Remote photoplethysmography measurement and filtering using deep learning based methods

Deivid Botina Monsalve

THÈSE DE DOCTORAT DE L'ÉTABLISSEMENT UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ

PRÉPARÉE À L'UNIVERSITÉ DE BOURGOGNE

École doctorale n°37

Sciences Pour l'Ingénieur et Microtechniques

Doctorat d'Instrumentation et Informatique de l'Image

par

DEIVID JOHAN BOTINA MONSALVE

Remote photoplethysmography measurement and filtering using deep learning based methods

Thèse présentée et soutenue à Dijon, le 15 Novembre 2022

Composition du Jury :

WANNOUS HAZEM	Professeur à l'Univserité de Lile Institut Mines-Telecom	Rapporteur
HISTACE AYMERIC	Professeur à ENSEA ETIS Laboratory	Rapporteur
DANTCHEVA ANTITZA	Chargée de recherche à INRIA Inria Sophia Antipolis - Méditerranée	Examinatrice
NOURY NORBERT	Professeur à INSA Université de Lyon	Examinateur
MITERAN JOHEL	Professeur des universités à l'Université de Bourgogne	Directeur de thèse
BENEZETH YANNICK	Maître de conférences à l'Université de Bourgogne	Codirecteur de thèse

N° | X | X | X |

SPIM
école doctorale sciences pour l'ingénieur et microtechniques

Title: Remote photoplethysmography measurement and filtering using deep learning based methods

Keywords: remote photoplethysmography, heart rate, heart rate variability, LSTM, 3DCNN.

Abstract:

RPPG is a technique developed to measure the blood volume pulse signal and then estimate physiological data such as pulse rate, breathing rate, and pulse rate variability.

Due to the multiple sources of noise that deteriorate the quality of the RPPG signal, conventional filters are commonly used. However, some alterations remain, but interestingly, an experienced eye can easily identify them. In the first part of this thesis, we propose the Long Short-Term Memory Deep-Filter (LSTMDF) network in the RPPG filtering task. We use different protocols to analyze the performance of the method. We demonstrate how the network can be efficiently trained with a few signals. Our study demonstrates experimentally the superiority of the LSTM-based filter compared with conventional filters. We found a network sensitivity related to the average signal-to-noise ratio on the RPPG signals.

Approaches based on convolutional networks such as 3DCNNs have recently outperformed traditional hand-crafted methods in the RPPG measurement task. However, it is well known that large 3DCNN models have high computational costs and may be unsuitable for real-time applications. As the second contribution of this thesis, we propose a study of a 3DCNN architecture, finding the best compromise between pulse rate measurement precision and inference time. We use an ablation study where we decrease the input size, propose a custom loss function, and evaluate the impact of different input color spaces. The result is the Real-Time RPPG (RTRPPG), an end-to-end RPPG measurement framework that can be used in GPU and CPU. We also proposed a data augmentation method that aims to improve the performance of deep learning networks when the database has specific characteristics (e.g., fitness movement) and when there is not enough data available.

Titre : Remote photoplethysmography measurement and filtering using deep learning based methods

Mots-clés : Photopléthysmographie à distance, rythme cardiaque, la variabilité de la fréquence cardiaque, LSTM, 3DCNN.

Résumé :

RPPG est une technique développée pour mesurer le signal du pouls et ensuite estimer les données physiologiques telles que la fréquence cardiaque, la fréquence respiratoire et la variabilité du pouls.

En raison des multiples sources de bruit qui détériorent la qualité du signal RPPG, les filtres conventionnels sont couramment utilisés. Cependant, certaines altérations demeurent, alors qu'un œil expérimenté peut facilement les identifier. Dans la première partie de cette thèse, nous proposons le réseau LSTMDF (Long Short-Term Memory Deep-Filter) pour réaliser le filtrage du signal RPPG. Nous utilisons différents protocoles pour analyser les performances de la méthode. Nous démontrons comment le réseau peut être entraîné efficacement avec un nombre limité de signaux. Notre étude démontre expérimentalement la supériorité du filtre basé sur le LSTM par rapport aux filtres conventionnels. Le réseau est ainsi peu sensible rapport signal/bruit moyen des signaux RPPG.

Les approches basées sur les réseaux convolutifs tels que les 3DCNN ont récemment surpassé les méthodes manuelles traditionnelles dans la tâche de mesure du RPPG. Cependant, il est connu que les grands modèles 3DCNN ont des coûts de calcul élevés et peuvent être inadaptés aux applications en temps réel. Comme deuxième contribution de cette thèse, nous proposons une étude d'une architecture 3DCNN, trouvant le meilleur compromis entre la précision de la mesure du pouls et le temps d'inférence. Nous utilisons une étude d'ablation où nous diminuons la taille de l'entrée, proposons une fonction de perte personnalisée, et évaluons l'impact de différents espaces de couleur d'entrée. Le résultat est le RPPG en temps réel (RTRPPG), un outil de mesure du RPPG de bout en bout qui peut être utilisé sur GPU et CPU. Nous avons également proposé une méthode d'augmentation des données qui vise à améliorer les performances des réseaux d'apprentissage profond lorsque la base de données présente des caractéristiques spécifiques (par exemple, les mouvements de type fitness) et lorsque les données disponibles sont peu nombreuses.

UBFC UNIVERSITÉ BOURGOGNE FRANCHE-COMTÉ
Université Bourgogne Franche-Comté
32, avenue de l'Observatoire
25000 Besançon, France

”Hay que disfrutar al máximo el
supremo placer de ser nadie.”

Mario Mendoza

ACKNOWLEDGMENTS

I wish to thank the members of my jury for agreeing to read the manuscript and to participate in the defense of this thesis. I am grateful to Hazem Wannous, Aymeric Histace, Antitza Dantcheva, and Norbert Noury for their constructive and supportive remarks and interest in this thesis.

Firstly, I would like to express my immense gratitude to my two supervisors, Yannick Benezeth and Johel Miteran, for their dedication in the development of my thesis. Without his support and knowledge of the subject, this manuscript would not have been possible.

I would also like to thank all the people in the laboratory who accompanied me during these three years and of whom I have very good memories. The smartest guy I know, Alpha Liu, Esteban Gaitan, Romain Cendre, Ramammorthy Luxman, Yuly Castro, Mathilde Vergnaud, Hermès McGriff, Rita Meziati, Sarah Leclerc and her cat Ori, Deepak Krishnamoorthy, and Siyu Lin.

I cannot forget to thank my closest friends who have always supported me from afar and who will always be in my heart. Juan Vavorro, Wilson Rengifo, Camilo Bermúdez, Andrés Chavez, Andrés Pachajoa, Victor Aros, Ginna Salas, and Ian Kingrey.

I thank all my family members, because it is only thanks to their support that I have been able to get to this point. This achievement is more theirs than mine. To my beloved mother Luz Monsalve, my super hero and father Jose Botina, and the two best brothers in the world, Jhon Botina, and Jabrini Botina.

Last but not the least, I am immensely grateful to my girlfriend Diana Marin, the most loving and supportive person I know, she has been my main support during all this time and I have no way to thank her enough for everything she has done for me.

CONTENTS

List of Definitions

BP: Bandpass filter
BSS: Blind source separation
Chrom: Chrominance-based RPPG
CNN: Convolutional neural network
CPU: Central processing unit
CV: Cross validation
dB: Decibels
ECG: Electrocardiography
EEG: Electroencephalography
EVM: Eulerian video magnification
fps: frames per second
G-R/GR: Green and Red image channels
GRU: Gated recurrent unit
GPU: Graphics processing unit
HR: Heart rate
HRV: Heart rate variability
ICA: Independent component analysis
LSTM: Long short-term memory
MAE: Mean absolute error
NN: Neural network
PbV: Blood-volume pulse vector
Pc: contact-based pulse rate
PCA: Principal component analysis
POS: Plane-orthogonal-to-skin
PPG: Photoplethysmography
Pr: remote-based pulse rate
PRV: Pulse rate variability
PR: Pulse rate
PVM: Pulse rate variability measurement
r: Pearson's correlation coefficient
RT: Real-time
RNNs: Recurrent neural networks
RoI: Region of interest
RPPG: Remote photoplethysmography
RPPG-SNR: SNR level in RPPG
SG: Savitzky-Golay filter
SNR: Signal to noise ratio
TMC: Template match correlation
WV: Wavelet filter
2CNN: Two-dimensional convolutional neural network
3CNN: Three-dimensional convolutional neural network

I

CONTEXT AND STATE OF THE ART

1

INTRODUCTION

Below, we present the context of this thesis in order to propose the objectives to be developed. Finally, we present the distribution of the content of this thesis.

1.1/ CONTEXT

Biomedical signals are any kind of biological or electrical signals that can be used to measure a bodily function or activity. They are important in medicine because they can be used to diagnose and monitor various medical conditions. There are various types of biomedical signals, including electrocardiograms, electroencephalograms, electromyograms, and heart rate variability signals. These signals can be measured using diverse methods, including electrodes attached on the skin, sensors placed inside the body, or special devices that measure blood pressure, heart rate, or other bodily functions. Biomedical signals are usually processed using computation techniques to extract information about the underlying medical condition. This processed information can be used to diagnose a disease, monitor a patient's progress, or even predict future health problems.

Electrocardiography (ECG) and photoplethysmography (PPG) are two methods that are used to measure different physiological parameters of the body, such as heart rate (HR) and heart rate variability (HRV). ECG is a method that measures the electrical field caused by heart activity. On the other hand, PPG measures variations in light absorption in tissues due to the pulsatile nature of the cardiovascular system and the variation in blood volume [25]. Heart rate monitoring can be conducted by invasive methods such as pulmonary artery catheterization [110], and non-invasive methods classified as contact-based and non-contact-based. ECG and PPG methods perform contact-based HR measurements, which may cause hygiene issues, discomfort, or even be impossible on fragile skins. Due to these possible disadvantages, in [9], Verkruysse *et al.*, demonstrated that PPG signals could be measured remotely from a standard video camera, using ambient light as an illumination source. This technique, known as remote photoplethysmography (RPPG), offers the advantage of measuring the same parameters as PPG in an entirely remote way. In fact, RPPG is the non-contact equivalent to the reflective mode of PPG, using ambient light as a source and a camera as a receptor. The light reflected by the skin is then estimated by capturing subtle skin color variations by the camera as blood volume changes.

Several biomedical parameters can be evaluated from RPPG or PPG signals, such as

pulse rate (PR), pulse rate variability (PRV), breathing rate (BR), vascular occlusion, peripheral vasomotor activity, and blood pressure by pulse transit time [7, 66]. Likewise, the applications are multiple, and some examples are mixed reality [47], physiological measurements of car drivers [70], living skin segmentation [49], control of vital signs in the elderly and newborns [76], and face anti-spoofing [92].

Initially, the general overview of video-based physiological parameters estimation methods was based on hand-crafted methods. First, face detection and tracking are performed within the scene. Then, a hand-crafted feature extraction technique is used to obtain the information present in the RGB channels, and a color channel combination algorithm generates the RPPG signal. Additionally, a filtering process is performed to extract only the frequencies related to the biological parameters. Finally, the physiological parameters are estimated by employing a temporal or frequency analysis. Figure 1.1 shows the described pipeline.

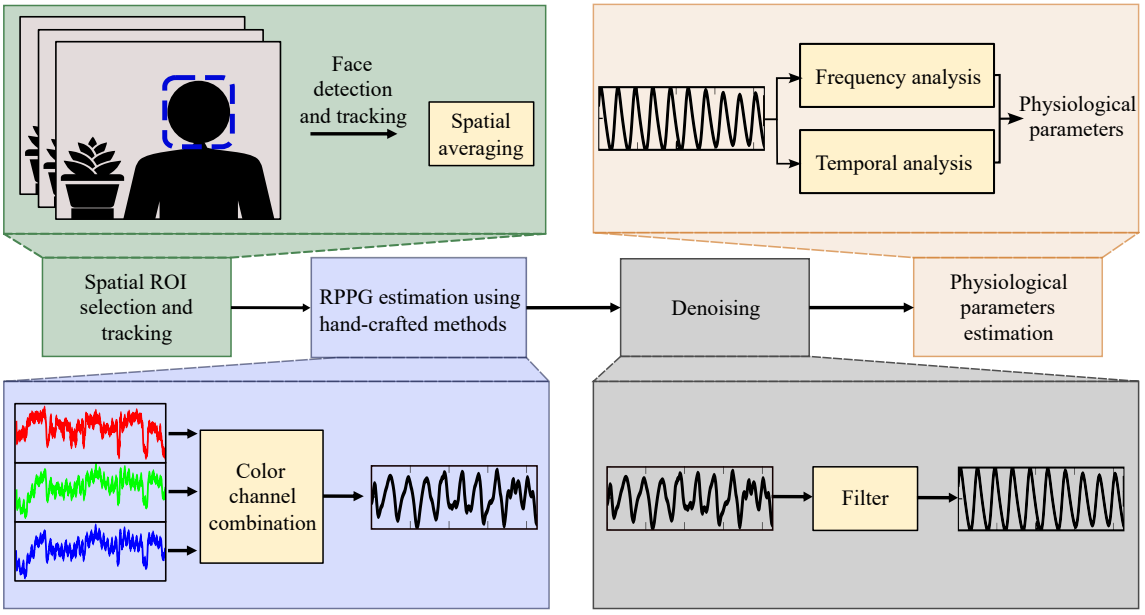

Figure 1.1: General overview of video-based physiological parameters estimation using hand-crafted RPPG measurement methods.

More recently, pipelines based on deep-learning methods have become relevant in this area. Deep-learning is a branch of machine-learning concerned with designing and developing algorithms that can learn from data that is too unstructured or too complex for traditional machine-learning methods. The most relevant deep-learning-based applications include computer vision, natural language processing, and time series analysis. Deep learning is particularly well suited for tasks that require the extraction of high-level features from data, such as biomedical images and signals. In the medical area, deep-learning is often used in computer-aided diagnosis (CAD) systems to automatically extract features from medical images and make predictions about a patient's disease state. Many different applications of deep-learning in CAD have been proposed, including cancer detection, detecting brain lesions, and cardiovascular abnormalities [94]. On the other hand, deep-learning has been used over physiological signals, such as electrocardiography, electrooculography, electroencephalography, electromyography, and photoplethysmography [55, 72]. Thus, employing physiological signals, interesting applications have been developed. Some examples include, limb movement estimation [68], movement in-

tention decoding [38], neuroprosthesis control [51], gesture recognition [35], hand movement classification [32], speech recognition [40], robotic arm guidance [31], and emotional state recognition [72].

Usually, the RPPG signals contain noise from the acquisition. Consequently, once the RPPG signals are acquired, unnecessary information such as frequencies out of the normal physiological range of interest are removed using a filtering process. The smoothing operation is commonly performed by a bandpass filter (BP) [22, 14, 11, 12, 10, 26], wavelet-based filter (WV) [60, 95, 16, 64, 50], and recently, by the Savitzky-Golay filter (SG) [52, 113, 111]. However, they do not necessarily remove particular signal alterations, which can, however, be easily identified by experts. This RPPG signal filtering problem could be exploited by deep-learning techniques, specifically, through recurrent neural networks.

Deep learning models have also been used to measure physiological parameters from video [84, 62, 82, 61]. Perhaps the most interesting frameworks are the end-to-end ones where the RPPG or PR are measured from video [65, 90, 91, 112, 119, 108, 112, 122, 108]. However, the current end-to-end frameworks have been implemented on high-performance GPUs. Thus, it is necessary to build upon previously proposed architectures while focusing on optimizing their inference speed for real-time applications (potentially on low-end devices).

We will now present the objective of this work, and the way in which this thesis will be developed.

1.2/ OBJECTIVE AND THESIS OVERVIEW

This thesis aims to advance research in remote photoplethysmography by developing new deep-learning-based models. Specifically, we are interested in proposing an end-to-end framework to measure RPPG signals whose architecture fits in real-time applications. We use relevant public databases that the scientific community has explicitly proposed to evaluate physiological data measurement techniques. Additionally, we want the techniques developed to improve the estimation of the pulse rate, but also the quality of the temporal signal and thus pave the way for using RPPG signals in breathing rate and pulse rate variability estimation. To this end, we will improve the RPPG filtering process focusing on pulse rate and signal quality improvement.

The remainder of this thesis is composed of five chapters. The first one, chapter 2 explains the basic concepts of contact and non-contact methods to measure ECG and PPG signals, RPPG hand-crafted methods, and finally, supervised deep learning.

In chapter 3, we list the main databases developed by the research community to measure physiological data. Then, we discuss the classical RPPG filters. Finally, we present the deep-learning-based frameworks used in the literature to measure physiological data. Thus, we explain the motivations for realizing a deep-learning-based RPPG signal filter and a deep learning network for acquiring RPPG signals from video in real-time.

Chapter 4 presents the evaluation protocols used to study the proposed experiments. Then, we list the databases and the pre-processing techniques applied to them. Finally, we present the metrics used to evaluate the performance of the proposed methods in pulse rate measurement, signal quality, and pulse rate variability.

In chapter 5, motivated by filtering RPPG signals, a deep filter based on LSTM networks called LSTMDF (Long-short term deep-filter) is presented. Two filtering approaches are proposed, the first based on a many-to-one architecture and the second based on a many-to-many architecture. Both architectures are compared with classical filtering methods.

In chapter 6, three-dimensional convolutional networks are used to propose a method for RPPG measurement. With an ablation study, we reach the optimal network configuration. The optimal architecture is called RTRPPG (Real-Time Remote Photoplethysmography). The RTRPPG network can infer RPPG signals from video in real-time on the GPU and CPU hardware used. Furthermore, its PR measurement and signal quality performance are comparable to the literature.

Finally, chapter 7 presents some general conclusions, perspectives, and future work from the experiments presented throughout this thesis.

2

BACKGROUND

This chapter presents the background about contact and non-contact methods to measure ECG and PPG signals, RPPG hand-crafted methods, and finally, supervised deep learning. The mathematical notation used in this chapter is independent of the following chapters. That is, do not confuse the variables used in this chapter with those used in future chapters.

2.1/ HEART RATE AND PULSE RATE MEASUREMENT

It is important to monitor the heart because it is responsible for pumping blood throughout the body. If the heart is not functioning correctly, it can lead to serious health problems. Doctors can measure heart rate by using their fingers to feel the pulse or using a stethoscope to listen to the heartbeat. ECG and PPG are the most used technologies for measuring heart rate. More recently, non-contact techniques such as RPPG have been proposed. In the following, we will present a background on these methods.

2.1.1/ ECG: ELECTROCARDIOGRAPHY

An electrocardiographic signal is an electrical signal generated by the heart during the cardiac cycle. An electrocardiograph measures the ECG signal from the movement in the heart by measuring the electrical potential difference between two electrodes placed on the body. The initial electrocardiograph version was invented by Willem Einthoven (late 19th century), where he used three electrodes placed on the left arm, the right arm, and the left leg. Thus, forming the Einthoven Triangle. Today, electrocardiographs (devices using the ECG technique) have a set of ten electrodes: next to the Eithoven electrodes, there is an electrode placed on the right leg (used for grounding); the four electrodes are called peripheral electrodes; as well as six electrodes called precordial electrodes, placed on the thorax. ECG recordings yield consecutive sequences, each composed of three main components: a P wave, a QRS complex, and a T wave. The P wave corresponds to the depolarization of the atria, the QRS complex is related to the depolarization of the ventricles, and the T wave represents the repolarization of the ventricles. In addition to the analysis of the P, QRS, and T components in terms of shape, amplitude, and duration, the durations of the PR and ST intervals are also assessed by healthcare professionals. From an ECG signal, the heart rate can be estimated by counting the number of QRS complexes per unit of time (usually one minute). Figure 2.1 shows an example of ECG

signal.

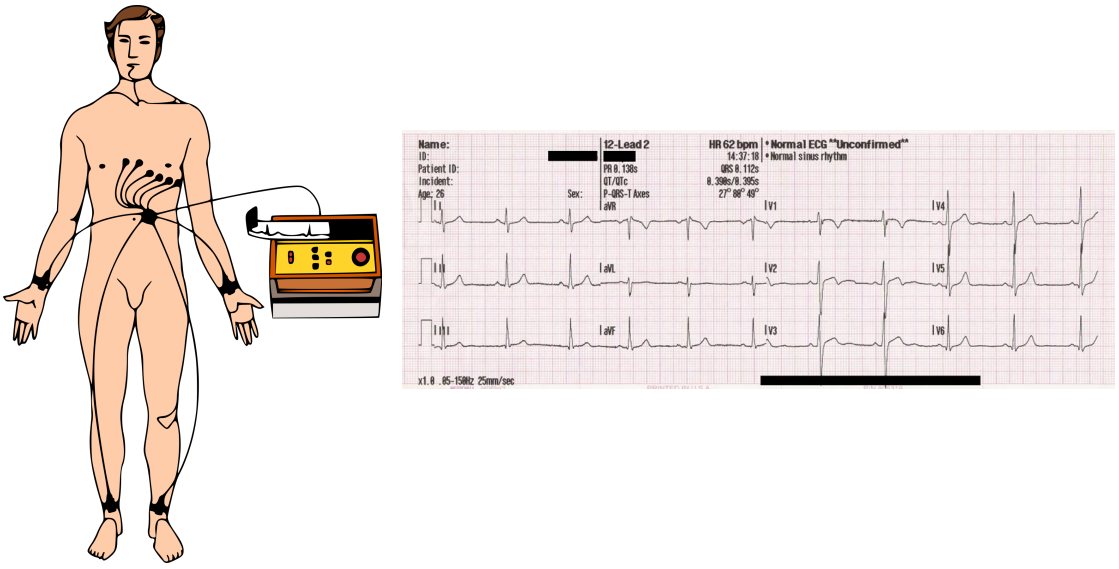

Figure 2.1: ECG measurement. An example of ECG signal. Credit: `wikimedia.org`.

2.1.2/ PPG: PHOTOPLETHYSMOGRAPHY

The change in blood volume due to cardiac activity can be perceived by pressing the artery against a bone with the fingers. This way, we can measure the pulse rate (PR) by counting the number of beats per minute (bpm). Note that we use the term heart-rate when we use the cardiac activity of the heart (ECG); we use the term pulse rate when we perceive the mechanics of the change in blood pressure.

The pulse rate is obtained from a pulse signal, or Blood Volume Pulse (BVP). BVP is a quasi-periodic signal consisting of successive pulses generated by the heart's pumping activity. The reference method for obtaining BVP is photoplethysmography, the technique used in pulse oximeters. The principle is to emit light with a light source and to quantify the light absorbed or reflected by the human skin through a light receiver. The blood volume in the tissues impacts the amount of light the skin absorbs or reflects. The most common PPG measurement sites are the fingertips, toes, or earlobes. The signal obtained by PPG consists of a variable component and a continuous component. The latter is related to respiration and thermoregulation, while the variable component contains pulsatile information about blood flow changes. Applications of PPG include monitoring of oxygen saturation, pulse rate, respiration rate, blood pressure, cardiac output, assessment of autonomic functions and detection of peripheral vascular diseases [9].

Figure 2.2 presents the waveform of ECG and BVP signals during the cardiac cycle. Both signals allow extraction of the heart rate. That is, the frequencies of both signals are highly correlated. Depending on the part of the body used to measure the BVP signal, there will be a greater or lesser offset between the BVP and ECG signals.

ECG and PPG measurement offer the advantage of being low-cost, non-invasive, and widely used in medical settings. However, the contact with the skin can destabilize some people or be difficult to apply to certain categories of patients. In response, some technologies have been developed to measure pulse rate remotely, for example, remote pho-

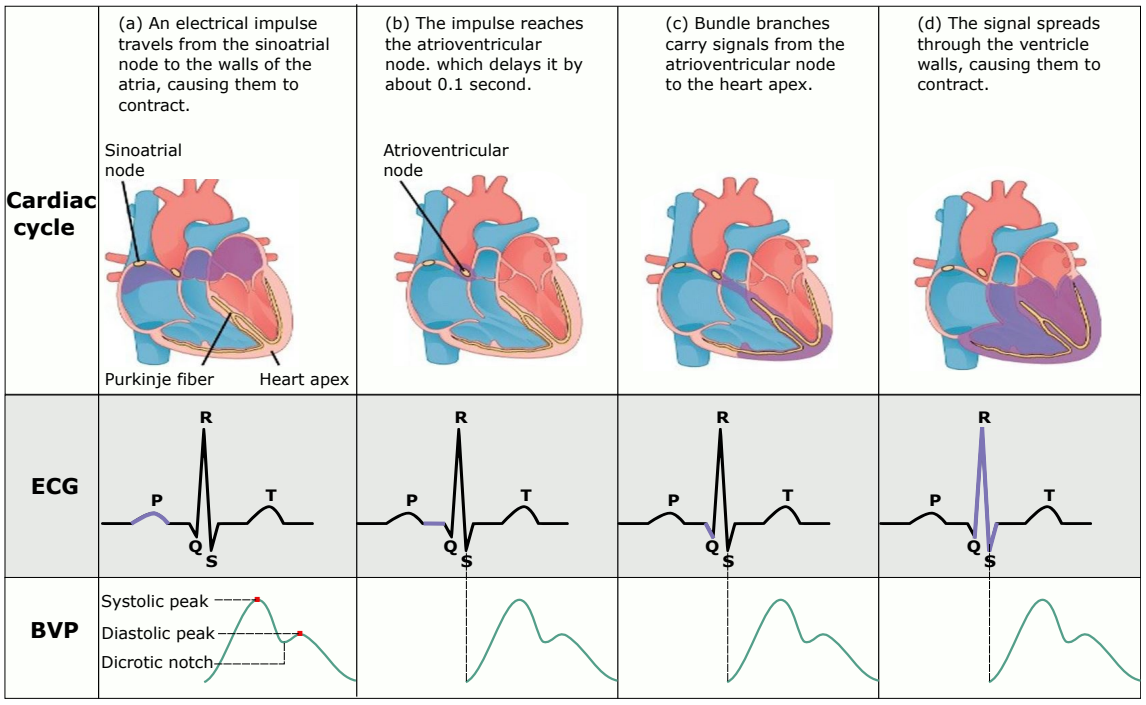

Figure 2.2: ECG and BVP waveform from cardiac cycle. Adapted from Karnan H. *et al*.[79].

toplethysmography.

2.1.3/ RPPG: REMOTE PHOTOPLETHYSMOGRAPHY

In 2008, Verkruysse *et al.* [9] proved that PPG signals could be measured remotely from a standard video camera, using ambient light as an illumination source. This technique, known as remote photoplethysmography (RPPG), offers the advantage of measuring the same parameters as PPG in an entirely remote way. RPPG is the non-contact equivalent to the reflective mode of PPG, using a camera as a receptor and ambient light as a source. Thus, blood volume changes are estimated according to subtle skin color variations, which are captured by the camera when lights are reflected by the skin. Remote photoplethysmography aims to ameliorate the problems faced by photoplethysmography, such as sensitivity to motion, single-point measurements, and the fact that it is contact-based.

Figure 2.3 shows how the BVP signal is acquired by the contact-based PPG method and the non-contact-based RPPG method. Both techniques allow to estimate physiological parameters such as pulse rate, breathing rate, and pulse rate variability.

In order to improve the RPPG signal quality and the PR estimation, approaches based on blind source separation techniques were proposed [12, 5]. Similarly, alternative methods based on a light tissue interaction model to determine a projection vector [20, 41, 18] were developed by the scientific community.

It is also worth mentioning that video-based ballistocardiography (BCG) is another remote method used to measure pulse rate. BCG is a procedure to obtain a graphical representation of repetitive movements of the human body. These movements arise from

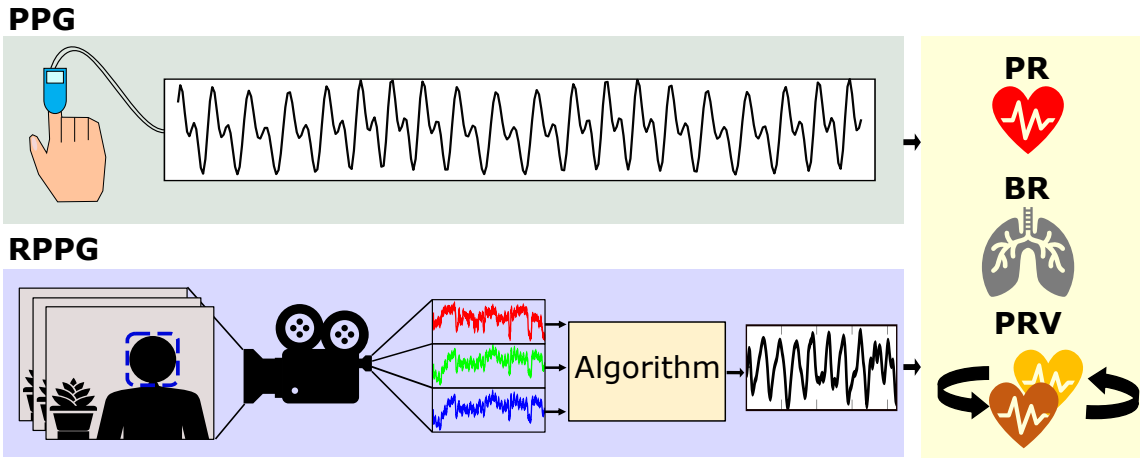

Figure 2.3: PPG vs RPPG. PR:Pulse rate, BR: Breathing rate, PRV: Pulse rate variability.

the sudden ejection of blood into the great vessels with each heartbeat [1]. For example, in [15], the authors take the minute motions resulting from the blood pulsations to amplify and quantify them to form a cardiac signal.

