journal of Physiology-Legacy Content*, 1938.

[20] Mary Lee Hummert and Jon F. Nussbaum. Aging, communication, and health: linking research and practice for successful aging. *Choice Reviews Online*, 2001.

[21] Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li Wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 2016.

[22] Deric P. Jones. Medical electro-optics: Measurements in the human microcirculation. *Physics in Technology*, 1987.

[23] A. A.R. Kamal, J. B. Harness, G. Irving, and A. J. Mearns. Skin photoplethysmography - a review. *Computer Methods and Programs in Biomedicine*, 1989.

[24] Alexei A. Kamshilin and Nikita B. Margaryants. Origin of Photoplethysmographic Waveform at Green Light. In *Physics Procedia*, 2017.

[25] Khan Academy. The circulatory system review. 2020.

[26] Kevin Kinsella and David R. Phillips. Global Aging: The challenge of success. *Population Bulletin*, 2005.

[27] Miroslav Kubat. Neural networks: a comprehensive foundation by Simon Haykin, Macmillan, 1994, ISBN 0-02-352781-7. . *The Knowledge Engineering Review*, 1999.

[28] Jitendra Kumar. Epidemiology of hypertension. *Elsevier*, 2013.

[29] Yuriy Kurylyak, Francesco Lamonaca, and Domenico Grimaldi. A Neural Network-based method for continuous blood pressure estimation from a PPG signal.

[30] Mathieu Lemay, Mattia Bertschi, Josep Sola, Philippe Renevey, Jakub Parak, and Ilkka Korhonen. Application of Optical Heart Rate Monitoring. In *Wearable Sensors: Fundamentals, Implementation and Applications*. 2014.

[31] C.J. Malanga. Structure and Function of the Heart. *xPharm: The Comprehensive Pharmacology Reference*, 2007.

[32] Giuseppe Mancia and Robert Fagard. 2013 ESH/ESC Guidelines for the management of arterial hypertension. *European Heart Journal*, 2013.

[33] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 1943.

[34] Andreia V. Moço, Sander Stuijk, and Gerard De Haan. New insights into the origin of remote PPG signals in visible light and infrared. *Scientific Reports*, 2018.

[35] Jermana L. Moraes, Matheus X. Rocha, Glauber G. Vasconcelos, José E. Vasconcelos Filho, Victor Hugo C. de Albuquerque, and Auzuir R. Alexandria. Advances in photopletysmography signal analysis for biomedical applications, 2018.

[36] Paul Muntner, Daichi Shimbo, Robert M. Carey, Jeanne B. Charleston, Trudy Gaillard, Sanjay Misra, Martin G. Myers, Gbenga Ogedegbe, Joseph E. Schwartz, Raymond R. Townsend, Elaine M. Urbina, Anthony J. Viera, William B. White, and Jackson T. Wright. Measurement of blood pressure in humans: A scientific statement from the american heart association. *Hypertension*, 2019.

[37] Vishal Nangalia, David R. Prytherch, and Gary B. Smith. Health technology assessment review: Remote monitoring of vital signs - current status and future challenges, 2010.

[38] Markolf H Niemz, J. Langer, and Markolf H Niemz. *Laser-tissue interactions: fundamentals and applications*. 2007.

[39] L. Peter, N. Noury, and M. Cerny. A review of methods for non-invasive and continuous blood pressure monitoring: Pulse transit time method is promising?, 2014.

[40] K. N. G. Priyanka, Paul C.-P. Chao, Tse-Yi Tu, Yung-Hua Kao, Ming-Hua Yeh, Rajeev Pandey, and Fitrah P. Eka. Estimating Blood Pressure via Artificial Neural Networks Based on Measured Photoplethysmography Waveforms. 2018.

[41] Raymond Kurzweil. *Age of Intelligent machines*. 1990.

[42] Andrew Reisner, Phillip A. Shaltis, Devin McCombie, and H. Harry Asada. Utility of the photoplethysmogram in circulatory monitoring, 2008.

[43] Juan C. Ruiz-Rodríguez, Adolf Ruiz-Sanmartín, Vicent Ribas, Jesús Caballero, Alejandra García-Roche, Jordi Riera, Xavier Nuvials, Miriam De Nadal, Oriol De Sola-Morales, Joaquim Serra, and Jordi Rello. Innovative continuous non-invasive cuffless blood pressure monitoring based on photoplethysmography technology. In *Intensive Care Medicine*, 2013.

[44] Stuart Russell and Peter Norvig. *Artificial Intelligence A Modern Approach Third Edition*. 2010.

[45] Rohan Samria, Ridhi Jain, Ankita Jha, Sandeep Saini, and Shubhajit Roy Chowdhury. Noninvasive cuffless estimation of blood pressure using Photoplethysmography without Electrocardiograph Measurement. *IEEE*, 2014.

[46] Janis Spigulis. Optical noninvasive monitoring of skin blood pulsations. In *Applied Optics*, 2005.

[47] STMicroelectronics. STM32Cube.AI, 2019.

[48] Peng Su, Xiao Rong Ding, Yuan Ting Zhang, Jing Liu, Fen Miao, and Ni Zhao. Long-term blood pressure prediction with deep recurrent neural networks. In *2018 IEEE EMBS International Conference on Biomedical and Health Informatics, BHI 2018*, 2018.

[49] Yu Sun and Nitish Thakor. Photoplethysmography Revisited: From Contact to Noncontact, from Point to Imaging, 2016.

[50] Satomi Suzuki and Koji Oguri. Cuffless and non-invasive Systolic Blood Pressure estimation for aged class by using a Photoplethysmograph. In *Proceedings of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS'08 - "Personalized Healthcare through Technology"*, 2008.

[51] Daniel Svozil, Vladimír Kvasnička, and Jiří Pospíchal. Introduction to multi-layer feed-forward neural networks. In *Chemometrics and Intelligent Laboratory Systems*, 1997.

[52] F. Nylund T. Persson. Development of a photopletysmography based method for investigating changes in blood volume pulsations: for the purpose of pressure ulcer prevention. 2017.

[53] Hirofumi Tanaka, Gerardo Heiss, Elizabeth L. McCabe, Michelle L. Meyer, Amil M. Shah, Judy R. Mangion, Justina Wu, Scott D. Solomon, and Susan Cheng. Hemodynamic Correlates of Blood Pressure in Older Adults: The Atherosclerosis Risk in Communities (ARIC) Study. *Journal of Clinical Hypertension*, 2016.

[54] X. F. Teng and Y. T. Zhang. Continuous and Noninvasive Estimation of Arterial Blood Pressure Using a Photoplethysmographic Approach. In *Annual International Conference of the IEEE Engineering in Medicine and Biology - Proceedings*, 2003.

[55] Texas Heart Institute. Heart Anatomy.

[56] Sophie White. The Conducting System of the Heart. 2019.

[57] Wikipedia. Neuron.

[58] Wikipedia. Overfitting.

[59] Michael Williams and John Haugeland. Artificial Intelligence: The Very Idea. *Technology and Culture*, 1987.

[60] Mehmet R. Yuce. Implementation of wireless body area networks for healthcare systems. *Sensors and Actuators, A: Physical*, 2010.

[61] Y Zhang and Z Wang. A hybrid model for blood pressure prediction from a PPG signal based on MIV and GA-BP neural network. 2017.

[62] Yan Zhang, Nirwan Ansari, and Hiroshi Tsunoda. Wireless telemedicine services over integrated IEEE 802.11/WLAN and IEEE 802.16/WiMAX networks. *IEEE Wireless Communications*, 2010.

Appendix A

Bibliography