2.2/ RPPG HAND-CRAFTED METHODS

The regular pipeline for estimating RPPG signals consists of the following steps: 1) face tracking and cropping are performed within the scene 2) the relevant part of the face is selected, e.g., the whole face, only the skin, or regions of interest (RoI), usually located on the forehead and cheeks 3) spatial averaging is performed among all pixels for each color channel, resulting in three one-dimensional vectors, one for the red channel, one for the green channel, and one more for the blue channel. 4) Finally, a hand-crafted method is applied to the RGB signals, and the BVP signal (also called RPPG in this thesis) is obtained. Figure 2.4 shows the steps mentioned above.

The first approach implemented to estimate RPPG signals was proposed in 2008 by Verkruysse *et al.* in [9]. The authors conducted an experiment where they demonstrated the possibility of acquiring the PPG signal remotely. In their experiment, they used a regular camera and ambient light. The patient under study was asked to exercise to raise his heart rate and then stop. Five seconds later, the recording of the face was started. After about three minutes, the patient was already at his standard heart rate. Then, he was asked to inhale and exhale deeply for about a minute to lower his heart rate. Incredibly, all these heart rate changes were perceived by the camera analyzing the video on the patient's forehead. From the RGB channels, the G channel (*Green*) and the RG combination (*G-R*) were the more promising ways of measuring the patient's RPPG signal.

Eventually, approaches based on blind source separation techniques were proposed [12, 5], and others based on a light tissue interaction model to determine a projection vector [20, 41, 18]. In-depth state-of-the-art reviews of these and more RPPG signal estimation techniques are presented in [27, 44, 29].

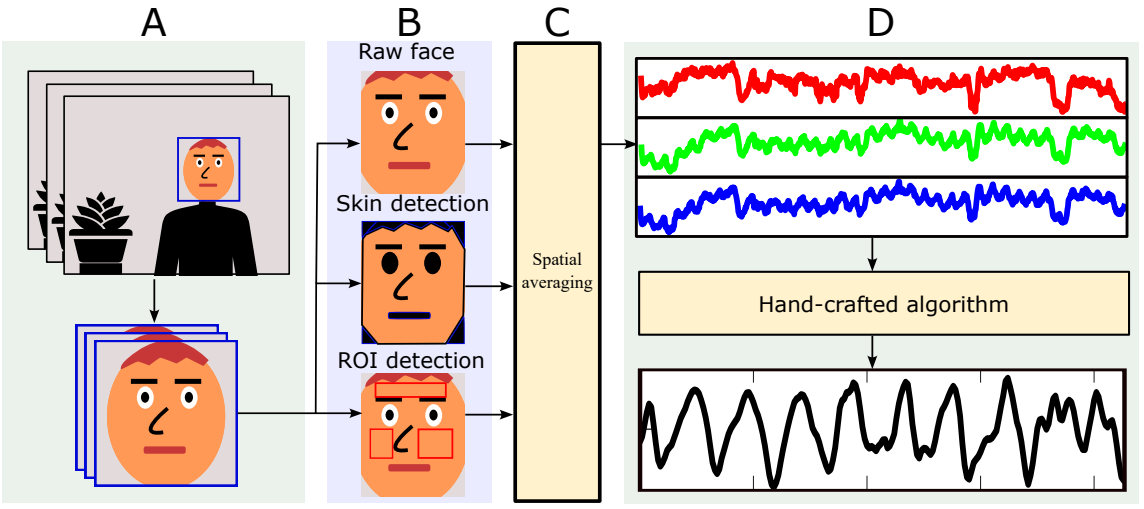

Figure 2.4: Regular pipeline in hand-crafted RPPG measurement methods. A) Face tracking and cropping B) Input selection C) Spatial averaging D) BVP estimation from RGB channels and a RPPG hand-crafted method.

2.2.1/ METHODS BASED ON CHROMINANCE

Some RPPG methods are based on the physical properties of the skin tissue while interacting with light. Techniques such as Blood-Volume Pulse vector (PbV) [20], Plane-Orthogonal-to-Skin (POS) [41], and Chrom [18] are based on specific skin characteristics. The main advantages of these methods are their computational simplicity. These methods use light's effect on human skin due to its distinctive physical properties resulting in specific absorption and reflection.

2.2.2/ SUBSPACE-BASED METHODS

Subspace-based methods exploit the relationship between the RGB and BVP signals to define a subspace representing a criterion that is eventually based on the physical characteristics of the skin when interacting with light. Compared with chrominance methods, subspace-based methods do not use models based on the skin properties.

These methods can be considered analogous to a change of basis, where the projections of the original RGB traces amplifies the RPPG information, *i.e.*, it intensifies the underlying BVP signal. The Spatial-Subspace Rotation (2SR) is a representative technique in this category. 2SR formulates that the periodic variations of the skin color reside in the temporally varying subspaces. An interesting aspect of this formulation is that the skin pixel subspace is not necessarily based on the skin tone.

Another interesting technique proposed in [59] is Periodic Variance Maximization (PVM). PMV uses an enhanced subspace-based decomposition procedure and an optimization algorithm to exploit the periodicity criterion embedded in the RGB temporal traces. The PVM algorithm tries to find the unknown period of the RPPG signal by combining two different approaches. The first approach is the iterative subspace decomposition procedure, which estimates a periodicity maximizing basis for a given frequency. The second approach is a global optimization algorithm of Tabu search, which tries to find the frequency with the highest global periodicity over the search space.

2.2.3/ BLIND SOURCE SEPARATION METHODS

Blind source separation (BSS) is a statistical signal processing technique used to separate independent sources from a set of observations. In the RPPG measurement context, in the RGB color channels exist BVP, illumination and noise perturbations. BSS techniques are used to extract the BVP while removing all kind of noise. Independent Component Analysis (ICA) is perhaps one of the most used hand-crafted methods [6]. The hypothesis to extract BVP from the RGB channels is based on that the original cardiac pulse signal is linearly merged with all kind of perturbations. ICA computes the signal separation by maximizing metrics of independence such as mutual information and non-gaussianity.

Another BSS technique used in RPPG measurement is Principal Component Analysis (PCA) [12, 5]. PCA is a statistical technique used to reduce the dimensionality of data. The goal of PCA in RPPG measurement is to find a subspace in which the covariance between the RGB channels is maximized, extracting the most significant component which is supposed to be the BVP signal.

Methods based on chrominance, subspaces, and BSS have been helpful, and research is still ongoing. However, more recently, methods based on deep learning have improved the pulse rate measurement results. Thus, in chapter 3 we will present a detailed revision of the main deep-learning-based frameworks for physiological data measurement. In order to understand deep-learning-based frameworks, an introduction to the operation of neural networks in supervised learning is presented below.

2.3/ SUPERVISED DEEP LEARNING

Deep learning is a branch of machine learning in the artificial intelligence field (Figure 2.5). Deep learning is concerned of designing and developing algorithms that can learn from data that is too unstructured or too complex for traditional machine learning methods.

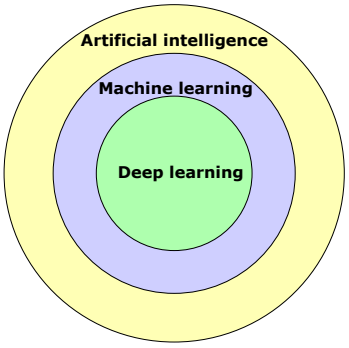

Figure 2.5: Deep learning in the artificial intelligence field. Credit: `wikimedia.org`

Supervised deep learning refers to a training approach where a neural network is given an input and its ground truth value. In this way, the training is supervised by checking that the output is as similar as possible to the ground truth value.

2.3.1/ NEURAL NETWORKS (NNS)

A neural network (NN) is a system of interconnected neurons that process information by propagating signals from input neurons to output neurons. The parts of a neural network include input nodes, hidden nodes, and output nodes (Figure 2.6).

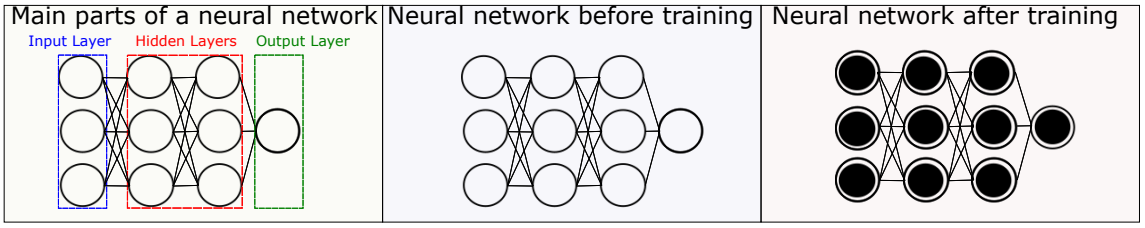

Figure 2.6: Neural networks. The black color inside the neurons indicates the level of knowledge about the context of the task.

Let us consider a single example (x, y), with x the input, and y the ground truth in a classification problem. Now, consider multiple examples in a training set M_{train} where $\mathbf{M}_{train} \in \{(x^1, y^1), (x^2, y^2), (x^3, y^3), ..., (x^m, y^m)\}$. We can represent all input examples as $\mathbf{X}_{train}$ and their ground truth as $\mathbf{Y}_{train}$. The learning process of a neural network is to use the input $\mathbf{X}_{train}$ to generate the output $\hat{\mathbf{Y}}_{train}$, and minimize the error between $\hat{\mathbf{Y}}_{train}$ and $\mathbf{Y}_{train}$. The goal of the neural network after training is to be robust enough to know the context of the task, and to be able to make accurate predictions on a new data set $\mathbf{M}_{test}$. There are mainly three types of neural networks: feedforward neural networks, convolutional neural networks, and recurrent neural networks.

The internal procedure of one single neuron is presented in Figure 2.7. Each component x_k of feature vector $\mathbf{X}$ is multiplied by the weights $\mathbf{W}$. Then, the output of the neuron is computed as follows $\hat{y} = s(\mathbf{W}^T\mathbf{X} + b)$, where s is the activation function. Let us consider the single neuron as the complete neural network, thus, the error is computed through a loss function $\mathcal{L}$ measured between the output $\hat{y}$ and y. $\mathbf{W}$ and b are learnable parameters.

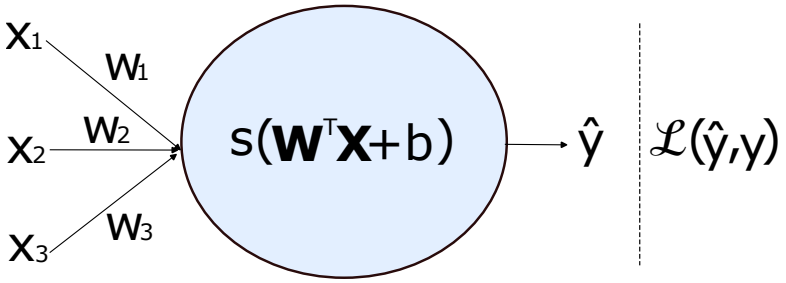

Figure 2.7: One neuron internal process. $\mathbf{X}$ input, $\mathbf{W}$ weights, b bias, s activation function, $\hat{\mathbf{y}}$ prediction, $\mathbf{y}$ ground truth, and $\mathcal{L}$ loss function.

In a neuron, the weights determine how much each input contributes to the output. The bias, how much the output will be offset from the input. The activation function provides a signal indicating the neuron activation level. Non-linear activation functions allow the model to learn complex representations to solve non-linear problems. The most commonly used activation functions are: Sigmoid or Logistic, hyperbolic tangent (*tanh*), and Rectified Linear Unit (*ReLU*), Figure 2.8 depicts these activation functions.

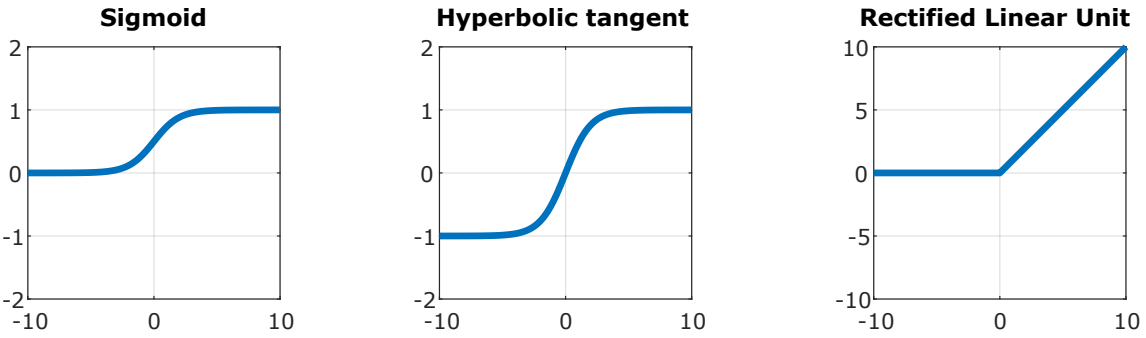

Figure 2.8: Commonly used activation functions.

2.3.2/ FEEDFORWARD NEURAL NETWORKS (FNNS)

The feedforward neural network (also called standard NN or fully-connected) is the simplest type of neural network. The FNN is composed of a series of fully-connected layers, and as the name suggests, in a FNN, the input data is fed through the network in the forward direction. Each hidden layer accepts the input data, processes it according to the activation function, and passes it on to the next layer. In order to generate output, the input data must only be fed in the forward direction (Figure 2.9).

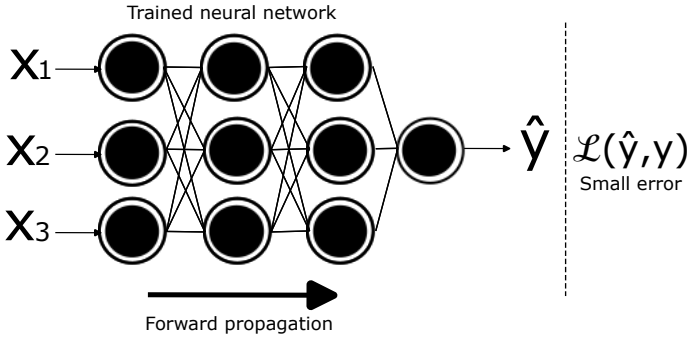

Figure 2.9: Forward propagation. The black color inside the neurons indicates the level of knowledge about the context of the task.

Backpropagation is the process of training a neural network from the output of the forward propagation [3]. Specifically, three fundamental tools are needed to perform backpropagation: a cost function, an optimization algorithm, and a learning rate. A cost function measures how far the actual output is from the desired output. It is based on the error measured by the loss function when changing the weights and biases of the network. An optimization algorithm finds the minimum value of the cost function. The most commonly used algorithm is the gradient descent [39]. A learning rate controls how fast the model adapts to the problem; it indicates in what proportion the weights and biases are updated. Figure 2.10 shows the backpropagation procedure, starting from a network implemented from scratch until it is fully trained, i.e., with a good knowledge of the problem context.

2.3.3/ CONVOLUTIONAL NEURAL NETWORKS (CNNS)

Convolutional neural networks (CNNs) are a type of neural network designed to mimic how the human brain processes visual information. In traditional machine learning, a

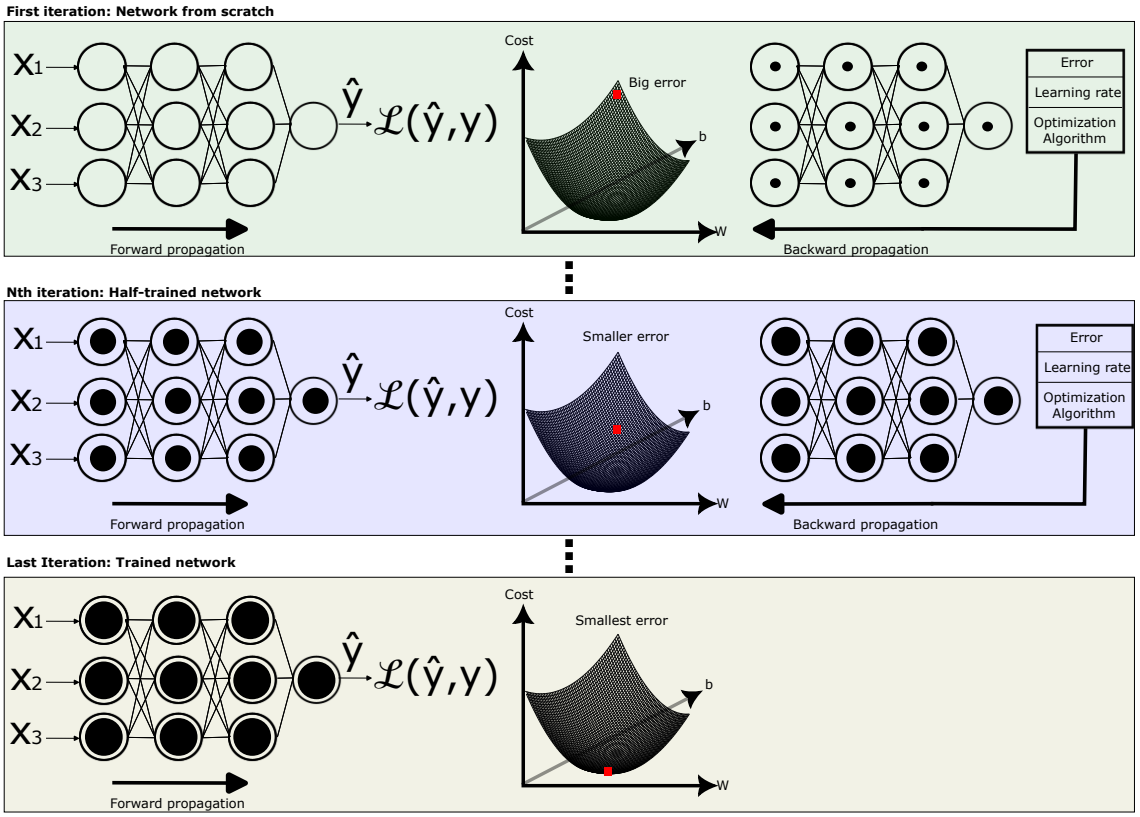

Figure 2.10: Neural network learning process. The black color inside the neurons indicates the level of knowledge about the context of the task.

feature vector is extracted from a signal (for example, an image), representing high-level information, such as texture, dimensional information, etc. Classification performance depends on the choice of this feature vector which is problem-dependent. CNNs were designed to directly process low-level (pixel-level) information without requiring a manual definition of a feature vector. The training process will automatically build the feature vector inside the CNN structure. Some applications of CNNs include image recognition, object detection, and video processing [33]. CNNs use convolutional layers to extract features from images, followed by pooling layers to down-sample the input. (Figure 2.11).

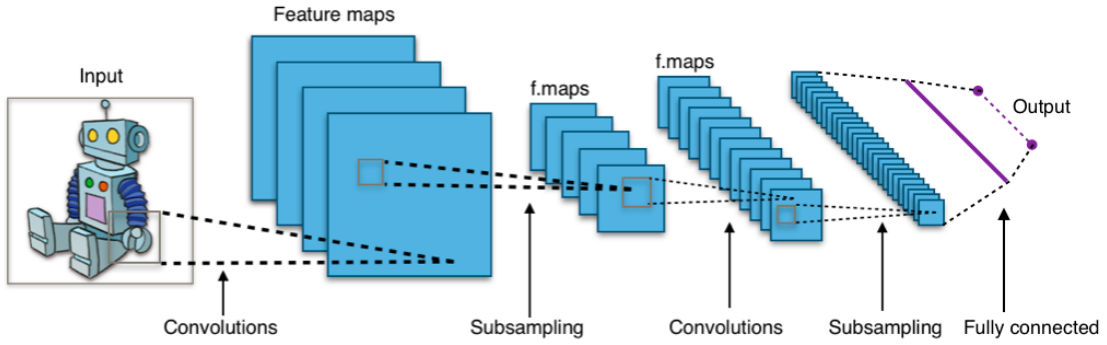

Figure 2.11: Convolutional neural network architecture. Credit: `wikimedia.org`

The convolution process is composed of a kernel, padding algorithm, and stride parameter. The kernel is the convolution matrix (generally, 3x3 or 5x5) that applies filters to the

input image. The stride parameter is the number of pixels the kernel moves over when using the filter. The padding process consists in adding pixels on the border of the image in order to avoid border effect during convolution process. The convolution process multiplies each element in the kernel by the corresponding elements of the input. The results are then added together and divided by the number of elements in the kernel. This process is repeated for every pixel in the image (Figure 2.12). Multiple convolutional layers allow sparse connectivity and parameter sharing. The deeper layers are indirectly linked to a larger receptive field, allowing for more abstract feature learning.

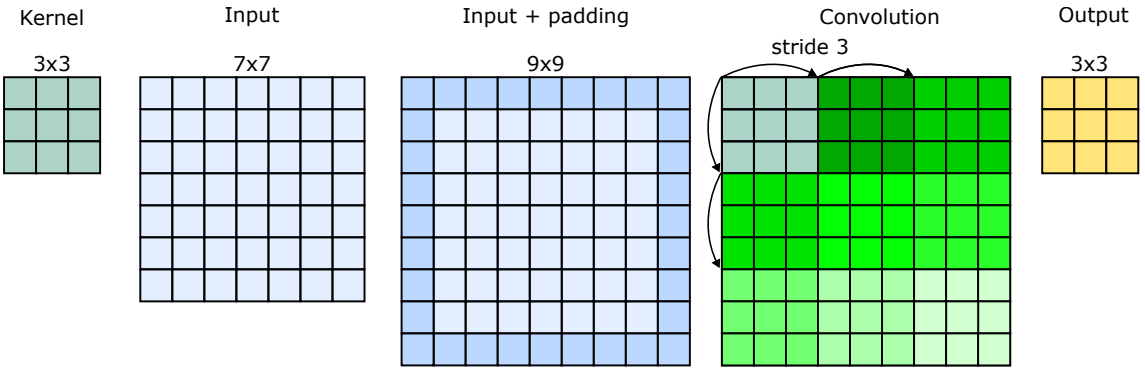

Figure 2.12: Convolution process.

Two-dimensional convolutional networks (2DCNNs) can only learn about relationships between features in two dimensions, i.e., 2DCNNs cannot learn temporal characteristics. On the other hand, three-dimensional neural networks (3DCNNs) don't have this problem. 3DCNNs can capture spatial and temporal information from multiple frames, allowing the network to be more effective at recognizing objects in video data (Figure 2.13).

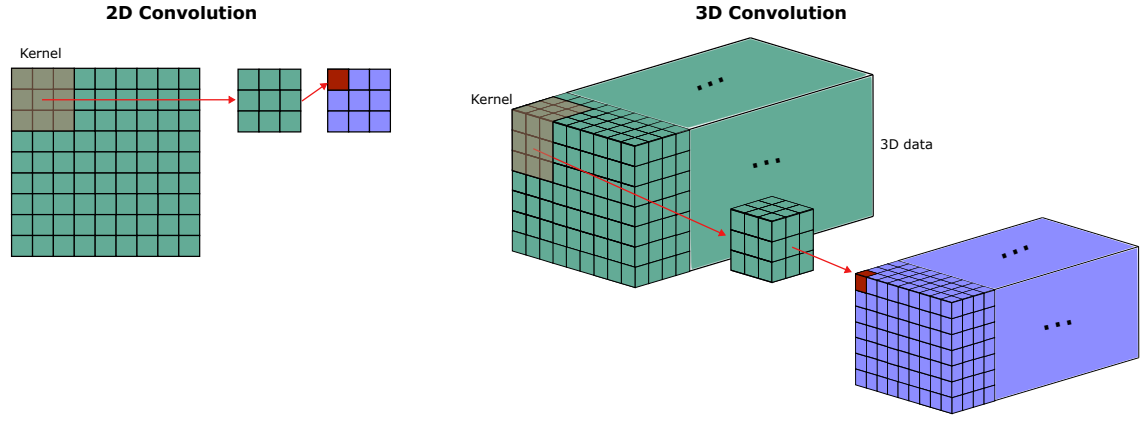

Figure 2.13: 2D vs 3D convolution. Adapted from [89]

2.3.4/ Recurrent neural networks (RNNs)

A sequence-to-sequence model is a learning model that takes a sequence of items as input and outputs a sequence of items. There are three principal types of sequence-to-sequence models: one-to-many, many-to-one, and many-to-many (Figure 2.14). A recurrent neural network (RNN) is a type of neural network that is used to model sequential data. RNNs are often used to model time series data, such as stock prices or weather

data. RNNs are important when using sequential data because they are able to capture patterns that may be missed by FNNs. RNNs are also able to handle variable-length inputs.

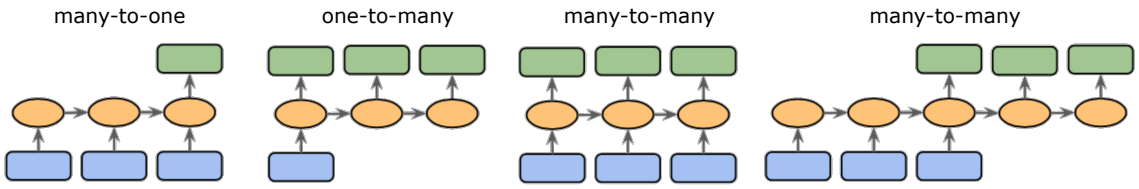

Figure 2.14: Main sequence-to-sequence models. Adapted from [86]

RNNs have been proposed as a way to share parameters for several computations within a sequence, in order to solve problems such as forecasting the movement of an object. FNNs could also forecast the movement by considering all previous examples as the input of the neural network. However, this approach is computationally expensive and inefficient due to the large number of parameters required. Often, sequences represent temporal data where each example within the sequence corresponds to a time t. Let us consider a simple RNN model with x an input sequence with size of T_x. $x \in [x^{<1>}, x^{<2>}, ..., x^{<t>}, ..., x^{<T_x>}]$, and $y \in [y^{<1>}, y^{<2>}, ..., y^{<t>}, ..., y^{<T_y>}]$ of size T_y the model's output. In Figure 2.15, we present the rolled and unrolled graphical versions of a simple RNN model where a shares the context through time steps. This way, the RNN can learn temporal features.

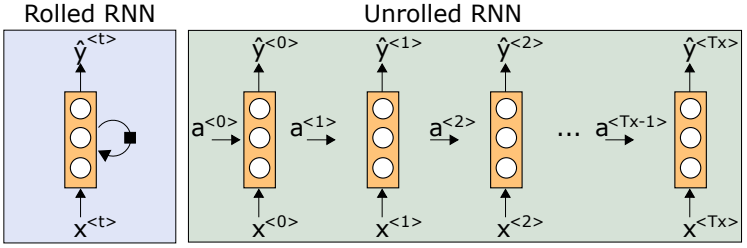

Figure 2.15: Simple RNN process. Rolled graphical version (left) and unrolled version (right).

RNNs show a clear advantage in learning temporal features compared to FNNs. However, their optimal performance is on short-term temporal features. After some time steps, the vanishing or exploding gradient arises. Gradient exploding is when the gradient gets so large that it "explodes" and causes the model to diverge. Vanishing gradient is when the gradient becomes so small that it "vanishes" and the model can no longer learn. To mitigate the exploding gradient, the gradient clipping technique is used. Basically, we clip the gradients during backpropagation so that they never exceed some threshold. Solving the vanishing gradient problem is a more complicated task. As an alternative, the long-short term memory LSTM was proposed [4]. LSTM networks solve the vanishing problem by using a memory cell c to store information, and a multiple gates to control what information is stored or forget in the cell. Figure 2.16 depicts the internal configuration of an LSTM unit.

The memory block consists of: one *update gate* Γ_u that learns which information should be stored in the memory block, a *forget gate* Γ_f that learns how much information must be forgotten or withheld from the memory block, and an *output gate* Γ_o that takes care of learning when the collected information can be used. Each gate has their respective

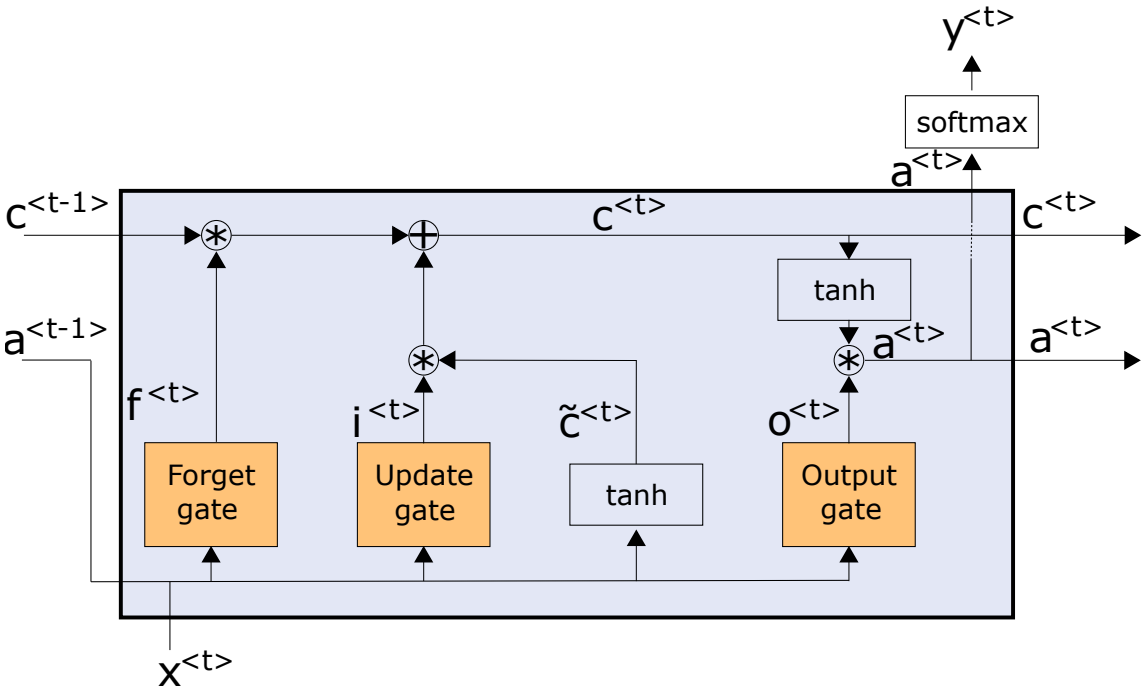

Figure 2.16: LSTM internal procedure.

weights and biases that will be optimized during the training procedure. We present the equations related to the LSTM in 2.1:

$$\begin{aligned}
\tilde{c}^{<t>} &= \tanh(W_c[a^{<t-1>}, x^t] + b_c) \\
\Gamma_u &= \sigma(W_u[a^{<t-1>}, x^t] + b_u) \\
\Gamma_f &= \sigma(W_f[a^{<t-1>}, x^t] + b_f) \\
\Gamma_o &= \sigma(W_o[a^{<t-1>}, x^t] + b_o) \\
c^{<t>} &= \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>} \\
a^{<t>} &= \Gamma_o * \tanh(c^t)
\end{aligned} \tag{2.1}$$

where $\tilde{c}^{<t>}$ is the cell state candidate containing information from the current time step that may be stored in c. The parts of $\tilde{c}^{<t>}$ that get passed on depend on Γ_u.

Note that LSTM is not the only recurrent network that solves the vanishing gradient problem, the Gated Recurrent Unit (GRU) is another alternative proposed in 2014 [19]. A GRU model is a recurrent neural network that uses gated units to update its hidden state, whereas an LSTM model uses gates to control the flow of information in its cells. GRUs have one gate less than LSTMs, which reduces matrix multiplication. However, in scenarios different than long text in small datasets, LSTMs overperforms GRUs [107]. Thus, an LSTM network could be the ideal recurrent network to work with sequences that require preserving long-term features.

2.4/ SUMMARY

In this chapter, we presented an introduction to how ECG and PPG signals are measured, followed by the non-contact version of PPG, RPPG. Then, we discussed the main RPPG hand-crafted methods used in the literature. Finally, we explained the basic concepts of supervised deep learning that will be used in this thesis. The following chapter presents state of the art on RPPG signal filtering methods and deep-learning-based frameworks used to measure RPPG and PR.

3

STATE OF THE ART

This chapter presents databases developed by the research community to train models or evaluate methods of physiological data estimation. Then, we discuss the conventional pipeline for video-based physiological parameter estimation, and some problems encountered with the classical RPPG filters. Then, we build the deep-learning based pipeline to measure physiological data based on related works. Finally, we analyze the end-to-end frameworks used in RPPG measurement and discuss the next steps for these methods.

3.1/ DATABASES USED IN PULSE RATE MEASUREMENT

In recent years, using public databases in medical applications has become common. Currently, there are multiple physiological signal databases presenting videos, ECG, EGG, and PPG signals. In this thesis, we are interested in databases used in remote photoplethysmography. Table 3.1 presents some of the databases used in the literature to measure RPPG. Initially, the first RPPG databases were developed for sentiment classification applications [13, 42]. Later, more specific databases were created whose goal was to extract the BVP signal from video in simple scenarios [24, 45, 74]. Eventually, databases for classifying medical issues such as atrial fibrillation in OBF appeared [57]. More complex scenarios were proposed with light changes, sporadic movements, and different recording devices, including NIR videos [62]. Similarly, we can find videos of subjects exercising involving a wide range of pulse rate values and movement in the scene [65].

Many of the databases used in pulse rate measurement have recently been released to the public. In addition, the databases are getting larger and more significant to meet the amounts of data needed to train deep learning models. Recent works use these databases to assess their RPPG and PR measurement methods. In this way, many authors have compared their results with those of other papers in the same databases. However, in 2020, Mironenko *et al.* [101], demonstrated that many factors had not been considered in making these comparisons. Perhaps the most important of these factors is to consider the size of the temporal window for PR calculation. Thus, it is difficult to compare methods even within the same database. To compare different RPPG approaches, it is necessary to specify the size of the temporal window. In the following section we will discuss the most relevant works in the literature related to RPPG filtering and RPPG measurement.

Table 3.1: Common databases used in RPPG measurement.

Database	No.Subjects	Illumination	Activity	Color space	No.Recordings
MAHNOB-HCI [13]	27	Natural	Stable	RGB	527
PURE [24]	10	Natural	Stable	RGB	60
MMSE-HR [42]	40	Natural	Stable	RGB	102
UBFC-rPPG [74]	49	Natural	Stable	RGB	50
PFF [46]	13	Natural/Dark	Stable/Motion	RGB	104
COHFACE [45]	40	Natural/Bright	Stable	RGB	160
OBF [57]	106	Natural	Stable	RGB/NIR	2120
VIPL-HR [62]	107	Natural/Dark/Bright	Stable/Motion/Talking	RGB/NIR	3130
VIPL-HR2 [98]	500	Natural/Dark/Bright	Stable/Motion/Talking	RGB/NIR	2500
ECG-Fitness [65]	17	Natural/Dark/Bright	Stable/Talking/Exercising	RGB	119

3.2/ RPPG FILTERING

Usually, the RPPG signals are noisy due to the estimation technique, illumination variations, internal noise of the digital camera, and motion. Consequently, once the RPPG signals are acquired, unnecessary information such as frequencies out of the normal physiological range of interest are removed using a filtering process. The smoothing operation is commonly performed by a bandpass filter (BP) [22, 14, 11, 12, 10, 26], sometimes by a wavelet-based filter (WV) [60, 95, 16, 64, 50], and recently, by the Savitzky-Golay filter (SG) [52, 113, 111]. Although these methods do smooth the RPPG signals, they do not necessarily remove particular signal alterations, which can, however, be easily identified by experts. Figure 3.1 shows the conventional pipeline used in the literature. The RPPG signal extracted from the video is smoothed through a classical filter for subsequent estimation of physiological parameters. However, even after the filtering process, irregular shapes of the RPPG signal are observed (see signal parts in green boxes in Fig. 3.1). The remaining alterations in the RPPG signal may affect the accuracy of heart rate measurements, but more gravely, avoid further advanced analysis of RPPG signals that can be based on peaks detection and pulse shape characteristics on temporal signals. For example, authors in [53] measured HRV by estimating the time elapsed between consecutive peaks of an RPPG signal to estimate emotional states. In this particular application, a noisy RPPG signal with false peaks would lead to erroneous measurement of HRV and, consequently, a misinterpretation of the emotional state. Therefore, there is a need to improve the accuracy of heart rate measurement and the RPPG signal quality.

LSTM networks have been used successfully in the literature to process medical signals such as ECG, proving the potential of this type of network [69, 63, 67]. In [56], the authors propose a method of PPG denoising based on a bidirectional recurrent denoising autoencoder (BRDAE) to retain the recurrent information in the PPG signal. The network training and testing are performed on an artificial noise-augmented PPG database, along with additional PPG signals acquired from subjects during their daily routine.

Slapnicar *et al.*[88], propose an LSTM based method to enhance reconstructed RPPG signals obtained by the POS method. For this purpose, they use a bandpass filter and a two-step wavelet filter to finally use a 2-layer LSTM network, using a many-to-one sequence-to-sequence approach. Although the results are satisfactory, using two clas-

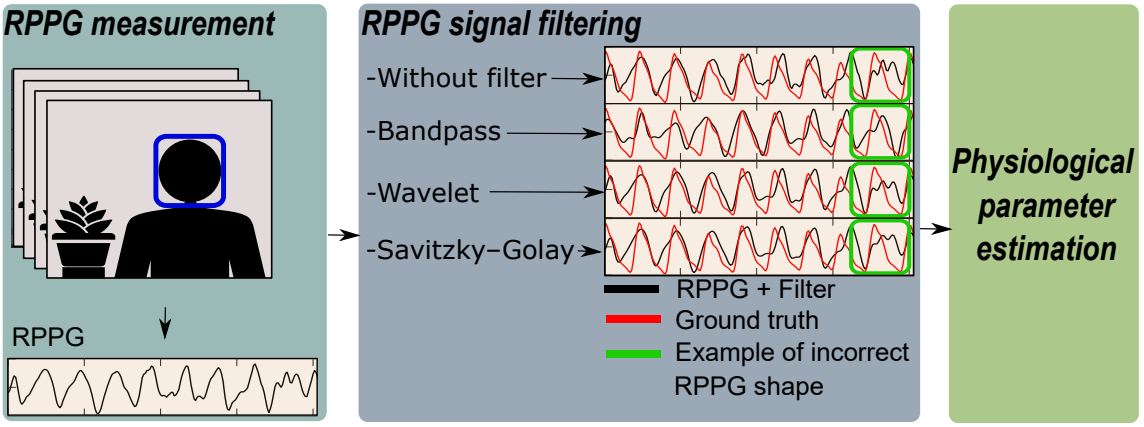

Figure 3.1: Conventional pipeline for video-based physiological parameters estimation. The RPPG signal is acquired from a video then a classical filtering method is used. The physiological parameters are calculated from the filtered signal. After the filtering, some irregular shapes of the signal remain (Ground truth is the Blood Volume Pulse (BVP) signal measured with the CONTEC CMS60C pulse oximeter).

sical filtering methods before using LSTM networks is necessary, when a single LSTM filtering stage could be sufficient.

One of the methods used in the literature to filter RPPG signals directly with an LSTM-based filter is the one proposed in [73], where the authors proposed a two-layer LSTM model to filter the RPPG signals in the MMSE-HR [42] database. The authors affirm that a deep neural network requires thousands of training data, and their way of facing this problem is to train the network in synthetic signals based on sine signals with random Gaussian noise and then fine-tune the network in the database. However, although the proposed synthetic signals manage to simulate the RPPG signals with frequencies corresponding to the heart rate, they do not contain the characteristic shape of the signals in a real acquisition scenario, preventing the network from learning the subtle details of the signal such as saving the dicrotic notch shape or suppressing double peaks as described in [77]. Table 3.2 shows the publications mentioned above organized according to the type of signal and the method used for their respective objective.

Table 3.2: Publications related to classical and machine learning methods for filtering ECG, PPG, and RPPG signals.

Publications	Signal	Method	Objective
[22, 14, 11, 12, 10, 26]	RPPG	Bandpass	Filtering
[95, 60, 16, 64, 50]	RPPG	Wavelet	Filtering
[113, 52, 111]	RPPG	Savitzky-Golay	Filtering
[69, 63, 67]	ECG	LSTM	Classification
[56]	PPG	BRDAE	Filtering
[88]	RPPG	Bandpass+Wavelet+LSTM	Enhance reconstructed RPPG
[73]	RPPG	LSTM	Filtering

In summary, these previous works filter PPG or RPPG signals with different approaches,

where the authors agree that it is necessary to obtain these signals with the best possible quality. Although these works already generate considerable progress in extracting quality RPPG signals, there is still room for improvement. In addition, despite the excellent performances of biomedical signal filtering techniques based on machine learning (e.g. [93], [56]), they are still rarely used compared to more classical techniques (like bandpass, etc.). We believe that this is due to the lack of studies investigating the advantages, limitations, and sensitivity of these methods. There is no in-depth study on the specific aspect of neural-network-based filtering, namely the influence of amount of training data, influence of input signal quality and influence of dataset used for training and testing. Therefore, in chapter 5 we will propose a deep-learning-based approach to filter RPPG signals.

In the following section, we will discuss the deep-learning-based solutions found in the literature to measure RPPG and PR from video.

3.3/ PHYSIOLOGICAL DATA MEASUREMENT WITH DEEP LEARNING

With the advent of the machine and deep-learning algorithms, a new field of research was opened to acquire RPPG signals [82, 75, 102, 91, 65, 96, 112, 100]. The main advantages of these methods are that they allow achieving good results without the need for the designer to analyze the problem in-depth [109]. The hand-crafted-based pipeline needs to detect and track the region of interest through the frames, combine color channels, filter them and estimate the physiological parameters such as respiration rate or pulse rate. Alternatively, in the deep-learning-based pipeline, such a series of steps are no longer necessary. Therefore, deep-learning-based approaches are less prone to error propagation in their pipeline. Nevertheless, recent work has focused on pulse rate measurement performance rather than understanding [109]. Subsequently, the limitations of the system are not always clear. Besides, it is well known that the training dataset used is critical.

3.3.1/ REMOTE PR AND PPG ESTIMATION

The works in the literature that use deep learning to measure physiological parameters from video can be divided into two groups. The first group measures the pulse rate value directly [15, 80, 83, 118, 96, 62, 82, 108, 61, 75, 105] (remote PR estimation), and the second measures RPPG, and then, as an additional step, calculates PR, BR, or PRV. [54, 65, 84, 97, 90, 91, 99, 106, 112, 122, 109, 119, 102]. Figure 3.2 presents the overview of these two groups.

In the first group (remote PR estimation in Figure 3.2), the input is a video, and the output is the pulse rate value. Thus, it is a straightforward estimation since no external process is needed to measure the pulse rate. However, the significant disadvantage of this approach is that the BVP signal is not available. Therefore, it is impossible to estimate any additional physiological parameter like pulse rate variability or breathing rate. PR estimation is normally considered as a regression problem [15, 83, 118, 96, 62, 82, 108, 61, 105]. However, there are also approaches where PR is estimated as a classification problem, where the classes refer to different pulse rate values. In [75], Bousefsaf *et al.* acquire PR from synthetic videos using a CNN as a feature extractor, the final activations feed a multi-layer perceptron to classify pulse rate with values between

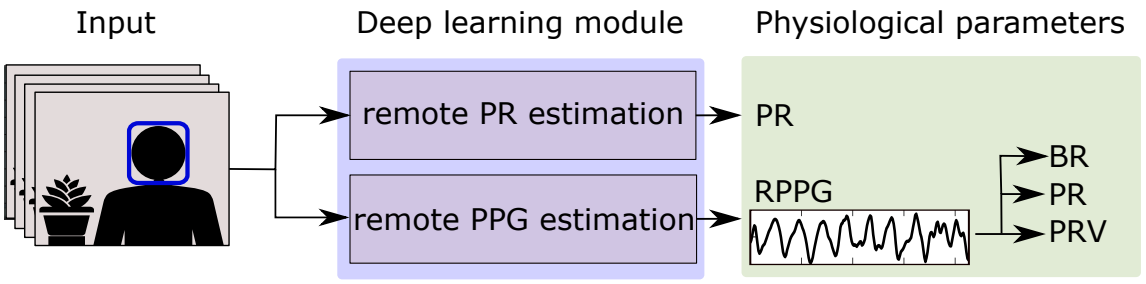

Figure 3.2: Remote physiological parameter estimation using deep learning. Remote Pulse rate estimation vs RPPG.

55 and 240 bpm. More interestingly, in [80], Kopeliovich *et al.* compare a regression approach with a classification one with pulse rate values between 40 and 125 bpm. Their results show a better pulse rate estimation performance with the classification approach.

The second group (remote PPG estimation in Figure 3.2) needs another step to measure the pulse rate value. Nevertheless, as we have mentioned before, having the RPPG signal, it is possible to estimate pulse rate variability and breathing rate. Consequently, it is also possible to evaluate additional applications such as atrial fibrillation classification.

Similarly, there are methods where the deep learning module is subdivided in different branches to measure physiological parameters. This is the case of the HR-CNN network proposed by Špetlík *et al.* in [65]. In this work, the two-step convolutional network has an *Extractor* and *HR Estimator* modules. First, a video is taken to detect and resize the faces, and then, the cropped video is passed through the *Extractor* to acquire its RPPG component. Finally, the RPPG is the input in the *HR Estimator*, and its HR is the output.

3.3.2/ Deep-learning-based pipeline

We found multiple works that measure physiological parameters from videos in the literature. All of them share the following characteristics: they choose a color space and use a spatial module and a temporal module. In Figure 3.3, we present the deep-learning-based pipeline in physiological data measurement based in the literature. Depending on the spatial and temporal module, 2DCNNs, RNNs, or both are used. Although some authors consider that computing a spatial-temporal map before using a CNN is an end-to-end framework [82, 62, 61, 15], we believe that an end-to-end framework should be as straightforward as possible, i.e., the neural network should be able to estimate physiological data from the input video, measuring the spatial-temporal context without additional processes. Thus, using only 2DCNNs or 3DCNNs will satisfy the end-to-end definition. In the following subsections, we will present the strategies proposed in the literature as spatial and temporal modules, followed by end-to-end frameworks.

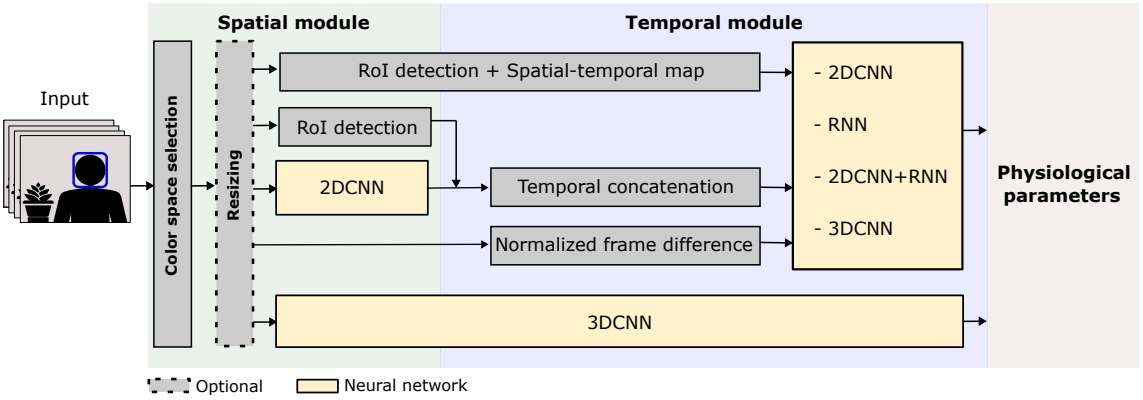

Figure 3.3: Deep-learning-based pipeline in physiological data measurement based in the literature.

3.3.2.1/ SPATIAL MODULE

Face tracking is performed on the video scene in the initial part of the deep learning pipeline, resulting in a series of frames with only the subject's faces. After that, different approaches are proposed in the literature to learn the spatial context, they usually transform the face frames into a convenient input to be used in a neural network. Figure 3.4 depicts the spatial-context techniques used as the spatial module in the literature. They consist in resizing the original face frames [90, 91, 65], hand-crafted RoI detection [102, 82, 62, 61], and even a specific CNN branch [54, 118]. Some spatial-context estimation techniques are also part of a spatial-temporal feature generated in the temporal module [82, 62, 61].

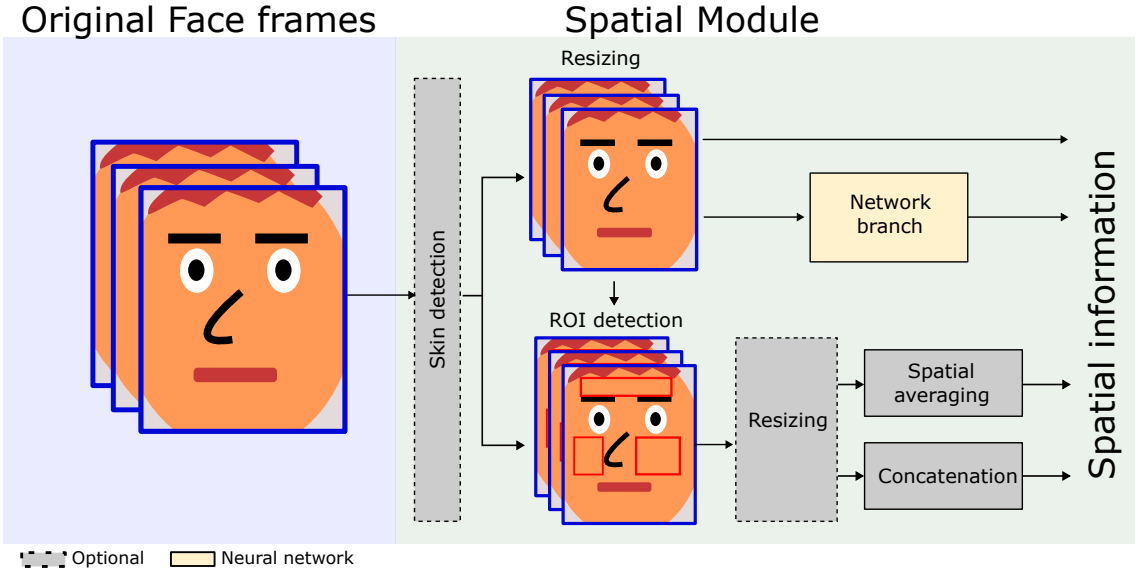

Figure 3.4: Spatial-context techniques used in the literature as the spatial module.

The spatial module usually starts with the process of resizing the input frames WxH. The dimensions after resizing include the following: 25x25 [75], 36x6 [54, 84], 64x64 [97, 112, 109], 128x128 [90, 91], 128x192 [65], 200x200 [83, 62, 82], and 256x192 [96].

Normally, authors do not explain why they chose their respective input size. However, we know that each input size is dependent for each network. That is to say, if a 180x180 input is optimal in one network, it does not mean it should be optimal for another. From the size of the input videos, we can only expect that the smaller they are, the fewer operations in the network, and therefore less training and inference time. In [109], the authors studied the importance of the spatial context in a 2DCNN, specifically, taking the input frames to the network of 64x64 pixels to down-sample to 1x1, 4x4, 8x8, 20x20, 30x30, 40x40, 50x50, and then up-sample back to 64x64. The results suggest that different resolutions cause minor fluctuations in network performance.

Similar to the regular pipeline in hand-crafted RPPG measurement methods, in the deep learning pipeline, there are also works where the subject face is divided in RoIs. Then, optionally, a RoI resizing procedure is performed [102, 82, 62, 61]. In [80], six RoIs are selected: nose, nose bridge, area under left eye, area under right eye, truncated facial box, and the full bounding box. The RGB channels of each RoI are spatial-averaged and then concatenated, having an 18-length vector. In [105], the face detection is made by 68 facial landmarks, then, three RoIs are selected, left cheek, right cheek, and both cheeks bounding box. In [102], three RoIs are selected: left cheek, right cheek and forehead in RGB and YUV color channels, the average pixel values are computed in every RoI as the spatial information. Another interesting work is the one proposed in [106], where the authors can estimate RPPG from only two RoIs, forehead and cheeks. Specifically, they propose Siamese-rPPG, a Siamese network architecture to simultaneously learn RPPG from the two face RoIs. Another RoI-based spatial module is the one proposed by Ying Qiu *et al.* in [15]. The authors detect 68 landmarks and choose 8 points to find a single RoI. The RoI is a rectangle comprising both cheekbones and nose. In [83, 62, 82], the authors find 81 landmarks in the input frame to get a 200x200 face bounding box, a skin detection process is performed in the bounding box and then divided in 25 5x5 RoIs. The spatial context is the YUV channel concatenation of the 25 spatial-averaged RoIs. Similarly, in [61], they authors use a very similar spatial module, the main difference is that they use the 81 landmarks to get the RGB cheek region instead of the full YUV skin face, then the bounding box is divided in 200 10x20 grids. Manual RoIs detection allows using zones of the face where the components of the PPG are located within the scene. However, it is an additional process, and the deep learning framework can not be considered end-to-end. In addition, it has been shown that neural networks can extract relevant information from the scene background [106, 121].

Interestingly, multiple authors have focused on assigning the acquisition of spatial information to a branch of their deep learning frameworks for the estimation of physiological parameters. For example, in [118], the PR measurement framework proposed by the authors is called Deep-HR, and it comprises an specific branch to manage the spatial information. The spatial module called Front-End (FE), is divided into two neural networks; the first is a 2DCNN based on RFB [58], which is in charge of face RoI detection, and the output of this network is refined using two adversarially trained encoder-decoders. The second FE network is a 2DCNN that distills the color information of the RoI. The output is three RGB signals that are also smoothed by an adversarially trained encoder-decoder. DeepPhys [54], is another 2DCNN network with an specific branch for acquiring spatial context. The authors named this branch as *Appearance model*, and it behaves as an attention module. Specifically, it learns soft-attention masks and assigns higher weights to skin areas.

3.3.2.2/ SPATIAL-TEMPORAL MODULE

In addition to the spatial module, the deep learning pipeline to measure physiological signals uses a temporal module that processes features within the scene related to the BVP signal over time. However, at this point, instead of interpreting the temporal module individually, it is more appropriate to consider a spatial-temporal context. In the literature, we find different approaches used as spatial-temporal modules depending on the type of neural network used. Figure 3.5 shows how the spatial-temporal context is acquired by combining the spatial and temporal modules.

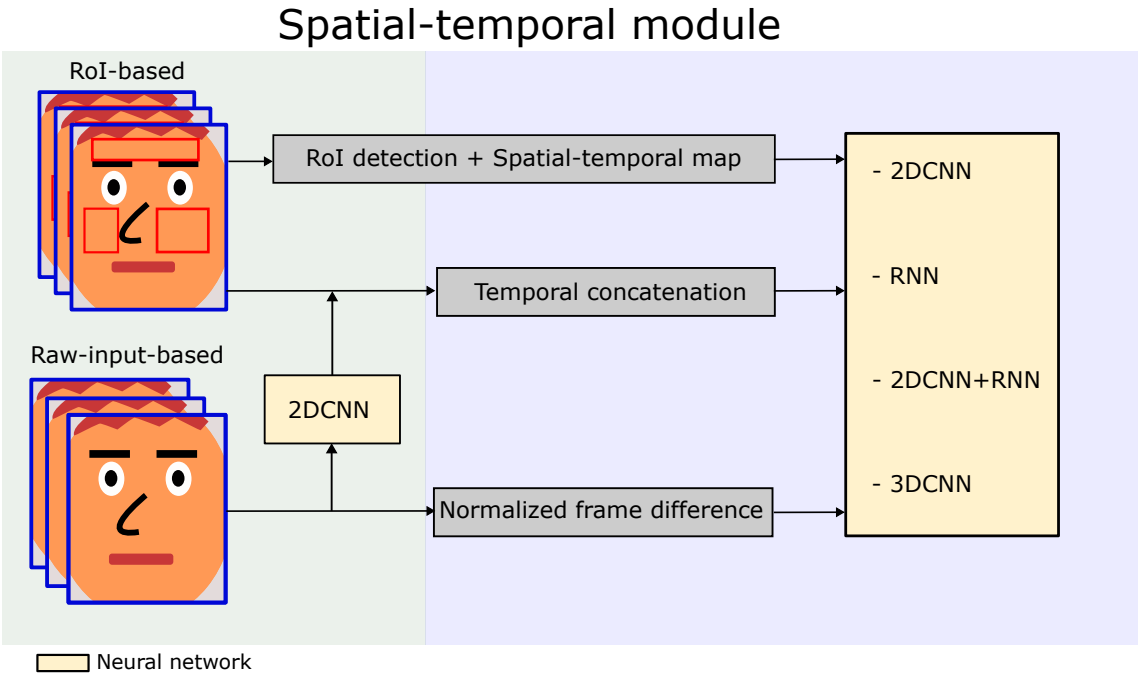

Figure 3.5: Spatial-temporal-context in the deep-learning-based pipeline.

Models with a branch as a spatial module also have a branch as a temporal module. Both branches compose a complete physiological data measurement framework. Regarding the DeepPhys network [54], since a 2DCNN lacks a temporal analysis, the authors propose normalizing the difference between two consecutive frames as input to a *Motion model*. Likewise, in the Deep-HR network [118], the Front-End branch is followed by a temporal module called Back-End (BE), capable of measuring HR from the FE output. For HR-CNN [65], the *HR Estimator* input is the temporal concatenation of the *Extractor*.

Several authors have proposed to give prior-knowledge to the neural networks by computing a spatial-temporal map [61, 102], also known as feature image [15], and spatial-temporal feature [83, 62, 62, 105]. Basically, they manipulate the spatial RoIs and concatenate them temporally, thereafter they are passed to a neural network. In [15], the blood flow information is extracted from the RoI by spatial decomposition for color magnification using Gaussian pyramid decomposition. Each frame is down-sampled, temporal concatenated and filtered between 0.7 and 4 Hz (45-240 bpm). Finally, the PR is estimated through an 2DCNN framework called EVM-CNN. In [83, 62, 61], the temporal feature map is the 300-frames (around 10 seconds) temporal concatenation of a 25-length [83, 62] and 200-length [61] spatial vector, the output is used to feed a 2DCNN Residual network. Interestingly, in [105], the spatial temporal feature map is generated by using an

sliding window of 5-seconds with 1-frame step on the RPPG measured with Chrom [18] on three face RoIs. This way, the spatial temporal map is in fact, overlapped segments of the RPPG signal, the output is then given to a 2DCNN residual network to measure PR. In [102], multi-scale spatial-temporal maps (MSTmaps) are generated from the RoIs concatenated in temporal sequences. To measure physiological data, the authors use a cross-verified disentangling strategy to train an auto-encoder composed of two encoders. The inputs are pairwise face video clips that generate two MSTmaps; these are used to train the two 2DCNN-based encoders. The first encoder will be related to physiological measurements, while the other encoder will have non-physiological information. Finally, a physiological estimator module takes the disentangled features related to RPPG and average PR.

Some authors have explored adding 2DCNNs+RNNs to the spatial-temporal maps. Bin Huang *et al.*[96], for example, used a series of stacked long-short-term-memory layers (LSTM). In [82], similar to [83, 62, 62, 105], a 300-frames temporal-feature map is given to a 2DCNN residual network with an additional GRU to consider the relationship of adjacent PR values. In [97], the authors use a Bidirectional Long short-term memory (BLSTM) RPPG estimator to model the temporal information from the spatial-temporal map. In the same way, we can find works using 3DCNNs from RoIs. In [106], the Siamese-rPPG network comprises two 3DCNN branches with the same architecture and weights (shared-weights). Each branch of the Siamese network encodes one RoI into its corresponding RPPG. In [99], the DeeprPPG network is fed with one or multiple RoIs and then the 3DCNN find the spatial-temporal information to estimate RPPG.

The works just exposed can measure physiological data by means of 2DCNN, 2DCNNs+RNNs, and 3DCNNs. But they need and an additional procedure where a prior spatial context knowledge is taken into account. As mentioned before, these works should not be considered end-to-end implementations, besides, the additional process makes them harder to implement compared with an end-to-end framework. In the next subsection we will discuss the end-to-end physiological data measurement frameworks.

3.3.2.3/ END-TO-END FRAMEWORKS

End-to-end frameworks are useful because they are easy to implement (only neural network implementation). In the physiological data measurement pipeline, an end-to-end framework should estimate the physiological data directly from the resized video input. To the best of our knowledge, the only work which performs the end-to-end approach using 2DCNNs, is the HR-CNN network proposed in [65]. The two-step convolutional network has an *Extractor* and *HR Estimator* modules. First, a video is taken to detect and resize the faces, and then, the cropped video is passed through the *Extractor* to acquire its RPPG component. Finally, the RPPG is the input in the *HR Estimator*, and its HR is the output.

Alternatively, three-dimensional convolutional networks can simultaneously analyze a video's spatial and temporal characteristics. Using 3DCNN-based networks is a simple way to measure end-to-end RPPG. For instance, perhaps one of the most iconic 3DCNN-based networks is PhysNet, proposed by Zitong Yu *et al.*in [90], as it has been taken as a reference in other studies [112, 119, 108]. The authors make a performance comparison of spatial-temporal networks using 2D, 3D, LSTM, and bidirectional LSTM layers. Where the use of 3DCNN networks outperformed the combination of 2DCNN and recur-

rent networks. With this method, it is possible to acquire RPPG signals from video in an end-to-end approach. In [91], authors used a 3DCNN network called rPPGNet. rPPGNet is composed of a skin-based attention module that helps to adaptively select skin regions, a partition constraint module that learns a better representation of the RPPG signal features, and a spatial-temporal CNN. The input is the resized face of the subject present in each frame, and the output is the RPPG. Gideon and Stent in [112] use a modified version of the 3DCNN-based PhysNet architecture to learn spatial-temporal features over the input video. Interestingly, this is the first approach that allows the acquisition of RPPG signals with a 3DCNN in a self-supervised way. Moreover, they propose a saliency sampler to obtain an interpretable output to ensure that the system behaves correctly.

Another 3DCNN-based end-to-end framework is the one presented in [122] where the authors propose a multi-hierarchical fusion divided into four parts. The first part is a 3DCNN-based low-level face feature generator that extracts the facial color distribution features, the input is an RGB face video clip, and the output is the low-level feature map. The second part is a 3DCNN module for deeper feature extraction. The third part is a multi-hierarchical feature fusion module composed of a feature extractor and a skin map generator. This module retains more relevant information related to the RPPG signal. Finally, the last part is a 3DCNN signal predictor module that generates the RPPG output. Likewise, in [108], the authors use convolutional layers and neural architecture search to find an optimal RPPG 3DCNN baseline. Architecture search aims to find cells to form a network backbone for RPPG measurement automatically (optimization). Apart from regular convolutions, the authors propose a novel 3D convolution called Temporal Difference Convolution (TDC) for network architecture optimization. TDC is a procedure that takes RGB frames as input to describe the temporal differences on feature levels.

In this way, we can see the versatility of end-to-end frameworks. All of the works presented in this subsection are the implementation of a single neural network composed of different branches. However, when reviewing the inference times in the works that provide such information, we notice that they are unsuitable for a real-time application in low-end devices. Real-time capability, in our context, typically refers to when a model runs faster than a webcam at 30 fps (33.3 ms per frame). For example, in PhysNet [90] for a video of 30 seconds, the inference speed is 235 ms, which is a quick response, the disadvantage lies in the hardware used. They used a high-performance NVIDIA Tesla P100 GPU. The same GPU is used in rPPGNet [91], however, they inference time is not given. Similarly, in [112] the authors use an high-performance NVIDIA Tesla V100 GPU but the inference time is not given. In [122] they use a high-performance NVIDIA GeForce RTX 2080Ti GPU. None of the work was evaluated on a CPU. Thus, developing an end-to-end framework for estimating physiological data from videos that can be used in real-time applications on both CPU and GPU is important, and this is precisely what we propose in Chapter 6.

Table 3.3 summarizes all the works discussed in this chapter related to the physiological data measurement pipeline. We present the reference of the networks proposed in the literature and the primary constitution of their architectures. In addition, we compile the information used as input and the output of each method. Finally, we show the methods used to measure PR, the color input color channel used, and the evaluated databases.

Table 3.3: Publications related to convolutional neural networks in the physiological data measurement task

Publication	Network Architecture	Input	Output	PR estimation	Color Channel	Data
DeepPhys [54]	2DCNN	Feature map	RPPG	FFT	RGB,NIR	RGB Video 1,RGB Video 2,MAHNOB-HCI, Infrared Video (NIR)
HR-CNN [65]	2DCNN	Video	RPPG/HR	–	RGB	RGB MAHNOB-HCI,COHFACE,PURE, ECG-Fitness
EVM-CNN [15]	2DCNN	Feature Image	HR	–	RGB	MMSE-HR
[80]	2DCNN	Six RoIs	HR	–	RGB	Self-collected
[84]	2DCNN (DeepPhys-based)	Feature map	RPPG	FFT	RGB	RGB Video 1
[83]	2DCNN (ResNet-18-based)	Spatial-temporal maps	HR	–	YUV	MMSE-HR, VIPL-HR
Deep-HR [118]	2DCNN + FCN	Video	HR	–	RGB	MAHNOB-HCI, HR-D(Self-collected)
[96]	2DCNN+LSTM	Video	HR	–	RGB	Self-collected
[62]	2DCNN (ResNet-18-based)	Spatial-temporal maps	HR	–	YUV,NIR	MMSE-HR, VIPL-HR
RhytmNet [82]	2DCNN (ResNet-18-based)+GRU	Spatial-temporal maps	HR	–	YUV,NIR	MAHNOB-HCI, MMSE-HR, VIPL-HR
Meta-rPPG [97]	2DCNN+BLSTM	Video	RPPG	Peak detection	RGB	MAHNOB-HCI, UBFC-rPPG
PhysNet [90]	2DCNN+LSTM, 3DCNN	Video	RPPG	Peak detection	RGB	MAHNOB-HCI, OBF
rPPGNet [91]	3DCNN	Video	RPPG	Peak detection	RGB	MAHNOB-HCI, OBF
DeeprPPG [99]	2DCNN+3DCNN	Three RoIs	RPPG	FFT	RGB	PURE, COHFACE, MAHNOB-HCI
Siamese-rPPG [106]	3DCNN	Two RoIs	RPPG	FFT	RGB	PURE, COHFACE, UBFC-rPPG
[112]	3DCNN (PhysNet-based)	Video	RPPG	FFT	RGB	PURE,COHFACE,UBFC-rPPG, MR-NIRP-Car(RGB only)
[122]	3DCNN	Video	RPPG, HR	–	RGB	COHFACE, UBFC-rPPG, Self-collected
AutoHR [108]	3DCNN (PhysNet-based)	Video	HR	–	RGB	MAHNOB-HCI, MMSE-HR, VIPL-HR
Synrythm [61]	2DCNN (ResNet-18)	Spatial-temporal maps	HR	–	RGB	MAHNOB-HCI, MMSE-HR
[75]	3DCNN	Video	HR	–	RGB	UBFC-rPPG
[105]	2DCNN (ResNet-18-based)	Spatial-temporal maps	HR	–	RGB	MAHNOB-HCI, VIPL-HR, UBFC-rPPG
[109]	2DCNN (DeepPhys-based)	Feature map	RPPG	FFT	RGB	PURE, HNU(self-collected)
RPNet [119]	3DCNN (PhysNet-based)	Video	RPPG	FFT	RGB	Self-collected
[102]	2DCNN	Pairwise video	RPPG,HR	–	RGB, YUV	MMSE-HR, OBF, VIPL-HR

Databases: RGB Video 1 [21], RGB Video 2 [43], Infrared video [34], MAHNOB-HCI [13], COHFACE [45], PURE [24], ECG-Fitness [65], MMSE-HR [42], VIPL-HR [62], UBFC-rPPG [74], OBF [57], MR-NIRP-Car [103].

3.4/ SUMMARY

In this chapter, we presented multiple interesting works that have been done for physiological data measurement with convolutional networks and RPPG signal filtering. Due to the rapid growth of the topics covered, this review does not present every work found in the literature, but perhaps the most important ones. Inspired by the possibilities and weaknesses of all the reviewed methods, in chapter 5, we will propose a deep learning-based method for RPPG signal filtering, and in chapter 6, a new 3DCNN-based architecture for real-time RPPG signal acquisition.

The following chapters are related to the contributions of this thesis. We will start with the experimental setup.

II

CONTRIBUTIONS

4

EXPERIMENTAL SET-UP

This chapter explains all the experimental procedures we performed during this thesis. The section starts with elaboration of the evaluation protocols, then the constitution of the databases, and finally the metrics used to assess pulse rate, signal quality, and pulse rate variability. For the remainder of this thesis, we will use the notation pulse rate (PR) to refer to the number of heartbeats per minute, measured in beats per minute (bpm).

4.1/ EVALUATION PROTOCOLS

In chapters 5 and 6, the evaluation protocols used are based on intra-dataset and cross-dataset. Intra-dataset refers to the training and testing of neural networks within a specific database, i.e., data with the same configuration. Whereas, cross-dataset refers to the training in one database and testing in a different one; this kind of evaluation is important since it assesses the generalization capacity of the neural network. For both protocols, subject independent evaluation is carried out, i.e the subject's information is not shared between training and testing data.

Cross-validation evaluates a model by training it on a subset of the data and testing it on the remaining data; this is done multiple times, with each subset of the data being used as both the training and testing data. Cross-validation provides an accurate estimation of the model's performance. In experiments using the intra-dataset evaluation protocol, we will use 5-fold cross validation. Figure 4.1 depicts the process: the data is divided into five-folds, then, in five splits, each fold is taken as a validation set and the remaining four as training. The overall result is the average of the results of each division.

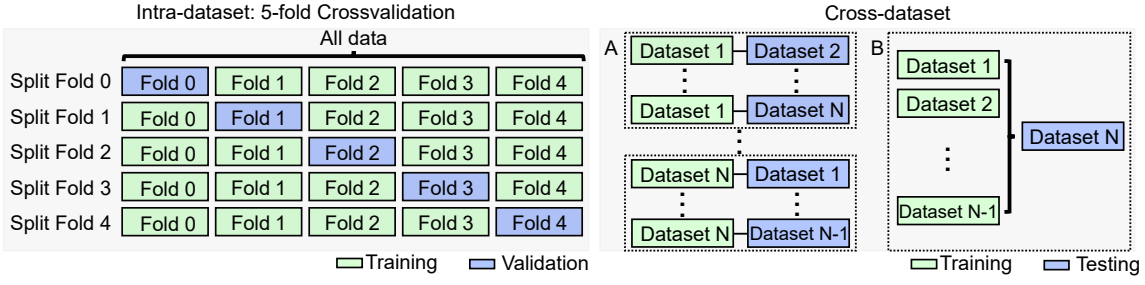

Figure 4.1: 5 fold cross-validation. The data is divided into five folds, the overall result is the average of all splits. Cross-dataset can be A)Training in each dataset and test in the remaining datasets, and B) Train in one or multiple datasets to test in a different one.

The cross-dataset protocol can be done in two ways. The first is to train each available database to test the model on the remaining databases individually (Figure 4.1 Cross-dataset A). However, this approach can require a large amount of computational time. Some authors [90, 82, 91, 83] have opted to train on one or several databases and choose a single database for testing (Figure 4.1 Cross-dataset B).

The pulse rate in the RPPG signals varies depending on the subject's activity. If the subject is stable and starts exercising, the pulse rate will increase; if the same subject relaxes, the pulse rate will decrease. For this reason, to detect all the PR changes in the signals to be analyzed, we will use a sliding window of 15 seconds long with a step of 0.5 seconds. In each resulting window, the pulse rate is measured as shown in Figure 4.2.

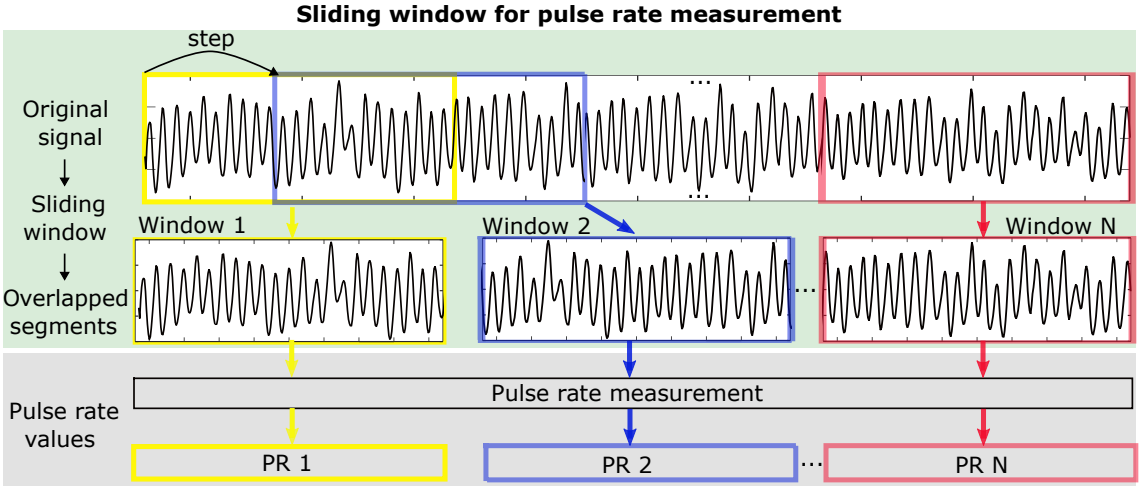

Figure 4.2: Sliding window for pulse rate measurement.

4.1.1/ OVERLAP-ADD

The neural networks we will use in chapters 5 and 6 evaluate signals through a sliding window process similar to the one presented in Figure 4.2; the main difference is the sliding window step. Also, we compute the deep learning model instead of measuring pulse rate. Thus, the output is still a series of overlapped segments. In order to generate a single resultant signal from the overlapped segments, we use the overlap-add procedure explained below.

Let us define the temporal signal $\mathbf{s}$ as a vector of size T. Then, the overlapped segments computed using a L-length sliding window with a step of one on $\mathbf{s}$ are the input to a neural network η, which gives the matrix output $\mathbf{S}$ of T-L+1 rows and L columns. This procedure is defined by the Equation 4.1:

$$\mathbf{S} = \eta(\mathbf{s}[j : j + L - 1]) \tag{4.1}$$

with $j \in [1, T - L + 1]$. $\mathbf{S}$ is composed of L-length rows with overlapped fragments of the resulting signal. In order to unify them into the resulting signal, the overlap-add procedure is presented by the following equation:

$$\hat{s}[j] = \sum_{l=1, j-l+1>0}^{L} S[j-l+1][l], \tag{4.2}$$

with $\hat{\mathbf{s}}$ as the output combined signal. The overlap-add process is illustrated in Figure 4.3.

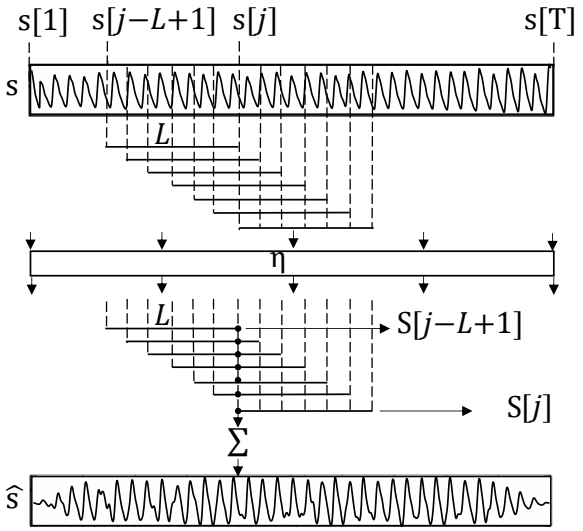

Figure 4.3: Overlap-add. Illustration of the overlap-add procedure used to generate a single signal from the combination of multiple L-length windows.

This process makes a summation of all the overlapped fragments causing the generated signal to have a larger amplitude than the input signal. In addition, the beginning and end of the resulting signal will have small values. Therefore, performing a normalization process of the resulting signal between the desired range of values is necessary.

4.2/ DATABASES

Databases are essential in machine and deep-learning because they provide a way to store and organize data. Databases keep this data organized so the algorithms can easily access and use them. This section presents the databases that are used in this thesis, but it is essential to explain the pre-processing of the databases beforehand.

4.2.1/ PRE-PROCESSING: GROUND TRUTH CONDITIONING

During data acquisition, due to the movement of the subjects, or failures in the acquisition devices, some ground truth signals present anomalies. These inconsistencies (gaps and false peaks) usually happen at the beginning and end of the acquisitions and less frequently during the measurement (Figure 4.4). However, due to the nature of the neural networks, it is not suggested to perform the training procedure with ground truth signals that present these types of problems. Therefore, a ground truth signal conditioning step is necessary. For this purpose, ground truth signals are checked individually to take only the continuous segment with a reliable ground truth morphology. In most signals, it was

sufficient to remove the first and last seconds, while only a considerable part of the signal was removed in a small group.

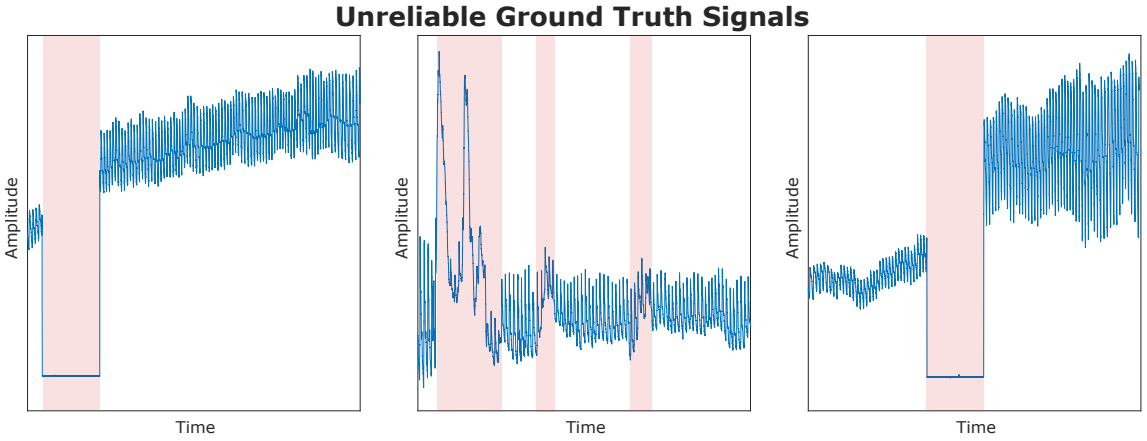

Figure 4.4: Unreliable ground truth signals. Segments highlighted in red are anomalies due to the movement of the subjects, or failures in the acquisition devices.

In order to work correctly with the videos and ground truth signals in any neural network, we must perform a pre-processing procedure. First of all, the sampling rate of the videos and the ground truth signals must be equal. As we do not want to modify the videos, we will necessarily have to decrease or increase the sampling rate of the ground truth signal at the frames per second (fps) of the videos (in most cases, the sampling rate of the ground truth signal is decreased). Typically, with the same sampling rate, the number of frames and ground truth signal points should be the same. However, in some databases, this is not always the case. For subjects where the duration of the video and ground truth signal are different, we will remove the last frames/points of the longer video/signal until it has the same duration as the shorter video/signal. This way, we guarantee that each frame corresponds to a single value in the ground truth signal.

As mentioned before, removing the non-reliable parts of the ground truth signal is necessary, so we plot the signals and visually select the start and end of the reliable segment. Then, we compute the signal to zero-mean, followed by a detrending process. Now we normalize the signal between minus one and one. Depending on the database, we will perform a video synchronization process employing the RPPG signal. Some databases do not guarantee that the ground truth signal and the frames are synchronized, and a time offset between both signals can affect the training of the network [109]. We use the PVM method [60] to acquire the RPPG signal on the videos, then we plot the RPPG and ground truth signals superimposed. Both signals should be synchronized; if not, we shift the ground truth signal until all the peaks of this signal match those of the RPPG signal using a phase alignment procedure[1]. All the mentioned steps are shown in Figure 4.5. Finally, to perform a consistent pulse rate analysis, we only used signals equal to or greater than 15 seconds for all databases.

Now that we know the pre-processing performed on the databases, we will introduce the databases that we will use in this thesis.

[1] https://github.com/pearsonkyle/Signal-Alignment/blob/master/signal_alignment.py

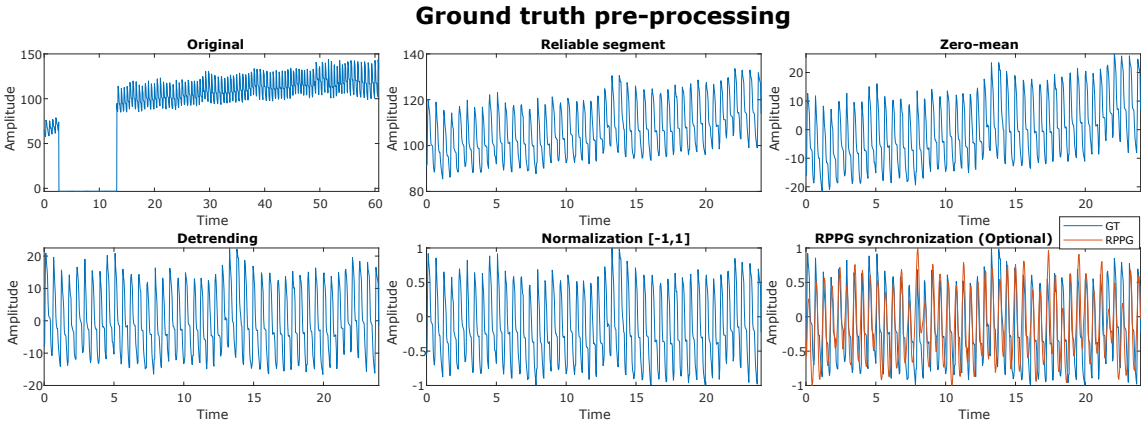

Figure 4.5: Ground truth pre-processing. All ground truth files were modified following the next steps: Reliable segment selection, zero-mean, detrending, normalization between -1 and 1, and synchronization using RPPG (optional).

4.2.2/ DESCRIPTION OF THE DATABASES USED

This section describes the databases used in this work: VIPL-HR, MMSE-HR, COHFACE, ECG-Fitness, and UBFC-rPPG.

4.2.2.1/ VIPL-HR

We used the VIPL-HR database collected by the Institute of Computing Technology Chinese Academy of Sciences [82, 62]. The database contains 107 subjects recorded by three different instruments in nine scenarios. Although this database contains 752 near-infrared videos, we only consider the 2378 visible light videos, the resolutions of the videos are between 960 x 720 and 1920 x 1080, at 25 and 30 fps, respectively. The ground truth photoplethysmography signals were recorded using the CONTEC CMS60C BVP sensor at 60 Hz. Some subjects examples are presented in Figure 4.6

Figure 4.6: Examples of subjects in VIPL-HR database. Taken from [82]

.

On this database, we performed subject-independent 5-fold stratified cross-validation. Specifically, taking each subject's labels related to the following nine scenarios: Stable, motion, talking, dark, bright, long distance, exercise, phone stable, and phone motion.

4.2.2.2/ MMSE-HR

The Multimodal Spontaneous Emotion Corpus - Heart Rate database (MMSE-HR) [42] includes videos with versatile facial movements and expressions. The database has been built for further investigation in emotion recognition. One hundred forty subjects, 58 males and 82 females, with ages ranging from 18 to 66 years old, are recorded performing multiple tasks. The resolution of each video is 1040x1392 pixels with a frame rate of 25 fps. The BIOPAC 150 data acquisition system obtained the blood pressure ground truth at 1 KHz. The duration of each video is between 30 and 60 seconds. Some subjects examples are presented in Figure 4.7

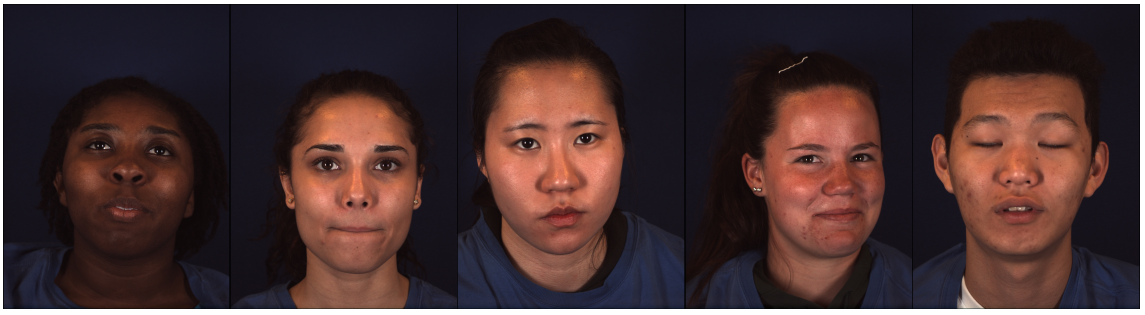

Figure 4.7: Examples of subjects in MMSE-HR database [42]

.

4.2.2.3/ COHFACE

In the COHFACE database [45] two experimental setups with studio and natural lighting are introduced. Four one-minute videos with different luminance were acquired from 40 subjects with a Logitech HD Webcam C525. In total, there are 160 videos with a resolution of 640x480 pixels and a frame rate of 20 fps. The videos were heavily compressed using MPEG-4 Visual. The BVP ground truth signals of photoplethysmography were acquired with contact RPPG sensor at 256 fps. The average subject age is 35.6 years old, with a standard deviation of 11.47 years. Gender-wise, there are 28 men (70%) and 12 women (30%). Some subjects examples are presented in Figure 4.8

Figure 4.8: Examples of subjects in COHFACE database. [45]

4.2.2.4/ EGG-FITNESS

In ECG-Fitness database [65], authors propose a new challenging database for the RPPG measurement task. Using two Logitech C920 web cameras, one attached to the currently used fitness machine and the other positioned close on a tripod, 204 one-minute videos are recorded on 17 subjects. Three lighting setups are used: natural, halogen, and led light. The 17 subjects include 14 males and three females with an age range of 20 to 53 years. They were recorded performing the following activities: speaking, rowing, exercising on a stationary bike and exercising on an elliptical trainer. The videos were recorded with 1920x1080 pixels, 30 fps, and stored in an uncompressed YUV planar pixel format. One subject performing the four scenarios recorded by the two cameras is presented in Figure 4.9.

Figure 4.9: One subject performing the four scenarios recorded by the two cameras in the ECG-Fitness database. Taken from [65].

During the video capture, an electrocardiogram was recorded with a two-lead Viatom CheckMeTMPro device with CC5 lead, giving the ECG signal as ground truth. However, in this thesis, we work with photoplethysmography signals. To transform the ECG ground truth signals into PPG ground truth signals, we use the generative adversarial network P2E-WGAN [120], trained specifically for this task (Figure 4.10).

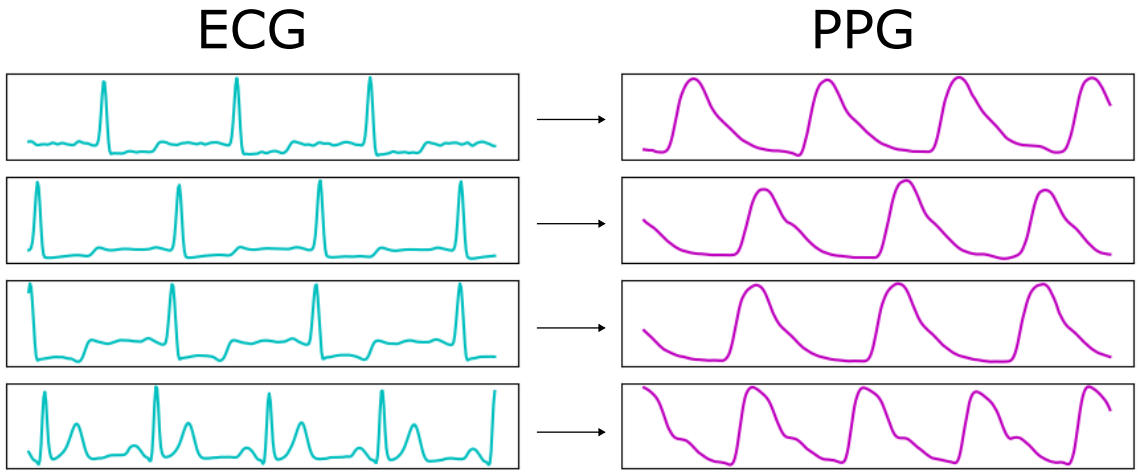

Figure 4.10: Examples of the ECG to PPG transformation using P2E-WGAN [120] in the ECG-Fitness dataset.

4.2.2.5/ UBFC-RPPG

In [74], authors propose the UBFC-rPPG database (stands for Univ. Bourgogne Franche-Comté Remote PhotoPlethysmoGraphy), it consists of two sub-sets of data, ”simple” and ”realistic”. In ”simple”, participants were asked to sit still. In ”realistic”, participants were asked to sit in front of the camera to play a time sensitive mathematical game that aimed at augmenting their pulse rate.

The UBFC-rPPG database was created using a simple low cost webcam Logitech C920 HD Pro at 30fps in uncompressed 8-bit RGB format with a resolution of 640x480. To obtain the ground truth PPG signal, a CMS50E transmissive pulse oximeter was used. Subjects were about one meter away from the camera, and each video was around one minute duration. We used the ”realistic” sub-set composed of 42 videos, some subjects examples are presented in Figure 4.11

Figure 4.11: Examples of subjects in UBFC-rPPG database [74].

4.2.3/ GROUND TRUTH SIGNAL DURATION AND PULSE RATE HISTOGRAMS

After performing the pre-processing step on each database, we plot the histogram of ground truth signal duration in Figure 4.12 (left side). In the same image (right side), we use a 15-seconds sliding window with 0.5-seconds step on the ground truth BVP signals to measure the PR value. Then, we plot the histogram of the PR values. The respective histograms per fold are presented in the annexes section A.2.

MMSE-HR has 98 signals, COHFACE has 164, VIPL-HR 2256, UBFC-rPPG 42, and ECGFitness 195. Regarding the total duration, MMSE-HR has approximately 64 minutes, COHFACE 166 minutes, VIPL-HR 1114 minutes, UBFC-rPPG 44 minutes, and ECG-Fitness around 187 minutes. Thus, UBFC-rPPG is the smallest database and VIPL-HR the biggest. In general, COHFACE, UBFC-rPPG, and ECG-Fitness have signals with around one-minute duration. MMSE-HR and VIPL-HR have shorter signals, around 40 seconds and 30 seconds, respectively.

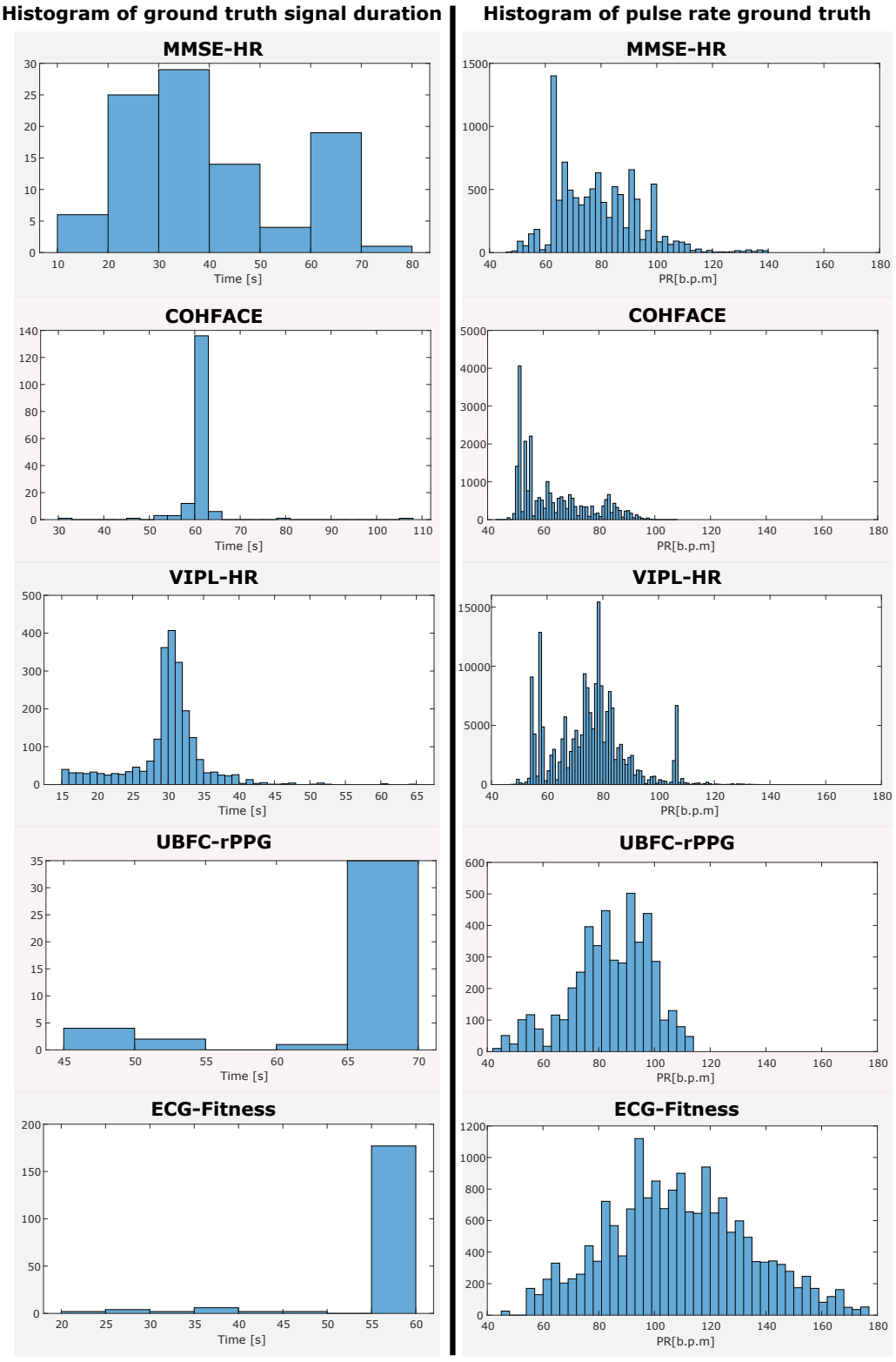

Figure 4.12: Histograms with the duration of ground truth signals and HR distribution in each database after pre-processing.

From the right side of Figure 4.12, there is a more significant amount of data for intermediate of pulse rate values, between 60 and 100 bpm. This distribution is related to subjects in a neutral state, i.e., not exercising. In ECG-Fitness, where there are more cases of subjects exercising, there is a more significant amount of data above 100 bpm. MMSE-HR has a Gaussian behavior between 80 bpm, COHFACE presents a peak around 50 bpm. VIPL-HR and UBFC-rPPG have a Gaussian distribution around 80 bpm. Finally, ECG-Fitness presents a normal distribution around 110 bpm.

4.3/ METRICS

In order to measure the pulse rate from RPPG signals, there are two methods. The first approach uses the peaks of the signal representing the heartbeats, and the time between the peaks allows us to calculate the pulse rate [97, 90, 91]. On the other hand, in this thesis, we will use the second method based on frequency analysis using the Fast Fourier Transform (FFT).

Returning to Figure 4.2, the "Pulse rate measurement" block is replaced by a frequency-based pulse rate measurement. This method consists of taking the signal $\mathbf{s}$ and computing its Fourier transform ($\mathcal{F}\{\mathbf{s}\}$). The resulting signal in the frequency domain will have the peak with the highest power at the frequency position related to the pulse rate; this measurement is given in Hz. To convert this value to bpm, we have to multiply the value in Hz by 60. Figure 4.13 shows an example of a signal whose highest peak in the frequency domain is at 1.82 Hz. By multiplying this value by 60, we obtain the pulse rate of 109.2 bpm.

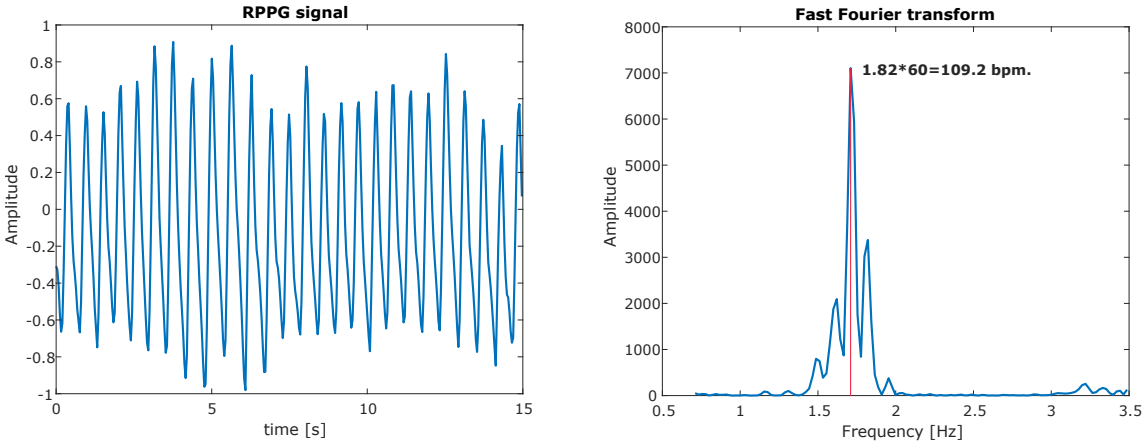

Figure 4.13: Example of how PR is measured using FFT. The FFT is performed on the RPPG signal. The frequency value of the highest peak (1.82 Hz) of power is multiplied by 60 to get the result in bpm (109.2 bpm).

Based on the literature, the most important metrics are those related to pulse rate measurement accuracy. In addition, we propose metrics to evaluate signal quality. Finally, to assess whether or not the resulting signals are helpful for medical applications, we also use metrics to measure pulse rate variability. In the following, we explain all the metrics used.

4.3.1/ Pulse Rate

To evaluate RPPG measurement, we use the Mean Absolute Error (MAE) an the Pearson's correlation coefficient (r) between the PR measured in the ground truth signal and the PR measured from the RPPG.

4.3.1.1/ Pulse Rate Mean Absolute Error (MAE)

The mean absolute error was calculated as the window-wise mean of the pulse rate calculated using the contact-based ground truth waveform obtained by pulse oximeter (**Pc**), and the pulse rate calculated using the RPPG signal (**Pr**). The MAE of the two vectors **Pr** and **Pc** of size n is presented in equation 4.3:

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}\left|\mathrm{Pr_j} - \mathrm{Pc_j}\right|, \tag{4.3}$$

where $\mathrm{Pr_j}$ and $\mathrm{Pc_j}$ are the value of **Pr** and **Pc** at position j, respectively.

4.3.1.2/ Pearson's correlation coefficient (r)

Pearson's correlation coefficient measures the linear correlation between vectors **Pc** and **Pr**. A value of $r = 1$ means a positive total linear correlation, while a $r = -1$ implies a negative linear correlation, finally, a $r = 0$ indicates that there is no linear correlation between the estimations and the reference values. r is given by equation 4.4:

$$r = \frac{\sum_{j=1}^{n}\left(\mathrm{Pr_j} - \overline{Pr}\right)\left(\mathrm{Pc_j} - \overline{Pc}\right)}{\sqrt{\sum_{j=1}^{n}\left(\mathrm{Pr_j} - \overline{Pr}\right)^2}\sqrt{\sum_{j=1}^{n}\left(\mathrm{Pc_j} - \overline{Pc}\right)^2}}, \tag{4.4}$$

where $\overline{Pr}$ and $\overline{Pc}$ are the averages of **Pr** and **Pc**, respectively.

4.3.2/ Signal Quality

To assess the signal quality of the RPPG signals, we use two metrics: The Signal to Noise Ratio (SNR) and the Template Match Correlation (TMC).

4.3.2.1/ Signal to Noise Ratio (SNR)

In the pulse rate measurement context, the SNR can be defined as the ratio of the power of the main pulsatile component and the power of background noise. Due to the wide dynamic range of the signals, SNR is computed in dB and it is presented in Equation 4.5:

$$\mathrm{SNR} = 10\log_{10}\left(\frac{\int_{f_l}^{f_u} h_{signal}|\mathcal{F}\{\mathbf{s}\}|^2\mathrm{d}f}{\int_{f_l}^{f_u} h_{noise}|\mathcal{F}\{\mathbf{s}\}|^2\mathrm{d}f}\right) \tag{4.5}$$

where f_l and f_u are the lower and upper limit of the integral defined by the possible physiological range of the pulse rate. h is a double step function used to include the first and second harmonics, defined by the convolutions presented in Equation 4.6.

$$h_{signal}(f) = [\delta(f - f_0) + \delta(f - 2f_0)] * \prod(\pm f_r)$$
$$h_{noise}(f) = 1 - h_{signal}(f) \tag{4.6}$$

where δ is the Dirac delta function, f_0 is the fundamental frequency (i.e. the highest peak in $F\{\mathbf{s}\}$), convoluted with Π, the *rect* function of half-width f_r.

We set the lower frequency range in $f_l = 0.7$ (42 bpm), the upper in $f_u = 3.5$ (210 bpm), and a width of $f_r = 0.4$. We found the first and second harmonics in the ground truth signal and measured the SNR in the RPPG FFT spectrum, as depicted in Figure 4.14.

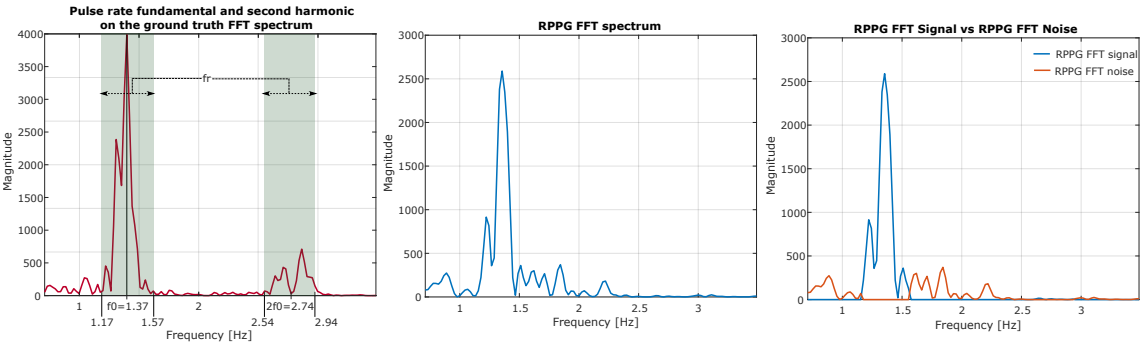

Figure 4.14: RPPG SNR measurement. The first and second harmonics (f_0 and $2f_0$, respectively) are founded in the ground truth FFT spectrum, and the SNR is measured in the RPPG FFT spectrum.

4.3.2.2/ Template Match Correlation

The Template Match Correlation (TMC) [23] is also used as an RPPG signal quality assessment metric. The following steps are performed regarding full-length signals: First, the signal peaks are detected and the median beat-to-beat interval (IBI) is calculated. Then, all pulses are extracted individually centered on their respective peak with a window width equal to the median beat-to-beat interval. Finally, the template is calculated as the average of all the pulses, and the TMC coefficient is calculated as the average of the correlation of all the pulses with the template. Figure 4.15 depicts the TMC measurement procedure. A value close to $\mathrm{TMC} = 1$ means that the pulse shape of the evaluated or filtered signal is uniform, and therefore close to the expected signal, while a value close to $\mathrm{TMC} = 0$ indicates the contrary.

4.3.3/ Pulse Rate Variability

Pulse rate variability (PRV) consists of changes in the time intervals between consecutive heart beats [48] (RR intervals). An optimal level of PRV is associated with health and self-regulatory capacity. At the same time, PRV abnormalities may reveal health problems such as atrial fibrillation [48]. Hence, it is vital that work focused on measuring RPPG

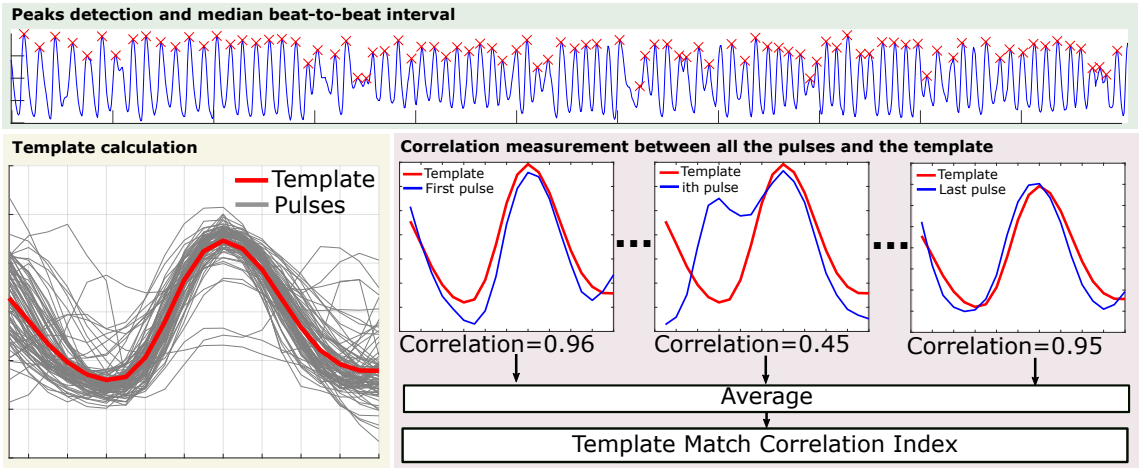

Figure 4.15: TMC measurement. The template is calculated as the average of all pulses. Then, the TMC is the average of the correlation of all the pulses with the template.

can be used to measure PRV accurately. This thesis uses two time-domain (SDNN and RMSSD) and two frequency-domain (LF and HF) metrics to evaluate PRV. These four metrics were measured on RPPG signals and ground truth signals. Since our objective is to measure the improvement of PRV compared to the reference signal, we propose to measure the absolute error between the PRV measured by RPPG and ground truth signals, using the prefix ”E” to indicate the error, *i.e.* E:SDNN, E:RMSSD, E:LF, and E:HF. The PRV features used in this thesis are considered ”ultra-short-term” since most of the signals used had a duration shorter than five minutes [48].

Time-domain pulse rate variability metrics measure the amount of PRV present during monitoring periods [48]. SDNN stands for Standard Deviation of NN intervals, where NN intervals refers to interbeat intervals from which artifacts have been removed. RMSDD is the Root Mean Square of successive RR interval differences. The RMSSD reflects the beat-to-beat variance in PR. Frequency-domain measurements estimate the power distribution of relative or absolute power into specific frequency bands. LF is related to the absolute power of the low-frequency band, while HF refers to the absolute power of the high-frequency band. The LF band (0.04–0.15 Hz) is comprised of rhythms with periods between 7 and 25 s and is affected by breathing from 3 to 9 bpm, and HF or respiratory band (0.15–0.40 Hz) is influenced by breathing from 9 to 24 bpm. In Figure 4.16, we show the overview of how pulse rate variability metrics are measured from an BVP signal.

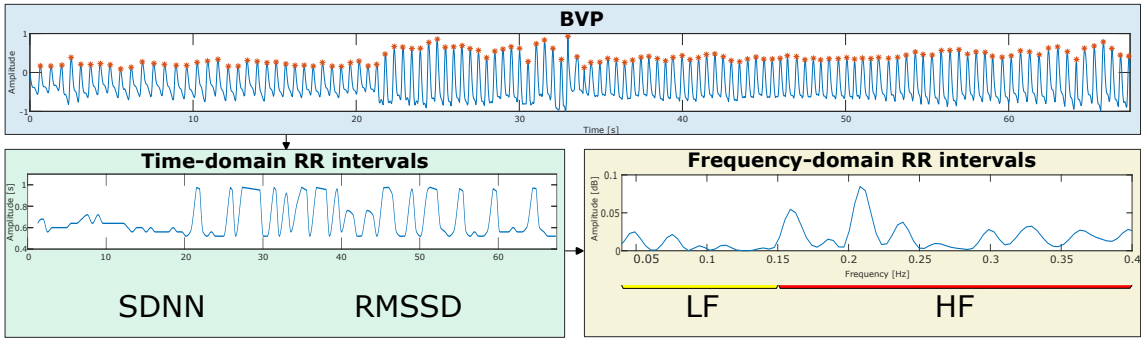

Figure 4.16: Pulse rate variability metrics. SDNN, RMSSD, LF, and HF.

The following two chapters will present two contributions to this thesis. The first is a filter for RPPG signals in chapter 5, and the second is a framework for measuring RPPG signals from video in real-time (chapter 6).

5

LSTMDF: Long Short-Term Memory Deep-Filter

5.1/ Overview

Like any other signal, the quality of RPPG signals deteriorates due to different noise sources. The causes include light changes within the scene, movement, quality of camera sensors, and the digitalization of the images. Conventionally, bandpass and wavelet filters have been used to suppress noise in RPPG signals. However, some alterations exist after using traditional filters, and an experienced eye can easily identify them.

This chapter proposes the Long Short-Term Memory Deep-Filter (LSTMDF) network in the RPPG filtering task. To identify and clean the RPPG artifacts, we analyze two sequence-to-sequence approaches: many-to-one and many-to-many. We use the MMSE-HR, VIPL-HR, and COHFACE databases in intra-dataset and cross-dataset scenarios with different protocols to explore the method's performance. We demonstrate that LSTMDF is efficiently trained with 90 signals totaling 45 minutes. On the other hand, the LSTMDF performance is stable after measuring RPPG with six state-of-the-art methods, namely PVM [60], POS [41], PbV [20], G-R [41], Chrom [18], and Green [9].

The results demonstrate experimentally the superiority of the LSTMDF when compared with conventional filters in an intra-dataset scenario. For example, we obtain an MAE of 3.9 bpm over the VIPL-HR database, whereas the traditional filtering improves performance on the same dataset from 10.3 bpm to 7.7 bpm. The cross-dataset approach presents a dependence in the network related to the average signal-to-noise ratio on the RPPG signals, i.e., the signal-to-noise ratio in the training and testing sets should be similar. Moreover, the results demonstrate that a relatively small amount of data is sufficient to successfully train the network and outperform the results obtained by classical filters.

5.2/ Contributions and chapter structure

In this chapter, we analyze the performance of LSTM networks in RPPG signal filtering. Specifically, we propose to use the LSTMDF network. We develop an in-depth study of the RPPG filtering network performance, adding experiments and approaches to determine the limitations and sensitivities of recurrent neural networks, allowing us to understand

the best configuration to train an LSTM-based model to filter RPPG signals.

The main contributions of this chapter include 1) The Long Short-Term Memory Deep Filter: LSTMDF. 2) Experimental demonstration of the advantages of using LSTM networks in RPPG signal filtering with the help of three public-domain databases in intra-dataset and cross-dataset scenarios. 3) Comparison between two sequence-to-sequence models, namely many-to-one and many-to-many. 4) Analysis of the stability of an LSTM-based filter in six state-of-the-art RPPG signal estimation methods: PVM [60], POS [41], PbV [20], G-R [41], Chrom [18], and Green [9]. 5) Analysis of the limitations and sensitivities involved in using an LSTM-based filter in the RPPG filtering task. 6) Insights from the LSTM method when evaluating pulse rate variability metrics.

The remainder of this chapter is organized as follows: In section 5.3, we explain the proposed LSTM-based filter with its two approaches: many-to-one and many-to-many, which are used in the RPPG signal filtering problem. In section 5.4, we show the network implementation details and the protocols used to understand the limitations and sensitivities of the LSTM-based network. In section 5.5, we discuss some perspectives of the pulse rate variability metrics when using the LSTMDF network. Finally, in section 5.6, we present the conclusions of this chapter.

5.3/ PROPOSED METHOD

The RPPG estimation and filtering workflow used in this chapter is presented in Figure 5.1. As a first step, a face tracking procedure is made, taking into account only the pixels of the skin of the face (RoI selection), followed by a spatial averaging process. Then, an RGB color channel combination is performed according to an RPPG estimation method. Afterward, the signal noise is removed by a filter, and finally, the physiological parameter estimation on the filtered RPPG signal is computed through a Fast-Fourier-Transform (FFT) based analysis.

To perform the RPPG signal filtering, in this chapter, we propose the Long Short-Term Deep-Filter. In the rest of this section, we will explain the LSTMDF filter with its two approaches, many-to-one (MTO) and many-to-many (MTM), followed by the organization of the databases to the network training.

5.3.1/ LSTM-BASED DEEP-FILTER

Having a T-frame video, we can compute a spatial RoI selection and tracking on the face of the subject to perform a spatial averaging between the pixels related to the skin. The result is a triplet of RGB values for each frame $t \in [1, T]$ as $r[t]$, $g[t]$, and $b[t]$. Hence, the vectors $\mathbf{r}$, $\mathbf{g}$, and $\mathbf{b}$ represent the RGB values for the full video. This way, being ζ a color channel combination function applied by an RPPG estimation method, the RPPG vector $\mathbf{y}$ is presented in equation 5.1:

$$\mathbf{y} = \zeta(\mathbf{r}, \mathbf{g}, \mathbf{b}). \tag{5.1}$$

With the signal to be filtered, we can use an LSTM network constituted by an input, a memory block, and an output. The memory block has the following parts: one *input gate*

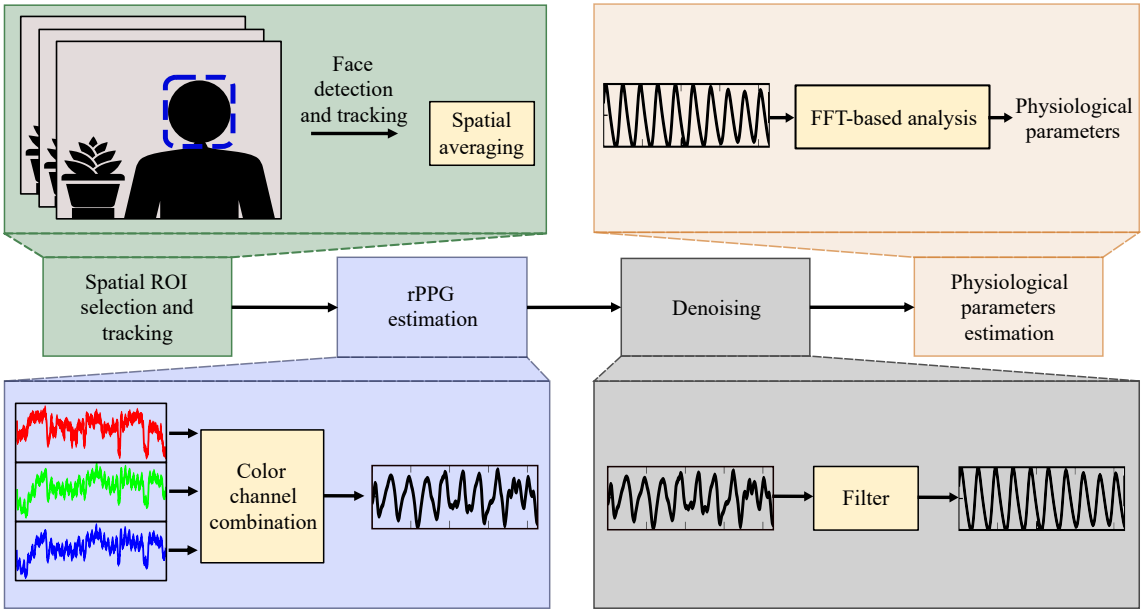

Figure 5.1: RPPG estimation and filtering workflow. A Spatial RoI selection is performed, then the RPPG signal is estimated and filtered. Finally, the physiological parameters estimation is performed through a FFT-based analysis.

that learns which information is stored in the memory block, a *forget gate* that learns how much information is forgotten from the memory block, and finally, an *output gate* that takes care of learning when the collected information can be used. We can consider the RPPG filtering as an sequence-to-sequence regression problem in univariate time series data. The two LSTM network approaches proposed are explained in the following subsections.

5.3.2/ MANY-TO-ONE: LSTMDF MTO

In this approach, multiple inputs allow to predict one single output, specifically, to forecast the next value based on the inputs. For this reason, we can use a L-length sliding window with a step of one frame on the RPPG vector $\mathbf{y}$. Then, we use the MTO filter φ_o to obtain the filtered RPPG vector $\hat{\mathbf{y}}$ as shown in equation 5.2:

$$\hat{y}[j] = \begin{cases} y[t] & t \in [1, L], \\ \varphi_o(\mathbf{y}[t-L:t-1]; \omega) & \text{others}, \end{cases} \tag{5.2}$$

where ω is the set of all learnable parameters already trained, and $t \in [1, T]$. Note that the first estimated sample given by the filter is actually at $L+1$.

5.3.3/ MANY-TO-MANY: LSTMDF MTM

The MTM approach uses multiple inputs to give multiple outputs, specifically the number of inputs is the same number of outputs. Similarly, as in MTO, we use a L-length sliding window with a step of one frame.

$$\tilde{\mathbf{y}}[t : t + L - 1] = \varphi_m(\mathbf{y}[t : t + L - 1]; \omega) \tag{5.3}$$

with $t \in [1, t - L + 1]$. The L-length outputs $\tilde{\mathbf{y}}[t : t + L - 1]$ are later combined to create a single filtered RPPG signal $\hat{\mathbf{y}}$ using the overlap-add procedure explained in section 4.1.1 by the Equation 4.2.

Below we explain the process to create the training set from signals of different duration.

5.3.4/ LSTMDF TRAINING: DATASET BUILDING

It is important to emphasize that the following procedure has to be performed only on the training subjects when using a subject-independent evaluation. The test subjects must be treated individually. To create the training set for the LSTM network, we can consider the RPPG signal $\mathbf{y}_i$ related to the video i with length T_i, where $i \in [1, N]$ and N is the number of videos. Then, for the N videos, we can use an L-length sliding window with a step of one frame to create the training matrices $\mathbf{Y}$ and $\mathbf{G}$ (vector $\mathbf{G}$ for MTO). In MTM and MTO, $\mathbf{Y}$ is composed of RPPG signals with a fixed length L. $\mathbf{G}$ represents the ground truth. In MTM, $\mathbf{G}$ is a matrix of PPG signals of same size than $\mathbf{Y}$. $\mathbf{G}$ is a vector in MTO approach. $\mathbf{Y}$ and $\mathbf{G}$ are given as the training set for the LSTM-based filter. The whole procedure is presented in Algorithms 1 and 2.

ALGORITHM 1: RPPG training dataset building for the LSTMDF MTO network

Input : **y** – original RPPG signals to train
g – original ground truth signals to train
L – sliding-window length
N – list of available signals
T – list of signal sizes

Output: Y – RPPG matrix
G – ground truth vector

```
1 c ← 1                                                    /* counter */
2 for i ← 1 : N do
3     for t ← 1 : T_i − L − 1 do
4         Y[c] ← y_i[t : t + L];
5         G[c] ← g_i[t + L + 1];
6         c ← c + 1;
7     end
8 end
```

ALGORITHM 2: RPPG training dataset building for the LSTMDF MTM network

Input : **y** – original RPPG signals to train
g – original ground truth signals to train
L – sliding-window length
N – list of available signals
T – list of signal sizes

Output: Y – RPPG matrix
G – ground truth matrix

```
c ← 1                                                    /* counter */
for i ← 1 : N do
    for t ← 1 : T_i − L do
        Y[c] ← y_i[t : t + L];
        G[c] ← g_i[t : t + L];
        c ← c + 1;
    end
end
```

5.3.5/ NETWORK ARCHITECTURE

The MTO architecture is presented in Figure 5.2, and the MTM in Figure 5.3. L is the input length, b is the batch size, and Return Sequence (RS) is an argument that decides whether a layer outputs each time step or its final time step. A dropout step is done on the first three layers to avoid the overfitting problem. The three LSTM layers contain 125 units. The experiments performed to find the configuration of the number of units and layers are presented in the annexes section A.3.

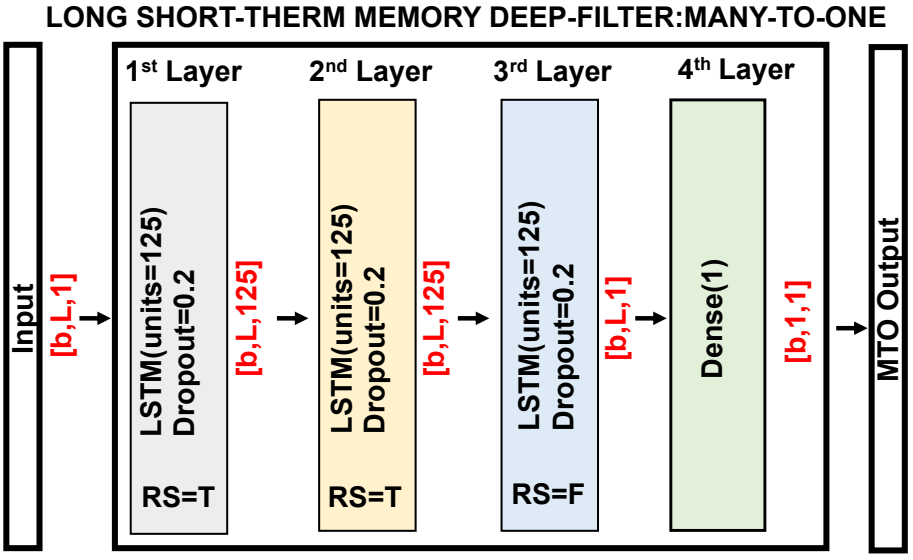

Figure 5.2: MTO architecture. Numbers in red represent the size of data in each layer. RS=Return sequences, T=True, F=False, b=batch size, L=sliding window length.

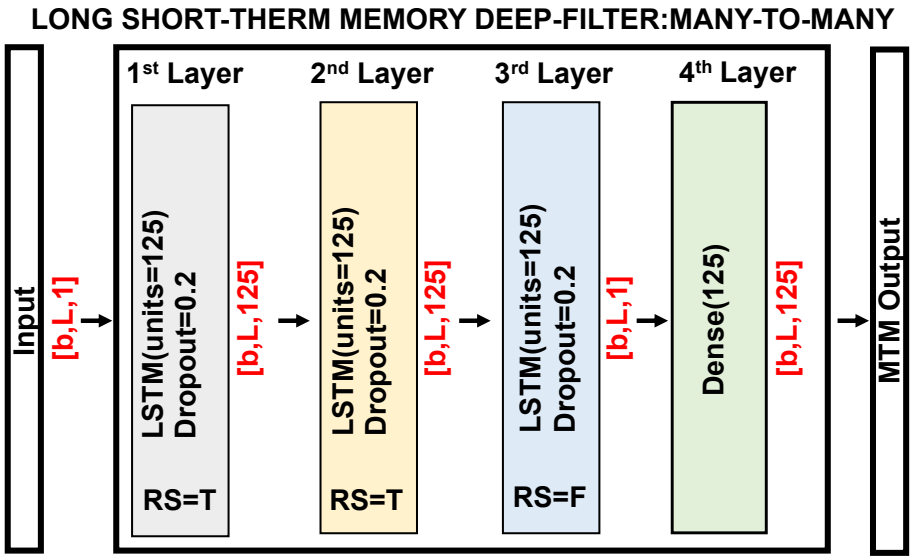

Figure 5.3: MTM architecture. Numbers in red represent the size or transformed inputs in each layer. RS=Return sequences, T=True, F=False, b=batch size, L=sliding window length.

5.4/ PERFORMANCE ANALYSIS

Following the pre-processing procedure explained in section 4.2.1, only the reliable ground truth signals are selected, the Table 5.1 contains the parameters related to RPPG signals acquired by the PVM method (PVM-RPPG) [60] from the three databases: MMSE-HR, VIPL-HR, and COHFACE. RPPG-SNR is the average of the SNR coefficient of RPPG signals. Figure 5.4 depicts examples of RPPG signals from the three databases used. Note how the quality of the acquired RPPG signals varies for each database. This is due to the nature of the scenarios where the videos were acquired for each database, i.e., luminance, movement, and other factors present during the acquisition.

Table 5.1: Parameters of the PVM-RPPG signals presented in the MMSE-HR, VIPL-HR, and COHFACE datasets

	MMSE-HR	**VIPL-HR**	**COHFACE**
RPPG-SNR [dB]	7.65 ± 3.78	1.07 ± 4.25	-0.96 ± 4.17
Duration [minutes]	64	1,114.5	166
Number of signals	98	2256	164

In this section, we propose experiments to study the behavior of an LSTM network in the RPPG filtering task. Thus, we will better understand the strengths and sensitivities of the proposed method. The VIPL-HR database was designed to estimate RPPG signals and contains a large amount of data during various scenarios. Due to this, its noise level measured as the average SNR on the RPPG signals (RPPG-SNR) is 1.07 dB. MMSE-HR and COHFACE, on the other hand, have a considerably smaller amount of data. MMSE-HR

is the database with the best quality in its RPPG signals with an average RPPG-SNR of 7.65 dB, and COHFACE, with an average RPPG-SNR of -0.96 dB. COHFACE is the most challenging database, and its negative RPPG-SNR is perhaps related to the video level compression or the luminance in the scenarios.

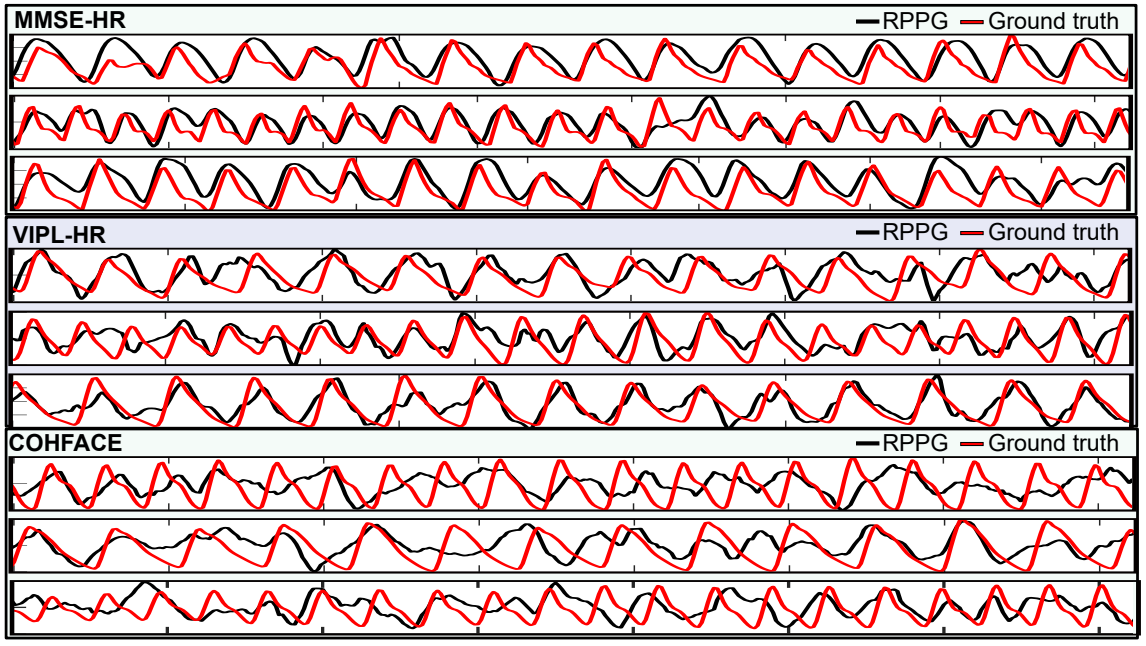

Figure 5.4: RPPG examples. Examples of RPPG signals from the three databases used: MMSE-HR, VIPL-HR, and COHFACE.

5.4.1/ IMPLEMENTATION DETAILS

For detecting the faces, we used the deep-learning-based OpenCV model, used in the Single Shot MultiBox Detector method implemented by Liu *et al.* [37]. Then, the Conaire *et al.* algorithm was used to select the skin pixels [8].

We used a personal computer with the following features: Intel Xeon 2.4 GHz CPU, 16 GB RAM, and a Graphics Processing Unit (GPU) NVIDIA GeForce RTX 2070 GPU. The LSTM network was implemented with *PyTorch* libraries version 1.9.0. The activation functions for the LSTM and Dense layers were hyperbolic tangent and linear, respectively. The loss function was mean squared error, the first two LSTM layers had a dropout equal to 0.2, and the Adam optimizer was set with a learning rate scheduler of 0.002 between 1 and 10 iterations, 0.001 between 11 and 20, 0.0005 between 21 and 50, and 0.0001 for 51 or more iterations. The Glorot uniform initializer, also called Xavier uniform initializer, was used for the random weights initialization.

Table 5.2 depicts the results of the hyperparameter tuning procedure conducted to determine the best values for the number of epochs and batch size. The number of epochs used was 100, with a batch size of 32. The sliding window used during the training setup had length $L = 125$ frames, equivalent to five seconds (25 Hz signals). The network training during the final evaluation took for each dataset approximately: one hour for the MMSE-HR, fourteen hours for VIPL-HR, and three hours for COHFACE. The process of filtering on a one-minute signal takes approximately 1 second.

Batch size	No. epochs	MAE [bpm]	SNR [dB]	TMC
8	30	1.50	**9.69**	0.71
32	30	1.51	9.59	0.70
32	100	**1.32**	9.44	**0.72**
64	100	1.32	9.50	0.70

Table 5.2: LSTMDF hyperparameter tuning: batch size and number of epochs.

5.4.2/ PROTOCOLS

Multiple experiments have been performed to validate using an LSTM network for filtering RPPG signals. We conducted a classical intra-database evaluation followed by a cross-dataset evaluation. Additionally, we present experiments to study in more detail the advantages and limitations of using an LSTM-based network for RPPG filtering. First, we demonstrate that it is not necessary to use a large amount of data to train the network successfully and that there is a clear dependence of the RPPG-SNR average in the RPPG signals during training.

Figure 5.5 shows the distribution made in the databases used to study intra and cross-dataset evaluations. **A** lists the databases used: COHFACE, MMSE-HR, and VIPL-HR. **B** contains the RPPG signals acquired by: PVM [60], POS [41], PbV [20], G-R [41], Chrom [18], and Green [9]. **C** presents the intra-dataset and cross-dataset evaluations with metrics measurement. Figure 5.6 resumes the PVM-VIPL signal generated in Figure 5.5, used in studying the influence of the amount of data and noise on the RPPG signals during the training of the LSTM network, the protocol related to the amount of training is later called *Amount of training data* (top panel), and the protocol related to noise is called *RPPG-SNR dependence* (bottom panel).

We present the results of the experiments comparing the two sequence-to-sequence approaches proposed: MTO and MTM with three classical filters namely bandpass, wavelet and Savitzky-Golay. Similar to [53, 115] we used a 8th order bandpass filtering with cutoff frequencies at 0.7 and 3.5 Hz, wavelet-based filtering using the same parameters as in [95], and the Savitzky-Golay filter with a 9-samples window length and a polynomial order of two[1]. For ease of data visualization, the metrics MAE, E:SDNN, E:RMSSD, E:LF, and E:HF metrics are presented with the vertical axis inverted on the figures; this is done so that the best performances for all metrics are always at the top of the graphs.

In the initial part of each experiment, we analyze the performance of the filters with the metrics typically used in the literature, i.e., the pulse rate metrics, MAE, and r. In addition to these metrics, interested in the effect of smoothing the RPPG signals after the filtering process, we are more interested in the improvements of the signal quality metrics: SNR and TMC.

[1] https://bit.ly/37DSVPY

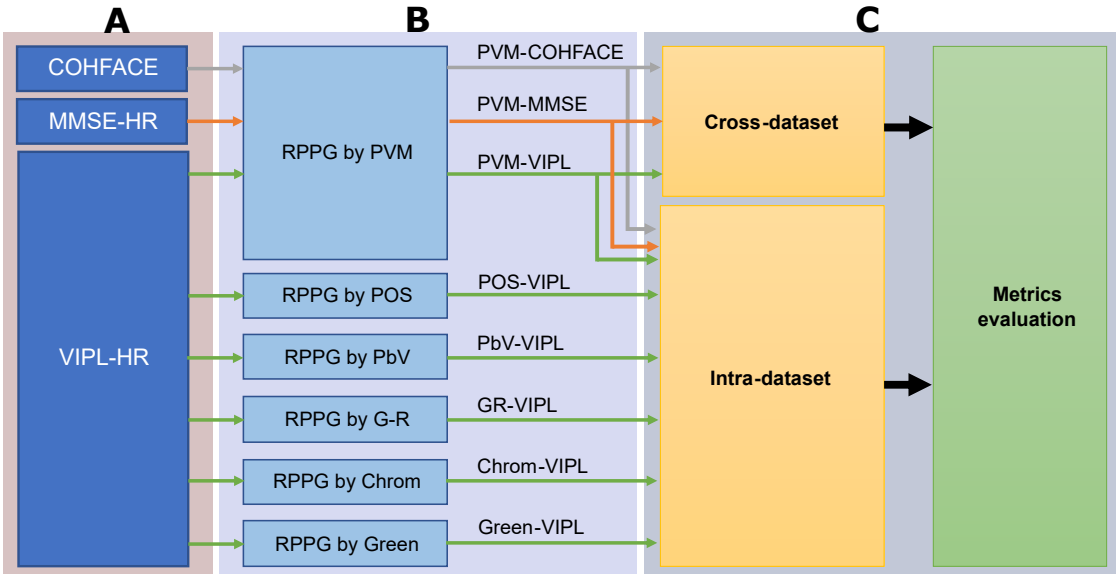

Figure 5.5: Intra and cross-dataset RPPG evaluation. **A** are the list of the databases used. **B** are the RPPG measurement methods used, and **C** are the evaluation criteria along with the measurement of metrics.

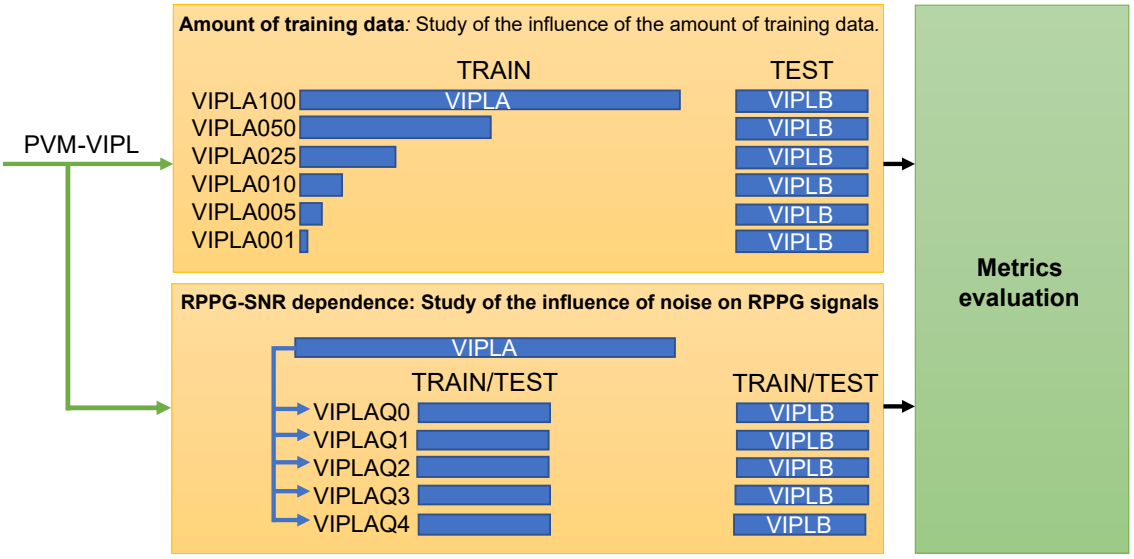

Figure 5.6: Protocols. PVM-VIPL performs the protocols: *Amount of training data* (top panel), and *RPPG-SNR dependence* (bottom panel).

5.4.2.1/ INTRA-DATASET

We used a subject-independent 5-fold cross-validation (CV for short) in this standard evaluation protocol. In the first part of this experiment, we evaluate the performance of the MTO and MTM networks for the PVM-MMSE, PVM-VIPL, and PVM-COHFACE sets. Figure 5.7 presents the results associated with the pulse rate and signal quality metrics in the MMSE-HR, VIPL-HR, and COHFACE databases. MAE is on the top, followed by r, SNR, and TMC. The best results are presented in bold red.

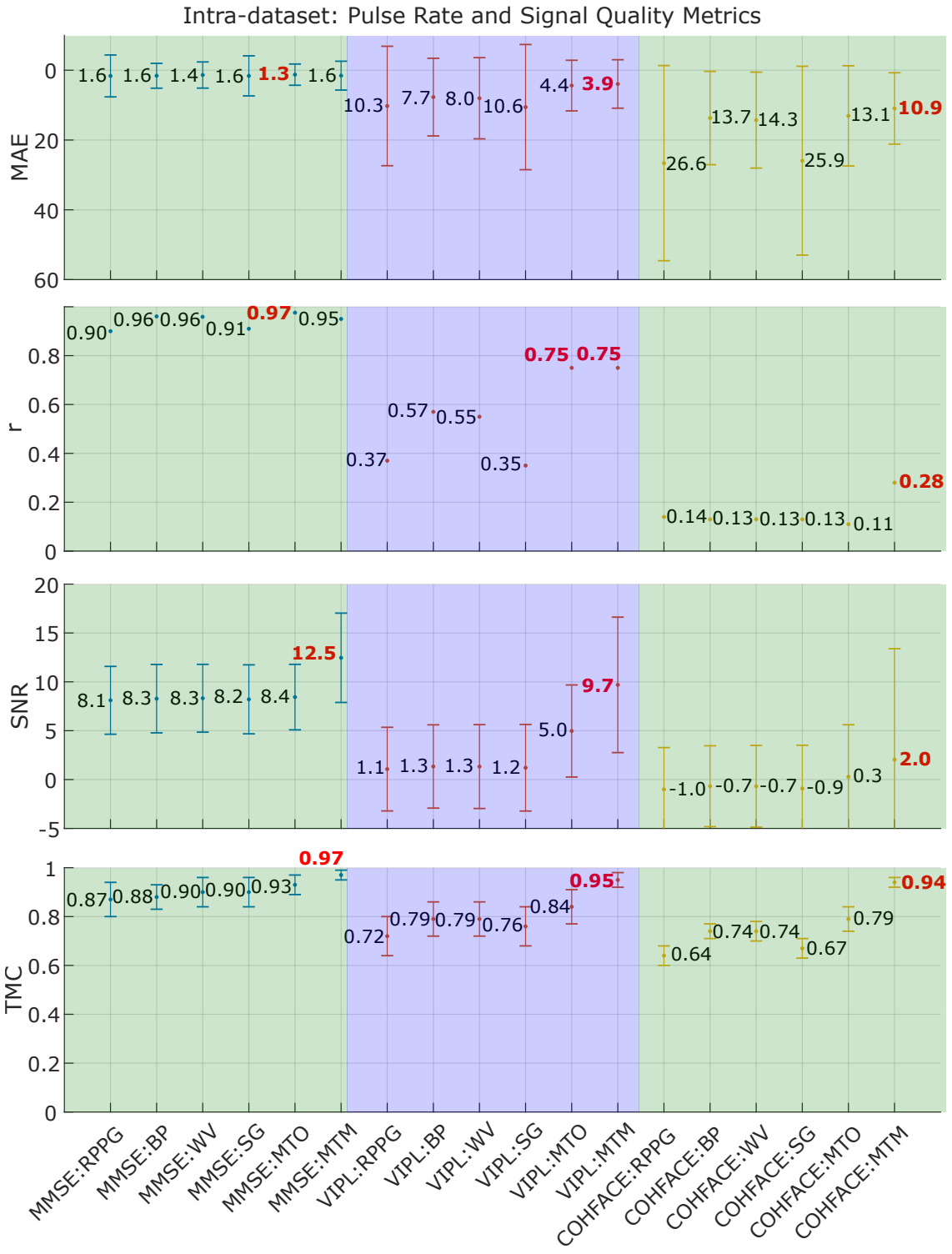

Figure 5.7: Intra-dataset pulse rate and signal quality metrics. Results of the filters: band-pass (BP), wavelet (WV), Savitzky-Golay (SG), LSTMDF MTO and MTM. Best values are presented in bold red.

Figure 5.7 shows the excellent quality of the raw RPPG input signals corresponding to the MMSE-HR database, with an MAE of 1.6 bpm, an r of 0.9, an SNR of 8.1 dB, and a TMC of 0.87. Although the pulse rate metrics seems challenging to overcome by classical filters, this is not a problem for MTO and MTM, with MTO being the best of the methods. Regarding signal quality metrics, the effect of the deep filters is more visible, especially in the SNR metric going from 8.1 dB to 12.5 dB. Although the MMSE-HR database is small (Total duration is 64 minutes), the excellent quality of the input video makes the raw RPPG signals easy to learn for the network. Hence, the deep filtering process is successful.

COHFACE raw RPPG signals are the opposite of MMSE-HR. With an MAE of 26.6 bpm, an r equal to 0.14, SNR of -1 dB, and TMC of 0.64, COHFACE presents the noisiest signals compared with the other databases, and this may be associated with the database level compression. RPPG signals with a high level of noise make the task difficult for any filter. In the r metric, all classical filters fail, while MTO seems to depend on the quality of the training signals. Even so, MTM improves the pulse rate and signal quality metrics, especially in SNR, being the best of the five filters.

The results in VIPL-HR are perhaps the most promising for MTO and MTM filters. The classical BP and WV filters improve the raw signal but not as much as the deep filters MTO and MTM. MTM is the best filter, decreasing MAE by 62% from 10.3 bpm to 3.9 bpm, and increasing r by 102% from 0.37 to 0.75. Regarding signal quality metrics, the result is equally promising, increasing SNR from 1.1 dB to 9.7 dB and the TMC from 0.72 to 0.95. The good performance of the LSTM-based filters in this particular dataset is due to a large number of signals available, allowing the network to learn in a greater variety of signals than the MMSE-HR and COHFACE databases. Therefore, in SNR and TMC metrics, we notice an improvement in the quality of the filtered signals for all three databases, especially in VIPL-HR.

In conclusion, both LSTMDF filter approaches can improve the pulse rate metrics in an intra-dataset experiment, except r for the MTO filter in a noisy database such as COHFACE. It is also important to note that for the three databases, the best metrics are those acquired by MTO and MTM. However, MTM increases the quality of the signals to a greater extent.

In the second part of this experiment, to analyze the filter robustness with RPPG signals acquired by different state-of-the-art methods, we chose the PVM-VIPL, POS-VIPL, PbV-VIPL, GR-VIPL, Chrom-VIPL, and Green-VIPL sets. Table 5.3 contains the RPPG-SNR average of the signals acquired in each method, where the same 2,256 subjects have a total RPPG signal duration of 1,114.5 minutes for the six sets. The RPPG-SNR average value is related to the signal quality.

Table 5.3: RPPG-SNR average in the VIPL-HR signals acquired by the methods: PVM, POS, PbV, G-R, Chrom and Green.

	PVM-VIPL	**POS-VIPL**	**PbV-VIPL**	**GR-VIPL**	**Chrom-VIPL**	**Green-VIPL**
RPPG-SNR [dB]	1.07 ± 4.25	0.32 ± 3.71	-0.29 ± 3.3	-0.51 ± 4	-0.92 ± 3.4	-1.36 ± 4.05

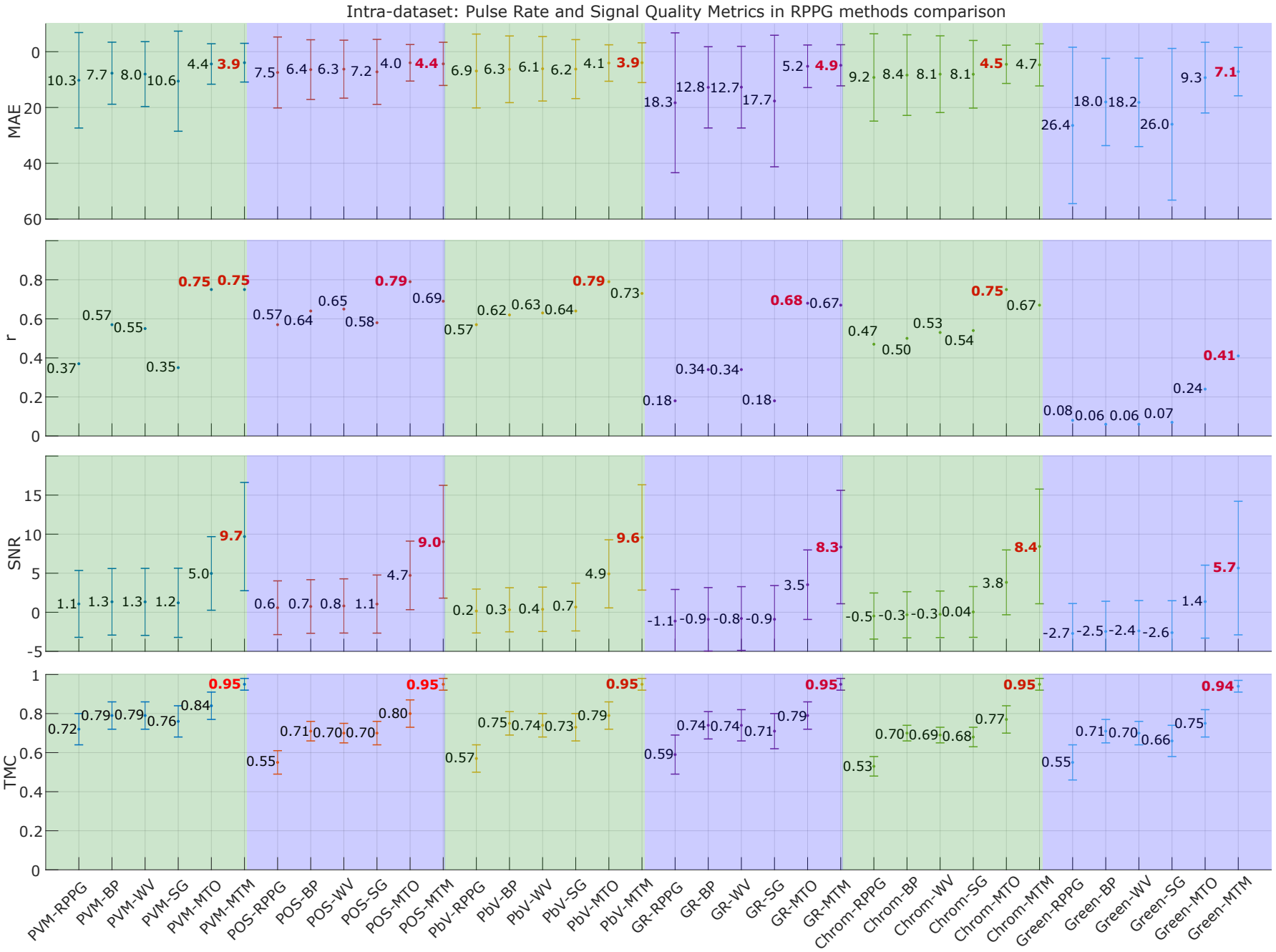

Figure 5.8: RPPG methods comparison. Results of pulse rate and signal quality metrics: MAE, r, SNR, and TMC in the *intra-dataset* scenario.

Figure 5.8 compares the six state-of-the-art RPPG signal acquisition methods after using the MTO and MTM filters. It shows the pulse rate and signal quality metrics (best results are in red bold). In this Figure, it can be seen that the LSTM-based filters always give the best results compared with the filtering methods used in the literature.

Regarding pulse rate metrics, MTO and MTM improve the MAE and r metrics. The only exception is presented in the *Green* method (the set with the noisiest signals), which seems to be more challenging for the deep filters. Interestingly, in this same set, the MTM network manages to decrease MAE by about 73% going from 26.4 bpm to 7.1 bpm Regarding signal quality metrics, MTM is the best filtering method, outperforming all other filters, including MTO. Even COHFACE shows a considerable improvement in the quality of its signals.

As conclusion, the two LSTMDF approaches MTO and MTM are stable and able to work with RPPG signals from different algorithms, even with the noisier method *Green*. In general, we could say that in an *intra-dataset* scenario, both sequence-to-sequence approaches outperform the results provided by the filters found in the literature; however, MTM turns out to be better than MTO, especially when it comes to improving the quality of the signal.

Next we will present the analysis in a more challenging scenario such as cross-dataset for the three proposed databases.

5.4.2.2/ CROSS-DATASET

A realistic evaluation of the filter is to train the system using one database and test it using other databases: this is the *cross-dataset* protocol. For this reason, the RPPG signals of the three sets, PVM-MMSE, PVM-VIPL, and PVM-COHFACE, were used individually as a training set to perform the test on the two remaining sets of data.

Figure 5.9 shows the results of the metrics related to estimating the pulse rate and signal quality during the *cross-dataset* experiment. On the top of Figure 5.9 are the pulse rate metrics, followed by the signal quality metrics.

The *cross-dataset* protocol is particularly interesting. Based on Figure 5.9, there is a dependence between the signals used during training and testing. This behavior may be because each dataset has its own range of RPPG signal quality. For example, the MMSE-HR database has an RPPG-SNR average of 7.65 ± 4 dB, indicating that the signals are mostly of good quality, VIPL-HR has a larger amount of data, and its RPPG-SNR average is 1.04 ± 4 dB, COHFACE on the other hand, has an RPPG-SNR average of -0.96 ± 4 dB. This signal quality difference between the three datasets is also visible in Figure 5.4. Therefore, if there is a dependence between the quality of the signals used for training and testing: first, training the network on high-quality signals (MMSE-HR) and testing on low-quality signals (COHFACE) or vice versa should give poor performance, and second, training on a broad spectrum of good and poor quality signals (VIPL-HR) should give good performance.

Analyzing the metrics in Figure 5.9, we notice that when training in VIPL-HR and testing in MMSE-HR, MTO performs better in r, SNR, and TMC than other filters, but in MAE, it is the second-best value after WV. MTM improves the signal quality metrics but fails to outperform the other methods in the pulse rate measurement. Training in VIPL-HR and testing in COHFACE show how the LSTM-based filters outperform the other filters.

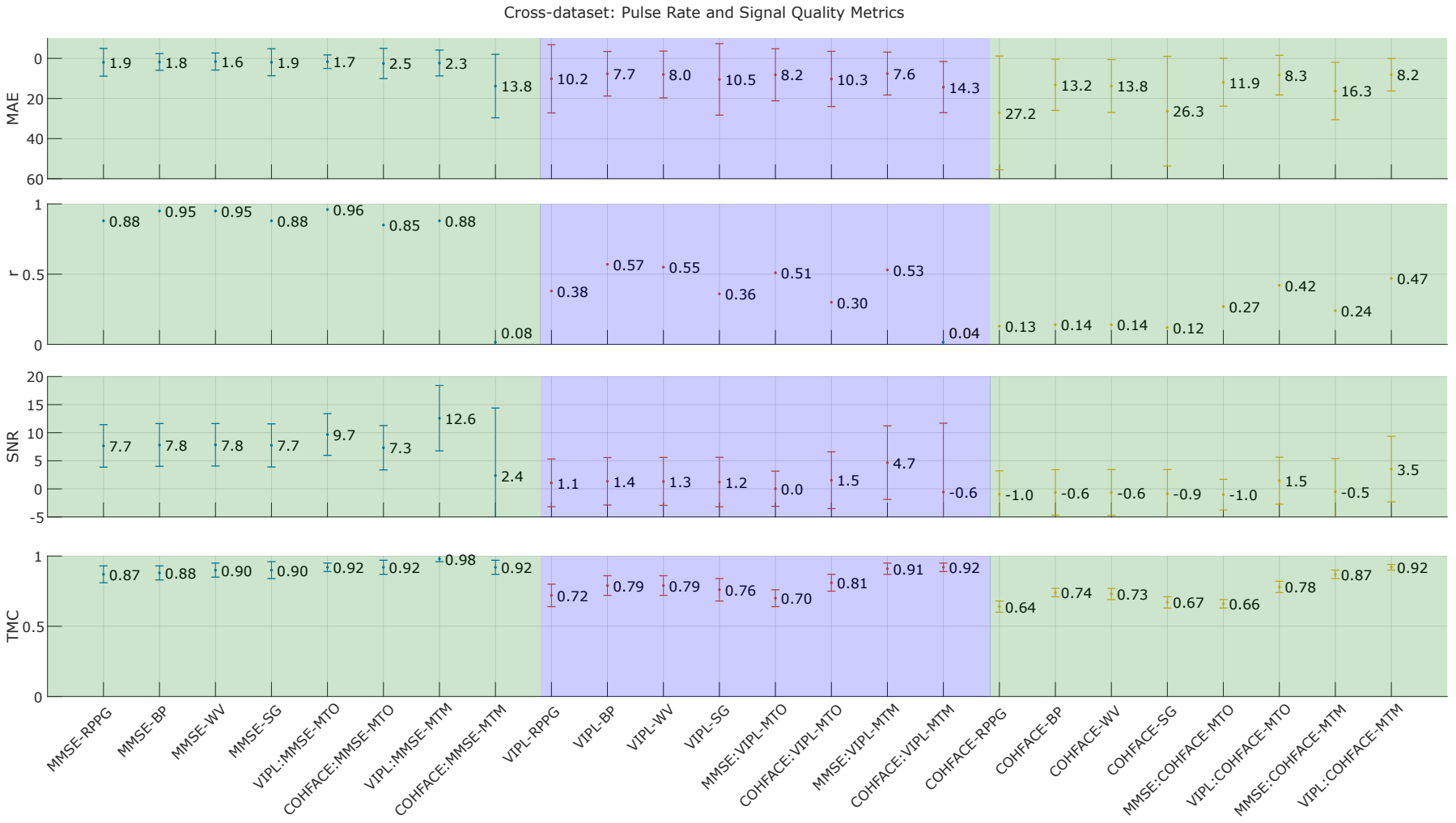

Figure 5.9: Cross-dataset. Results of pulse rate and signal quality metrics: MAE, r, SNR and TMC in the *cross-dataset* experiment. Experiments on the x-axis are listed as "Training database: Test database - Filtering method" except for classical filters

The MTM approach is the best of the two proposed. Training in MMSE-HR and testing in VIPL-HR allows the MTM filter to present the best result in MAE, a value very close to the best values given by BP and WV in r, and the best results in SNR and TMC. MTO, on the other hand, gives values close to the best values in MAE and r but decreases the values of SNR and TMC. Continuing with the training in MMSE-HR but testing in COHFACE, we can see that MTO has the best results for the MAE and r metrics, but some values are lower in SNR and TMC. MTM, on the other hand, has a contrary behavior, it improves the other filters in SNR and TMC, and although it improves the result of r, it does not manage to overcome the BP and WV filters in MAE. Therefore, neither of the proposed methods is sufficiently robust to over-perform the four metrics.

The *cross-dataset* scenario shows that for the two LSTMDF approaches, there is a dependence on training and test data related to the quality of RPPG signals, reflected in the RPPG-SNR value of the databases. We will corroborate this hypothesis in the *RPPG-SNR dependence* experiment (section 5.4.2.4). Nevertheless, before analyzing the dependence on signal quality, we will analyze the dependence on the amount of training data in the next section.

5.4.2.3/ AMOUNT OF TRAINING DATA

Indeed, in deep-learning-based applications, the amount of data used during the training of a network plays a fundamental role in its performance. For example, in computer vision applications, generalization tends to improve along with the size of the training sets [78]. However, there is no definitive answer to whether the amount of data during training improves all deep-learning applications. For this reason, it is critical to define in each deep-learning-based application if there is a dependency on the amount of data used during the training.

In this experiment, we wanted to study the sensitivity of the LSTM-based filter to the amount of information given during training. A stratified Train-test-split in PVM-VIPL was made, having the VIPLA and VIPLB sets with 80% and 20% of signals, respectively. Consequently, we used the VIPLB set as a testing set and VIPLA as the training one. This way, we take the set VIPLA and perform a stratified division at 100%, 50%, 25%, 10%, 5% and 1% of the data, calling these training sub sets VIPLAx with x=[100,050,025,010,005,001] (Figure 5.6 top panel). Note that the set VIPLA100 is indeed the same VIPLA, but this notation is used only to easily appreciate each subset's characteristics. The stratified division guarantees subsets with a balanced RPPG-SNR. This last detail is important as we will see in *RPPG-SNR dependence* that training and testing sets with unbalanced average RPPG-SNR may change the network's performance.

Table 5.4 contains the parameters of the RPPG signals used in this experiment, the average RPPG-SNR, the total duration of the signals in minutes, and the number of signals. Note how the average RPPG-SNR is similar for the seven sub sets; this ensures that the results obtained in this experiment are not affected by the quality of the signals but only by the quantity.

Table 5.4: Characteristics of the RPPG signals in *Amount of training data*

	VIPLA100	**VIPLA050**	**VIPLA025**	**VIPAL010**	**VIPLA005**	**VIPLA001**	**VIPLB**
RPPG-SNR [dB]	1.27 ± 4.07	1.25 ± 4.04	1.21 ± 4.0	1.15 ± 4.13	0.99 ± 4.11	1.03 ± 3.92	1.09 ± 4.28
Duration [minutes]	891.78	446.11	223.5	88.87	45.1	9.04	222.7
Number of signals	1084	902	451	180	90	18	452

Figure 5.10 depicts the results of pulse rate and signal quality metrics: MAE, r, SNR, and TMC for this experiment. Regarding pulse rate metrics, MTO shows that when decreasing the data from 100% to 50% and 25%, MAE and r remain stable. When decreasing from 10% and 5%, MAE is stable, but the value of r starts to decrease. Finally, when having 1% of data, MAE and r present the lowest performance. Concerning signal quality metrics, MTO still has higher values than the other classic filters, except for the TMC metric. In SNR, we see that the best performance is obtained using more data.

In pulse rate metrics, the MTM filter shows a higher sensitivity to the amount of training data than the MTO. Even though the MAE value remains relatively constant, r starts to decrease from taking percentages equal to and less than 25%. In the case of signal quality metrics, MTM shows a higher performance than MTO regardless of the amount of training data, and it is always higher than the conventional filters.

Interestingly, both sequence-to-sequence approaches can be trained with a set of RPPG signals similar to those in VIPL005 (only 45 minutes, 90 signals). Thus, we can start to obtain better results than conventional filters BP, WV, and SG in a test set with similar RPPG-SNR values. Therefore, a large amount of data is not needed to train an LSTM-based deep filter successfully. Now that we know the minimum amount of data needed to train MTO and MTM networks, we can continue analyzing the quality of RPPG-SNR signals.

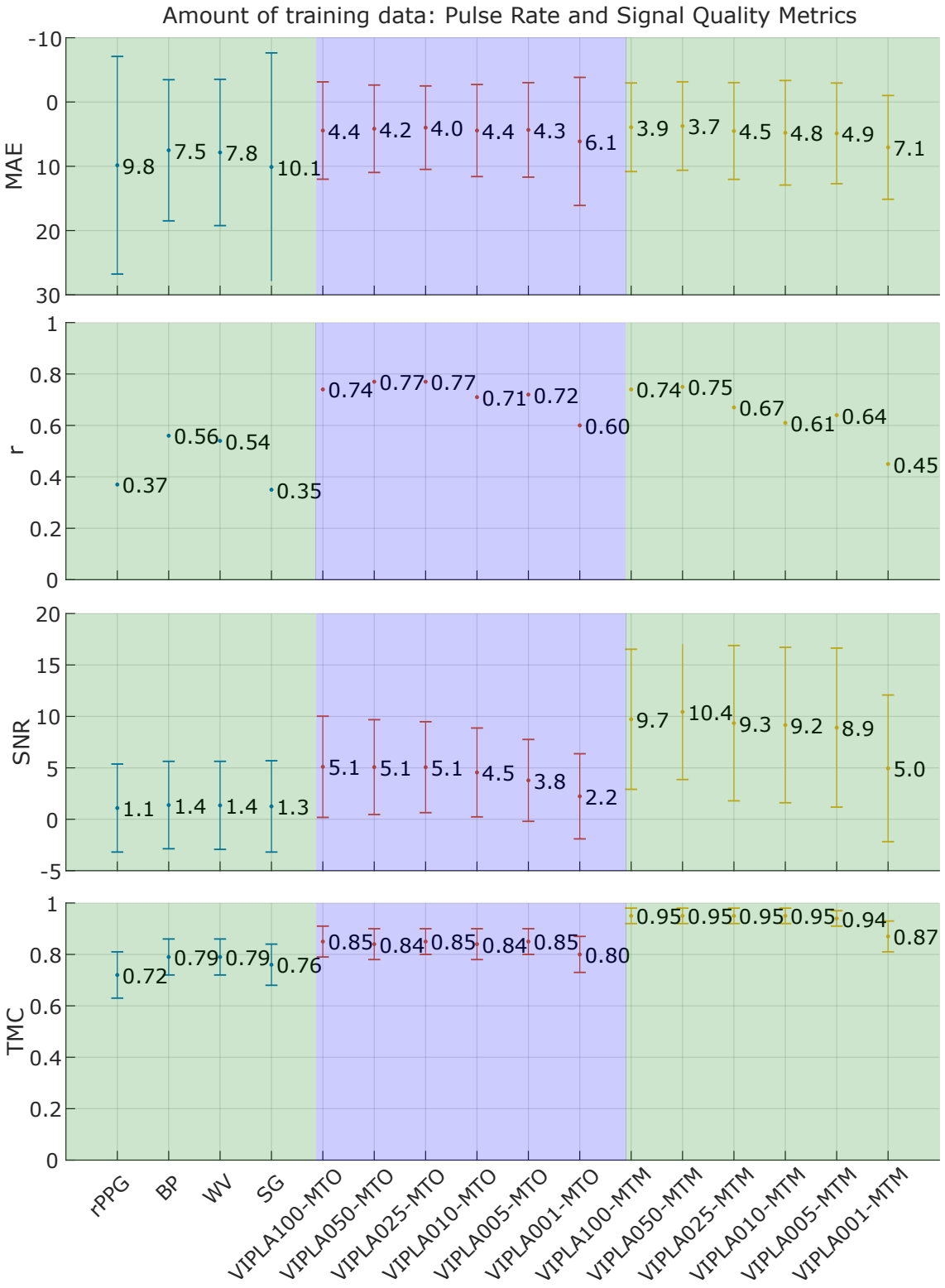

Figure 5.10: Amount of training data. Results of pulse rate and signal quality metrics: MAE, r, SNR, and TMC in the *Amount of training data* experiment.

5.4.2.4/ RPPG-SNR DEPENDENCE

Inspired by the results found in the *cross-dataset* protocol, we propose this new experiment where we wanted to simulate a scenario where the RPPG-SNR in the training and testing sets were different. This scenario may happen when RPPG signals are estimated from different approaches, as we did in *intra-dataset* with PVM-VIPL, POS-VIPL, PbV-VIPL, GR-VIPL, Chrom-VIPL, and Green-VIPL, but also when comparing different databases as we did in *cross-dataset*.

In order to assign a label representing the RPPG-SNR level, the SNR average of each subject (SNR_i with $i \in [1, N]$) was measured and assigned a label e_i: 0, 1, 2, 3, and 4, where 0 indicates a low value of RPPG-SNR and 4 a high one. We relate the value of RPPG-SNR to the quality of the signals, and to define the RPPG-SNR thresholds for the labels, the maximum and minimum SNR values were taken from all subjects, and this range was divided among h-length 5 sub-ranges. This RPPG-label-assignment procedure is defined in equation 5.4:

$$e_i = \begin{cases} 0 & \mathrm{SNR}_i < \min(\mathrm{SNR}) + h, \\ 1 & \min(\mathrm{SNR}) + h \leq \mathrm{SNR}_i < \min(\mathrm{SNR}) + 2h, \\ 2 & \min(\mathrm{SNR}) + 2h \leq \mathrm{SNR}_i < \min(\mathrm{SNR}) + 3h, \\ 3 & \max(\mathrm{SNR}) - 2h \leq \mathrm{SNR}_i < \max(\mathrm{SNR}) - h, \\ 4 & \mathrm{SNR}_i \geq \max(\mathrm{SNR}) - h, \end{cases} \tag{5.4}$$

where h is given by equation 5.5:

$$h = \frac{\max(\mathrm{SNR}) - \min(\mathrm{SNR})}{5} \tag{5.5}$$

Using these labels, we generated five new sets VIPLAQi with i=[0,1,2,3,4] (Figure 5.6 bottom panel). Where VIPLAQ0 represents a set of data with signals mostly of good quality (high RPPG-SNR), the four remaining new sets decrease in quality until they reach VIPLAQ4, whose signals mostly have bad quality (low RPPG-SNR). The first part of this experiment consists of taking the VIPLAQi as training sets and VIPLB as a testing set, and in the second part, the VIPLB set was chosen as the training set to be tested in VIPLAQi.

The characteristics of the RPPG signals used in this experiment are presented in Table 5.5, the average SNR of the RPPG signals, the total duration, and the number of signals. Note how the duration and number of signals for the VIPLA sets are balanced, while the more considerable variation is found in RPPG-SNR due to the quality of the videos and the RPPG estimation method.

Table 5.5: Characteristics of the RPPG signals in *RPPG-SNR dependence* protocol.

	VIPLAQ0	**VIPLAQ1**	**VIPLAQ2**	**VIPLAQ3**	**VIPLAQ4**	**VIPLB**
RPPG-SNR [dB]	6.42 ± 2.48	5.64 ± 3.15	1.89 ± 6.23	0.73 ± 5.56	-2.69 ± 2.66	1.09 ± 4.28
Duration [minutes]	195.22	196.42	195.22	198.3	197.56	222.7
Number of signals	400	400	397	407	400	452

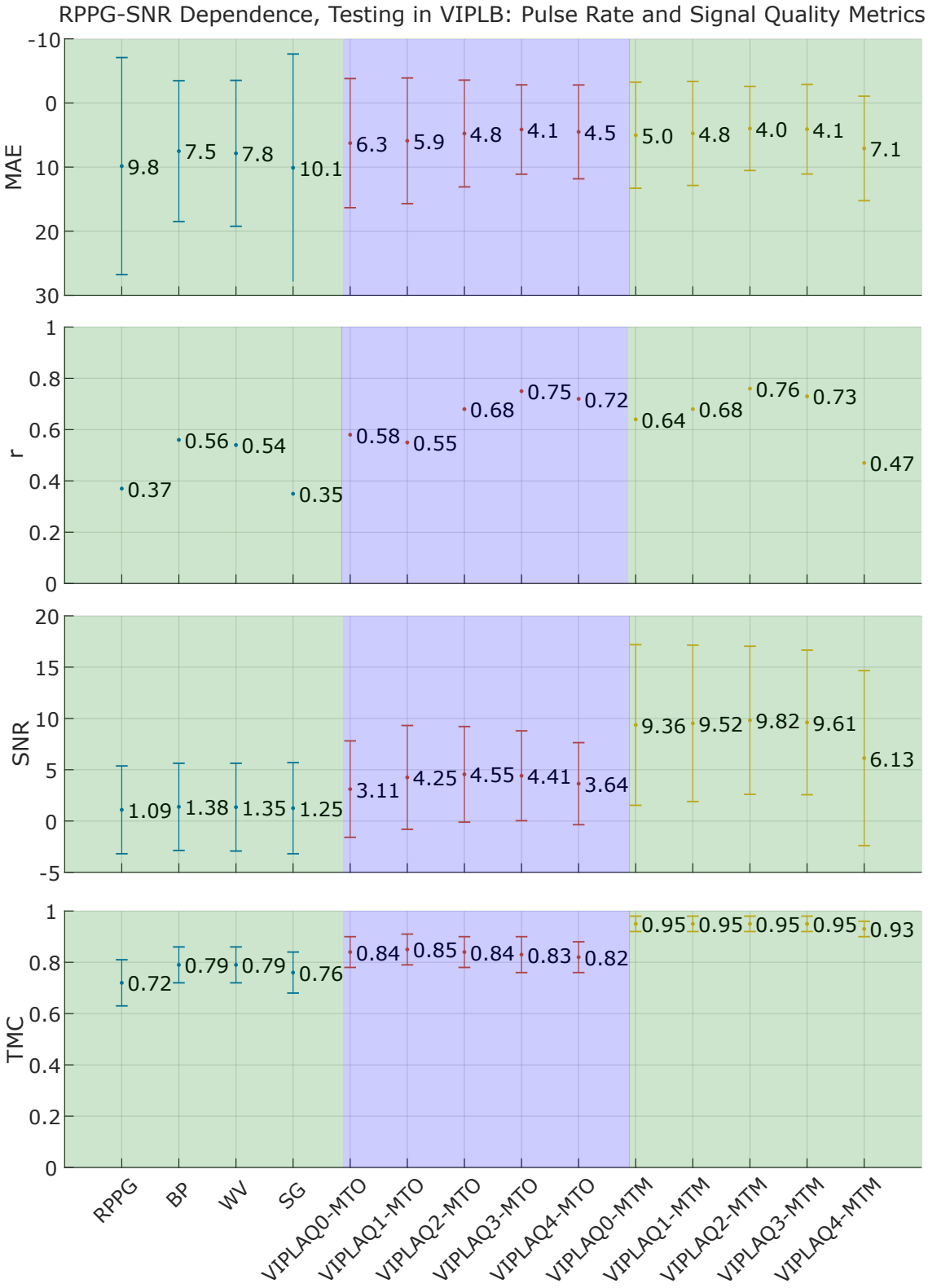

Figure 5.11: RPPG-SNR dependence during training. Results of pulse rate and signal quality metrics: MAE, r, SNR and TMC in the *RPPG-SNR dependence* experiment. Training in VIPLAQ0, VIPLAQ1, VIPLAQ2, VIPLAQ3, and VIPLAQ4, and testing in VIPLB.

Figure 5.11 presents the pulse rate and signal quality metrics for the first part of this experiment, where we trained the LSTM-based networks in VIPLAQi, i=[0,1,2,3,4] and tested them in VIPLB. In the pulse rate metrics presented in the experiment *RPPG-SNR dependence*, the MTO filter does present dependence on the RPPG-SNR during training; MAE values improve from VIPLAQ0 to VIPLAQ3, and the results decrease only a little in VIPLAQ4, r presents a similar behavior by having better results in the training sets with low values of RPPG-SNR than in the sets with the higher ones. The TMC metric remains stable, with the VIPLQ1 training set being the best choice. On the other hand, SNR has a more evident dependence on the quality of the training signals; this metric performs better with a dataset close to the test one, such as VIPLQ2 and VIPLQ3; specifically, SNR begins to decrease as the signal quality becomes higher or lower than these values.

Regarding pulse rate metrics, the MTM filter gives lower performance when training in the sets of signals with the lowest and highest indices of RPPG-SNR (VIPLAQ4 and VIPLAQ0, respectively), especially when training with VIPLQ4. The highest performances are presented in VIPLAQ2 and VIPLAQ3 for MAE and r. In signal quality, SNR and TMC seem to be sensitive to the training set VIPLAQ4; in the other four training sets, their values are constant and even higher than those presented by the MTO approach.

As a conclusion of the first part of this experiment, we can say that using the VIPLB test set whose RPPG-SNR average is 1.09 ± 4.28 dB, the two overall best training sets are VIPLAQ2 and VIPLAQ3. These two sets coincide as they are the closest in quality of the RPPG signals with an RPPG-SNR average of 1.89 ± 6.23 dB and 0.73 ± 5.56 dB respectively. Therefore, there is a dependence on the RPPG-SNR average in the training and testing sets. Specifically, the RPPG-SNR in the training set should be similar to that of the test set to reach the best results.

In the second part of this experiment, the LSTM network is trained in VIPLB to be tested in VIPLAQi, i=[0,1,2,3,4]. Figure 5.12 depicts the results related to the pulse rate and signal quality metrics. As expected, for all metrics, the initial values measured in the raw RPPG signals depend directly on the RPPG-SNR level, i.e., VIPLQ0 presents the best metrics in the raw RPPG signals; these start to decrease as the VIPLQ4 data set is reached. The only exception is presented in TMC given by MTM, where the values are independent of the testing set.

From the two experiments proposed in this section, we can infer that there is a dependence on the quality of the RPPG-SNR signals. If the LSTM training can be ensured in a dataset with a high and low average RPPG-SNR signal balance, there will be an improvement in the smoothed signals.

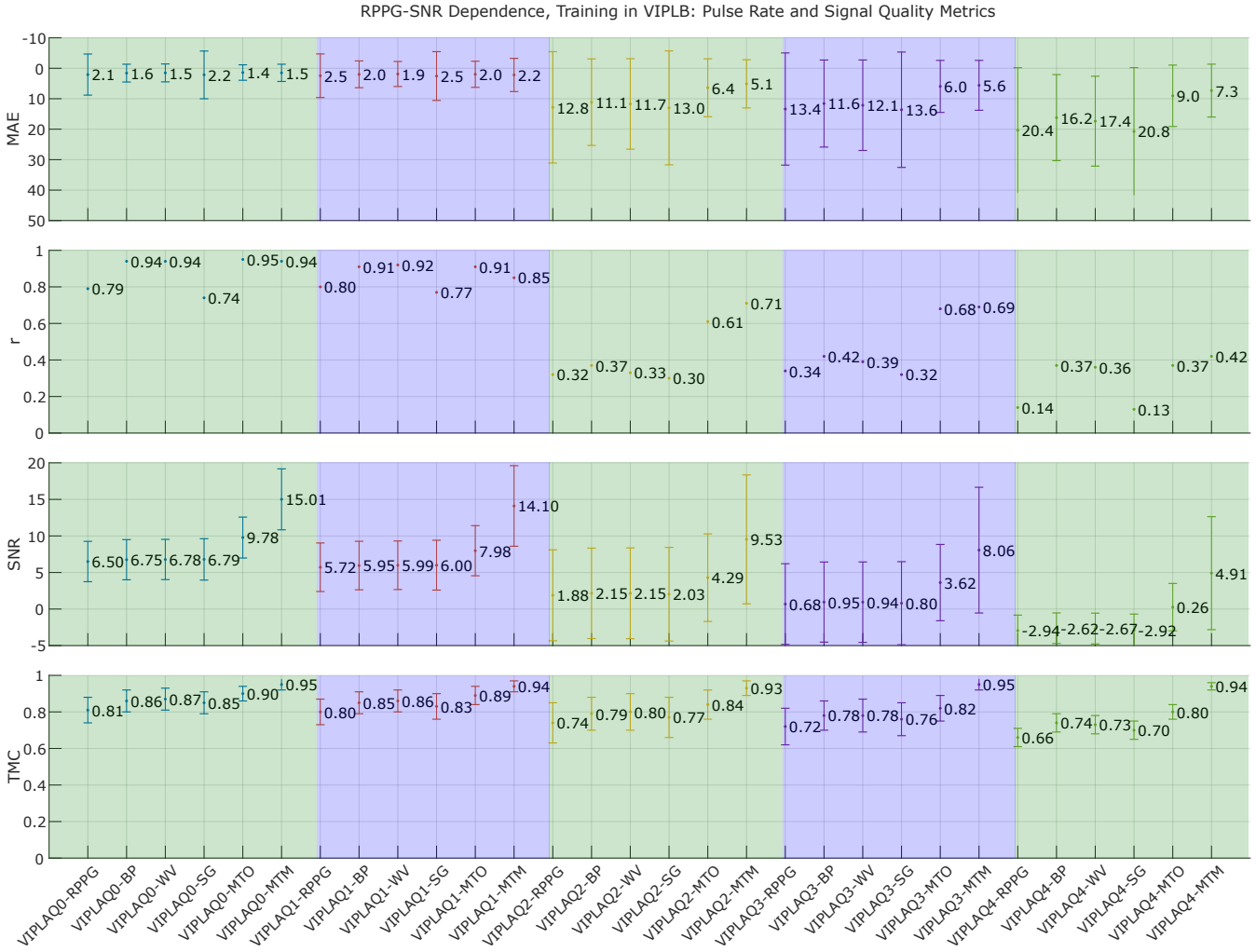

Figure 5.12: RPPG-SNR dependence during testing. Results of pulse rate and signal quality metrics: MAE, r, SNR and TMC in the *RPPG-SNR dependence* experiment. Training in VIPLB and testing in VIPLAQ0, VIPLAQ1, VIPLAQ2, VIPLAQ3, and VIPLAQ4.

5.5/ PULSE RATE VARIABILITY METRICS

So far, we have evaluated the LSTM-based method with two pulse rate metrics, demonstrating that LSTMDF is a viable option for filtering RPPG signals and measuring pulse rate more accurately. On the other hand, we use signal quality metrics to demonstrate that the filtering removes noise without affecting the pulse rate frequency. In this section, we want to verify that the improvement of the signal quality observed so far can be of interest for measuring cardiac variability. Indeed, the measurement of cardiac variability is based on the detection of peaks on the temporal RPPG signal and is therefore very sensitive to noise. Moreover, too strong filtering could improve the heart rate measurement and increase the metrics associated with the signal quality but suppress the subtle variations of the heart rate and eventually deteriorate the measurement of the cardiac variability. Therefore, we will measure two time-based pulse rate variability features, SDNN and RMSSD, and two frequency-based, LF and HF, on the RPPG and ground truth signals [48]. Then, we will compute the error between the results measured in RPPG and ground truth. Each feature is prefixed with the letter E followed by a colon to indicate that we are talking about the error (E:SDNN, E:RMSSD, E:LF, and E:HF).

In annexes section, we present the results of the pulse rate variability metrics of all the experiments developed in this chapter: *intra-dataset*, *cross-dataset*, *Amount of training data*, and *RPPG-SNR dependence*. Nevertheless, this section aims to evaluate the LSTMDF MTM (which proved to be the best choice between MTO and MTM) filter through pulse rate variability metrics in the most relevant protocols, intra-dataset, and cross-dataset.

In table 5.6 we show the results of the pulse rate variability features for RPPG from the PVM method, and after using MTM. In the MMSE-HR database, the error in the RPPG signal is the lowest compared to the other two databases, followed by VIPL-HR and COHFACE. As expected, these values before applying the filter show that the signal quality also affects the pulse rate variability features. By using the MTM filter, the signal quality improvement decreases the error in all four features for all three databases, demonstrating that in an *intra-dataset* protocol, the MTM filter can be used to improve the pulse rate measurement and pulse rate variability. Even in Figure A.3, where we compare all classical filters and MTO and MTM methods, MTM overperforms all filtering methods for the four metrics in the three databases.

Table 5.6: Pulse rate variability metrics in the *intra-dataset* protocol

Dataset	MMSE-HR		VIPL-HR		COHFACE	
Method	PVM	PVM+MTM	PVM	PVM+MTM	PVM	PVM+MTM
E:SDNN	0.04	0.01	0.09	0.03	0.15	0.05
E:RMSSD$*10^{-3}$	5.80	1.90	11.0	2.1	15.0	2.7
E:LF	0.17	0.12	0.44	0.17	2.08	0.63
E:HF	0.31	0.10	0.82	0.11	3.84	0.37

Finally, in Table 5.7, we can evaluate the performance of the MTM filter to improve the pulse rate variability features in the *cross-dataset* scenario.

Table 5.7: Pulse rate variability metrics in the *cross-dataset* protocol

Test set	**MMSE-HR**			**VIPL-HR**			**COHFACE**		
Training set	PVM$^{(*)}$	VIPL-HR	COHFACE	PVM$^{(*)}$	MMSE-HR	COHFACE	PVM$^{(*)}$	MMSE-HR	VIPL-HR
E:SDNN	0.04	0.02	0.05	0.09	0.07	0.05	0.15	0.14	0.06
E:RMSSD$*10^{-3}$	6.1	2.2	3.4	10.7	5.6	2.8	15.3	8.5	4.6
E:LF	0.22	0.14	0.35	0.44	0.63	0.29	1.99	3.39	1.04
E:HF	0.36	0.13	0.34	0.82	0.42	0.21	3.85	1.88	0.6

PVM$^{(*)}$: Reference value measured in the PVM-RPPG signals of the test set

For the MMSE-HR database, if we train on VIPL-HR, we manage to improve all four metrics, while training on COHFACE, E:SDNN, and E:LF gets a little worse. When we train in MMSE-HR and test in VIPL-HR, all metrics except E:LF are improved, whereas when we train in COHFACE, all four metrics are improved. If we train in MMSE-HR and test in COHFACE, E:LF worsens, whereas when we train in VIPL-HR, all four metrics are greatly improved.

These results shows the same behavior as the pulse rate and signal quality metrics of the previous experiments. That is, there is an improvement in the performance as the signal quality of the training set becomes more similar to the test set. Using techniques based on deep learning to filter periodic signals, there is a risk that the model smooths the signal too much and generates a sinusoidal-like signal. From the signal quality metrics and pulse rate variability metrics we realize that the filtering preserves the subtle variations of the RPPG signal. Removing the false peaks from the signal helps to have a more accurate value of the time between each heartbeat and, therefore, obtains better performance in pulse rate variability. Nevertheless, from the values found, it is difficult to conclude whether the improvement is sufficient for stress estimation or atrial fibrillation applications. We will discuss this insight in the conclusions section of this thesis.

5.6/ CONCLUSIONS

In this chapter, we analyzed the performance of the proposed LSTMDF network for RPPG filtering using multiple protocols. Two sequence-to-sequence filter approaches were evaluated: many-to-one and many-to-many. We used three public databases in different experiments from which we can draw the following conclusions: the experiment *amount of training data* showed that a relatively low number of signals is needed to train the LSTM network efficiently. It was shown that even a dataset of approximately 90 signals totaling 45 minutes could be sufficient as a training set. In an intra-dataset scenario, both MTO and MTM overperform the conventional filters, but the MTM approach gives the best results.

We compared the six state-of-the-art RPPG estimation methods: PVM, POS, PbV, G-R, Chrom, and Green, after using the LSTM-based filter. The results suggest stable performance in pulse rate, signal quality, and pulse rate variability metrics. Using the LSTMDF filter on RPPG signals acquired by the PbV method gave the best performance.

The *cross-dataset* and *RPPG-SNR dependence* protocols are perhaps the most interesting experiments as they reflect a dependence of the LSTMDF filter on the RPPG-SNR average of the training and testing sets. Our experiments showed that it is recommended that the RPPG-SNR average of the training set has to be as close as possible to those of the test set; if so, we can expect the LSTMDF to overperform classical filters even in a cross-dataset scenario.

The developed protocols let us appreciate how an LSTM-based filter is a better alternative than the classical filters, which improves the pulse rate measurement, and the signal quality. In the next chapter, we will propose a 3DCNN network to measure RPPG signals from video.

6

RTRPPG: Real-Time Remote Photoplethysmography

6.1/ Overview

Recently, deep-learning-based approaches such as 3D convolutional networks have outperformed traditional RPPG hand-crafted methods. However, despite their robust modeling ability, it is well known that large 3DCNN models have high computational cost and may be unsuitable for real-time applications. In this chapter, we propose a study of the 3DCNN architecture, finding the best compromise between pulse rate measurement precision and inference time. The fast inference is obtained decreasing the input size while the precision performance is obtained introducing a new time and frequency-based loss function by adding the signal-to-noise-ratio component to the regular Pearson's correlation loss function. In addition, changing the input color space from RGB to YUV slightly improved pulse rate measurement precision. Using the VIPL-HR database, we retained the pulse rate mean absolute error at 3.99 bpm which is comparable to 3.87 bpm of the state-of-the-art, while the GPU and CPU inference process improved around 96% from 51.77 ms to 2.32 ms in GPU and from 816.47 ms to 28.65 ms in CPU. The resulting network is called Real-Time RPPG (RTRPPG). Finally, we implement a synthetic RPPG video data augmentation strategy to improve the pulse rate and signal quality performance of RTRPPG when working with challenging databases.

6.2/ Introduction

When a neural network baseline is created, it is common to optimize its different modules through an ablation study [81], where different approaches are tested for each module, thus experimentally deciding the best configuration. This practice is also used in deep learning models for RPPG measurement. For example, In [109], an in-depth analysis was made on the behavior of a 2DCNN based on DeepPhys [54]. The authors studied in one of their experiments the importance of the spatial context, taking the input frames to the network of 64x64 pixels to down-sample to 1x1, 4x4, 8x8, 20x20, 30x30, 40x40, 50x50, and then up-sample back to 64x64. In this way, using the same network architecture, they analyzed different context information. The results suggest that different resolutions cause minor fluctuations in network performance. However, whether this conclusion is valid in a 3D convolutional network is unclear. On the other hand, even if the spatial

context is different, the network input was always 64x64, avoiding to improve the inference time. Similarly, in [119], the authors use the PhysNet 3DCNN network [90] as a baseline to propose a series of experiments that evaluate the importance of the frame rate. They propose 20 different frame rate configurations between 4.5 fps and 90 fps. The time and frequency domains evaluation suggest that decreasing the frame rate may lead to better network performance due to the increased length of time that a spatial-temporal kernel covers.

The choice of color space is also important in the RPPG/HR acquisition task [30], because video-frames contain the information of blood volume changes in their three channels. Hence some authors have proposed to use channels other than Red, Green, and Blue. In the literature, we can find RPPG/HR methods where authors use color channels such as Lab [17], Luv [17] or YCbCr [82]. Interestingly, in deep-learning-based RPPG/HR measurement the YUV color space has shown promising results [82, 102, 75]. Particularly, in [82], authors make an empirical study of which color space (RGB, HSV, YCrCb, and YUV) is more useful to create a spatial-temporal map to acquire HR, concluding that YUV is the best choice to train a neural network based on convolutional and recurrent layers. However, it is unclear whether this same conclusion applies to a 3D convolutional network.

The loss function is another essential part of any deep-learning-based application. In the particular case of acquiring physiological signals from a video, the loss function to choose will be related to the signal to be measured. For example, when measuring HR directly (single value), the Euclidean distance loss function is often used [15, 109]. On the other hand, to acquire RPPG signals (vector), a regression approach is normally used. At first, the Mean Squared Error loss was used in [54], then some authors used Negative Pearson's Correlation to have a more accurate comparison between the pulse peak locations with their respective ground truth signals [90, 91, 119]. However, this approach does not evaluate explicitly the frequency components of the signals, which is important in pulse rate measurement. In [65], the first frequency-based SNR loss function was proposed, showing that it is possible to extract the frequency information from the signals; however, a combination of temporal-frequency-based loss functions may provide even better results in the RPPG signal acquisition problem.

End-to-end methods based on 3DCNNs have demonstrated promising results in measuring RPPG signals, and pulse rate [90, 91, 108] (Figure 6.1). In this chapter, we build upon previously proposed architectures while focusing on optimizing their inference speed for real-time applications (potentially on low-end devices). Optimizing the inference time can be approached systematically through an ablation study, where various network components such as the size and color space of the input images and the loss function are evaluated. We expect to optimize network response time, signal quality, and pulse rate measurement precision. The main contributions of this chapter, obtained using an ablation study where we tuned network size, loss function, and color space, are :

- A new 3DCNN called Real-Time RPPG (RTRPPG). It achieves results comparable to those found in the literature, acquiring RPPG signals from real-time videos. The inference time is around 2.32 ms on GPU and 28.65 ms on CPU.
- A new temporal-frequency-based loss function that allows the 3DCNN to learn the essential features of the RPPG signal acquisition task. Our loss outperforms the baseline temporal-based loss function.

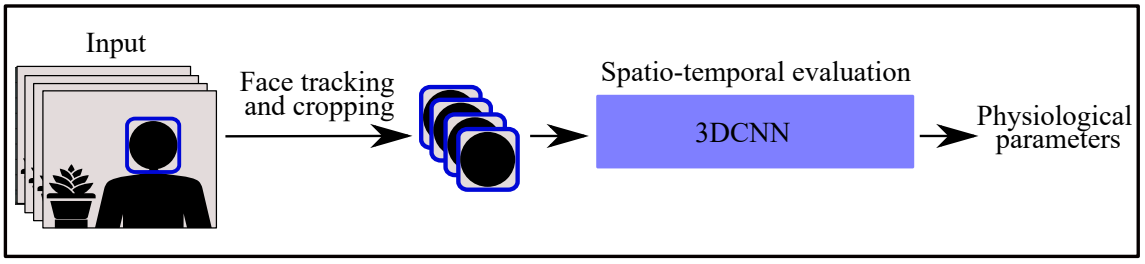

Figure 6.1: End-to-end 3DCNN RPPG measurement pipeline.

The remainder of this chapter is organized as follows: First, we will propose a temporal-frequency-based loss function, then a Spatio-temporal network for measuring RPPG signals. Specifically, we will start from a 3DCNN-based baseline. Then, we will optimize the baseline network by changing the spatial size of the input videos, followed by a change in the loss function, and finally, a change in the input color channel. Then, we evaluate data augmentation by means of synthetic RPPG videos to improve pulse rate, signal quality, and pulse rate variability metrics. Finally, at the end of this chapter we will summarize the work done along this chapter.

6.3/ FREQUENCY-BASED LOSS FUNCTION

Pearson's correlation coefficient (ρ) can measure the linear relationship between the temporal characteristics of RPPG and the blood volume pulse ground truth (PPG signal), ignoring the frequency-based characteristics. On the other hand, the frequency domain contains the components related to heart rate and signal quality; therefore, the Signal-to-Noise-Ratio can enhance the frequency-based components. Consequently, we use ρ and SNR to optimize the most important characteristics of the RPPG signals. In Equation 6.1, we propose a new temporal-frequency-based loss function Negative Pearson's correlation and Signal-to-Noise Ratio (NPSNR) that unites both metrics:

$$\text{NPSNR} = 1 - (\rho + \lambda \frac{\text{SNR}}{\kappa}), \tag{6.1}$$

where κ is an experimentally found constant that limits the SNR values within the range of ρ. To be specific, κ is the maximum possible SNR value in the time window L. κ was measured on a L-length sine signal with its clean FFT (Figure 6.2). λ is a constant that balances the frequency component, ρ is the Pearson's correlation between the RPPG signal and its ground truth, $\mathbf{y}$ and $\mathbf{g}$, respectively (6.2):

$$\rho = \frac{\sum_{t=1}^{T} (\mathrm{y} - \overline{y})(\mathrm{g} - \overline{g})}{\sqrt{\sum_{t=1}^{T} (\mathrm{y} - \overline{y})^2} \sqrt{\sum_{t=1}^{T} (\mathrm{g} - \overline{g})^2}}. \tag{6.2}$$

SNR is the ratio of the power of the main pulsatile component and the power of background noise, defined in equation 4.5, section 4.3.2.1. Figure 6.2 depicts two examples of NPSNR measurements. The upper part shows the results on a noise signal, and the lower part, on a sinusoidal signal with the same frequency as the ground truth.

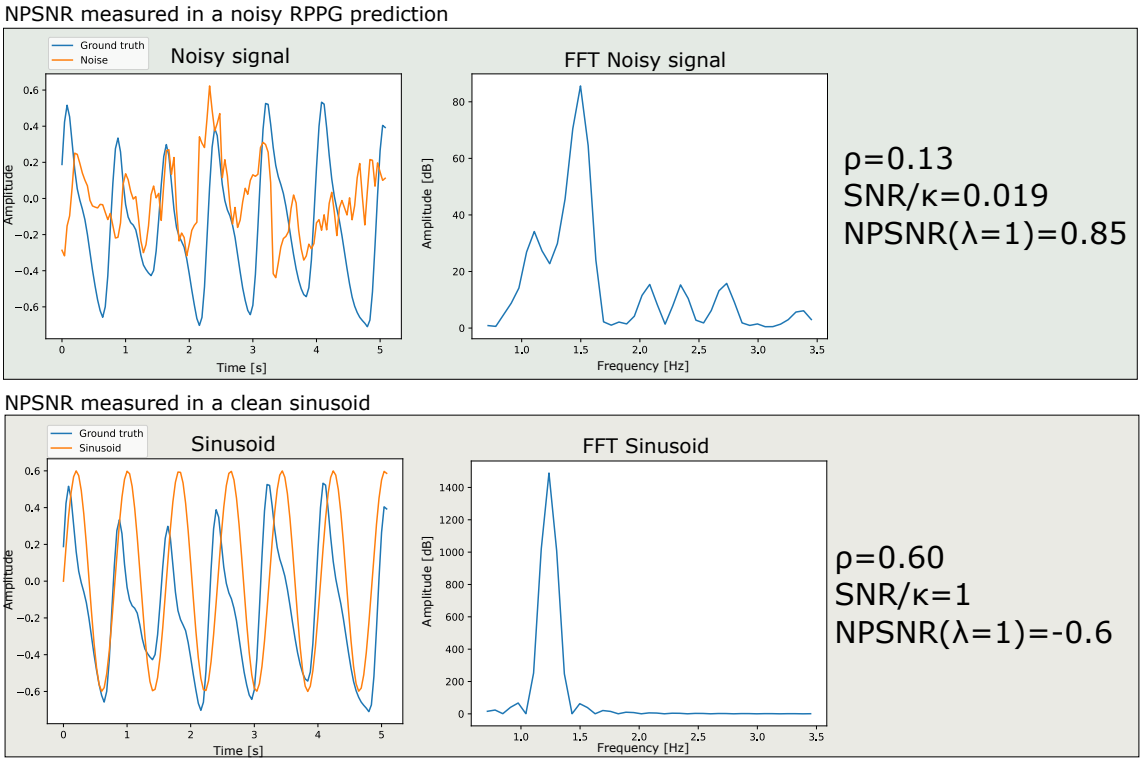

Figure 6.2: NPSNR measured in two signals. A noisy signal at the top, and a clean sinusoid at the bottom.

We will now present a series of experiments to find a 3DCNN fast and accurate in RPPG measurement. During the training of the networks used in the following experiments, we used the hardware available at Google Colaboratory [1] and a personal computer. However, when comparing the inference time and time spent during training (GPU only) between models, we used only the personal computer with the following technical specifications: Intel Xeon 2.4 GHz CPU, 16 GB RAM, and an NVIDIA GeForce RTX 2070 GPU. All networks were implemented with *PyTorch* libraries version 1.9.0. During the training process we adopted a subject-independent 5-fold cross validation evaluation protocol.

6.4/ SPATIO-TEMPORAL NETWORK

In this section we use an encoder-decoder neural network based on 3DCNNs as a baseline to measure RPPG. We propose an ablation study to improve inference speed while maintaining accuracy. We tune image size and color space, and use our new temporal-frequency-based loss function. All training history loss, and correlation plots of the following experiments are presented in the Annexes, section A.5.

The input is a T-frame video with frames $[f_1, f_2, ..., f_t, ..., f_T]$ with $t \in [1, T]$. The video is of any three-dimensional color space. For video face detection, we used the neural network implementation based on BlazeFace [71] called MediaPipe[2] (denoted as F_d), in charge of extracting the face of the subjects found within each frame. The choice of MediaPipe

[1] https://colab.research.google.com
[2] https://github.com/google/mediapipe

has been justified by an experimental study showing better stability than other evaluated methods. Then, we use a resizing procedure denoted as Ω with the *OpenCV* INTER-AREA interpolation method denoted as *I*, thus, we have square images of dimensions bxb. The overall procedure is presented in 6.3:

$$[f'_1, f'_2, ..., f'_T] = \Omega(F_d([f_1, f_2, ..., f_T], \varphi), I), \tag{6.3}$$

where $[f'_1, f'_2, ..., f'_T]$ are the bxb T-frame face images after being resized, φ are the parameters of F_d.

To use the video frame's spatial and temporal characteristics, we use a 3DCNN that takes the resized video frame $([f'_1, f'_2, ..., f'_T])$ as input, generating the RPPG output $\mathbf{y} = [y_1, y_2, ..., y_T]$. The RPPG estimation by the 3DCNN is presented in 6.4:

$$[y_1, y_2, ..., y_T] = \text{3DCNN}([f'_1, f'_2, ..., f'_T]); \omega), \tag{6.4}$$

where ω represents the parameters of 3DCNN. Below we present the procedure performed to establish the 3DCNN baseline.

6.4.1/ 3DCNN BASELINE

To establish our 3DCNN baseline, we reviewed the literature to choose the most widely used and flexible network to be modified, PhysNet [90]. The following study is a comparison between the PhysNet network and our baseline network.

6.4.1.1/ PHYSNET

PhysNet is the result of a comparison between different configurations of spatio-temporal networks. This neural network uses 3D convolutions and pooling operations to generate an encoder and a decoder. Its architecture is presented in Figure 6.3.

The 3DCNN PhysNet input consists of an RGB video frame with dimensions: width (W) 128 pixels, height (H) 128 pixels, and temporal (T) 128 frames. After a series of convolutions, activation functions, and pooling operations, the input information is encoded in a latent space; this information is decoded by a series of transposed convolutions, activation functions, and pooling operations. The output is a one-dimensional signal of T=128, one value for each time instant, thus generating the RPPG signal of the video frame.

To observe the performance of PhysNet in a series of challenging scenarios, we performed a subject-independent 5-fold cross-validation on the VIPL-HR database. Following the information given by the authors of PhysNet, we trained the neural network with the Negative Pearson's Correlation loss function (NP) and the Adam optimizer with learning rate of 0.0001 for 15 epochs.

In Figure 6.4 we present on the left side, GPU and CPU inference time, pulse rate and signal quality metrics with the correlation plot, and on the right side, three different subject predictions. The correlation plot is between the measured and real PR values. The RPPG signals of the three plotted subjects were chosen as follows: the first an easy scenario where the acquired signal is almost perfect, the second a moderately difficult video, and

the third, a more challenging video. In this way, we can visually evaluate the RPPG signals generated as we test new 3DCNNs configurations. The level of difficulty in the databases is due to changes in light, different skin colors, moving subjects and other external sources of noise.

The results of the metrics provided by PhysNet are good even in a database as complex as VIPL-HR. However, on our hardware, the inference time for one particular time window by PhysNet takes 51.77 ms on GPU and 816.47 ms on CPU. Therefore, PhysNet may be unsuitable in a real-time context. Real-time capability, in our context, typically refers to when a model runs faster than a webcam at 30 fps (33.3 ms). Therefore, in the following sections, we will start from the encoder-decoder configuration proposed by PhysNet. We will propose a lighter network, from which we will start making the necessary changes to obtain accurate RPPG measurement and real-time use.

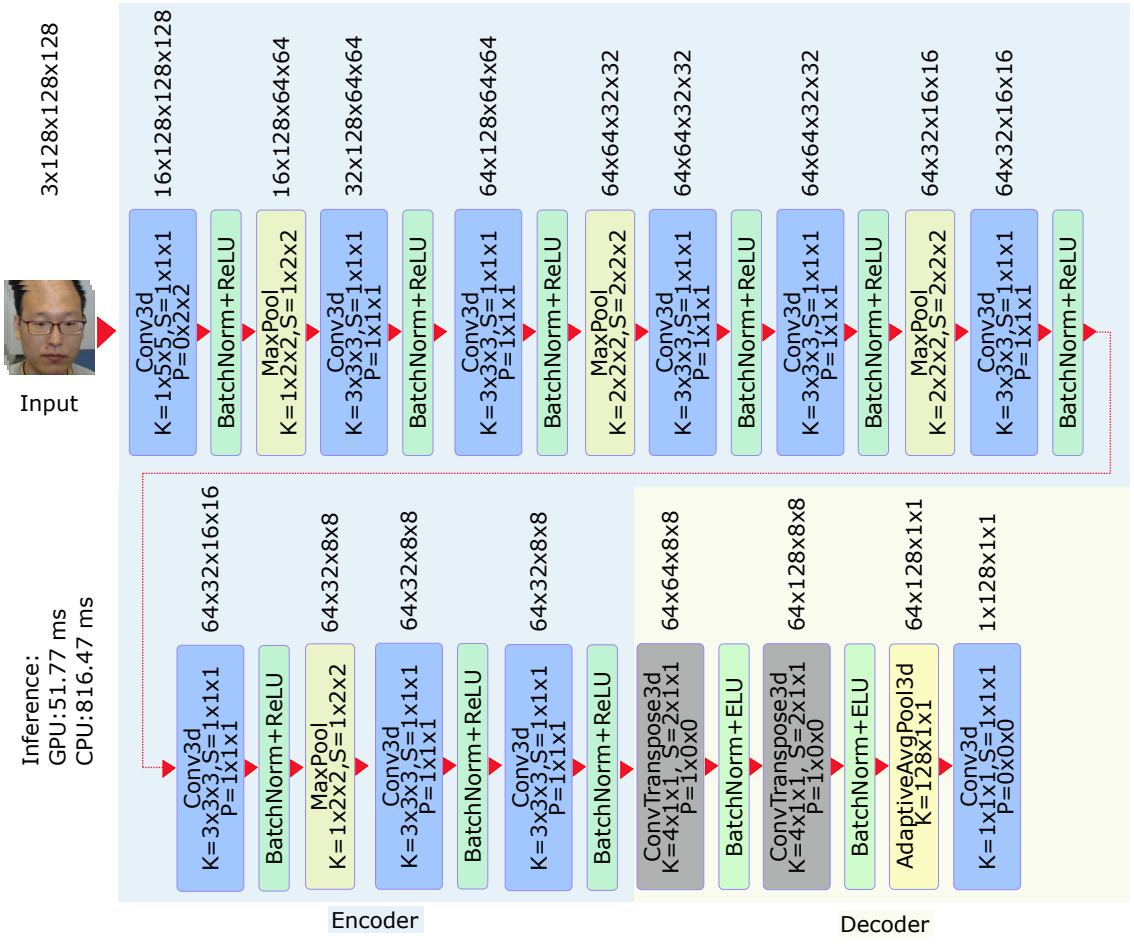

Figure 6.3: PhysNet Architecture. Adapted from [90]

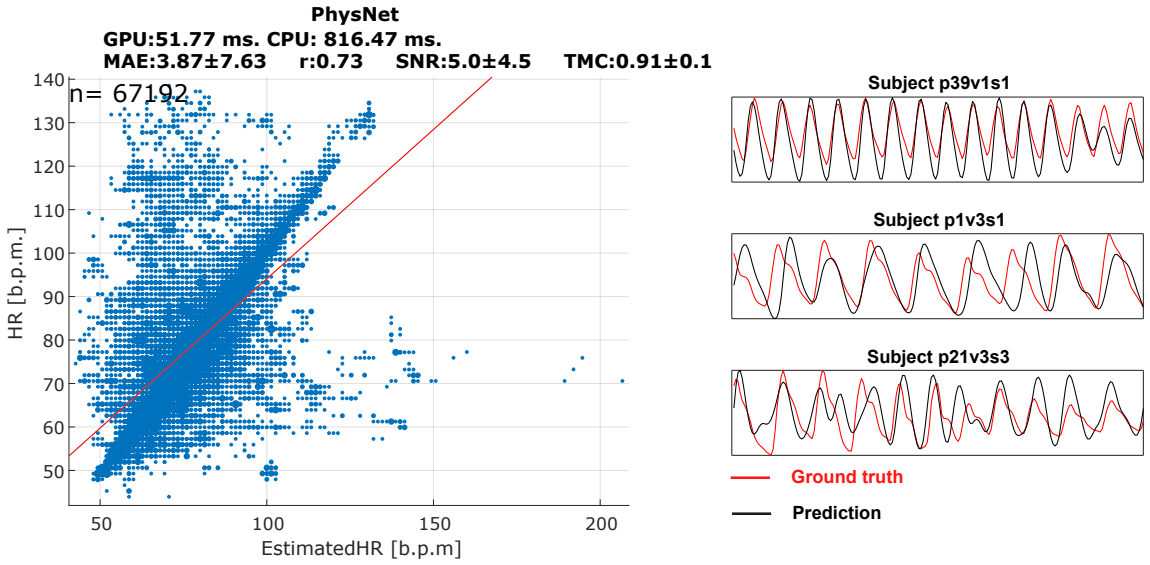

Figure 6.4: PhysNet results. On the left side, pulse rate and signal quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

6.4.1.2/ 3DED

Inspired by the spatio-temporal network implemented in [90], we propose a 3DCNN-Encoder-Decoder denoted as 3DED as a baseline to find the RPPG signals associated with a video. This network is divided in two main parts. The first one is the encoder E where the input data is transformed in a latent-space with more significant spatio-temporal information. The second part, receiving the latent-space feature as an input, is the decoder D that generates the RPPG output. The 3DED architecture (Figure 6.5) is an encoder-decoder composed of convolutional layers, activation functions, and pooling operations.

3DED decreases the inference time from 51.77 ms (PhysNet) to 20.38 ms on GPU and from 816.47 ms (PhysNet) to 240.57 ms on CPU. That is an improvement of approximately 61% and 71% on GPU and CPU, respectively. However, when comparing the metrics and the correlation plot presented in Figure 6.6, we notice that the network performance when measuring RPPG decreases. The pulse rate metrics MAE and r increase from 3.87 bpm and 0.73 (PhysNet) to 6.32 bpm and 0.53, respectively. signal quality is also affected, decreasing SNR from 5 (PhysNet) to 2.7 dB and TMC from 0.91 (PhysNet) to 0.86. Although with 3DED, the inference speed is closer to being real-time, it is still necessary for the network to be faster.

Visually, on the right side of Figure 6.6, the RPPG signals of subject p39v1s8 do not seem to have a significant difference when using PhysNet or 3DED, only the amplitude changes, but the frequency and signal quality are maintained. In subject p1v3s1, compared to PhysNet, the resulting RPPG signal maintains the frequency but begins to lose the characteristic shape of a PPG signal. Finally, subject p21v3s3 depicts a signal similar to that generated with PhysNet.

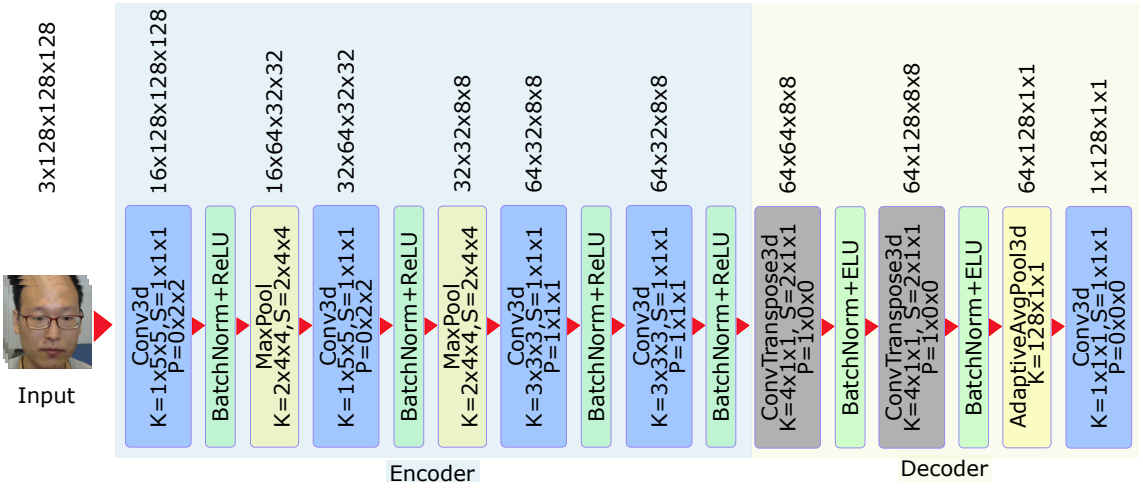

Figure 6.5: 3DED Architecture. Inference time of GPU: 20.38 ms, CPU: 240.57 ms.

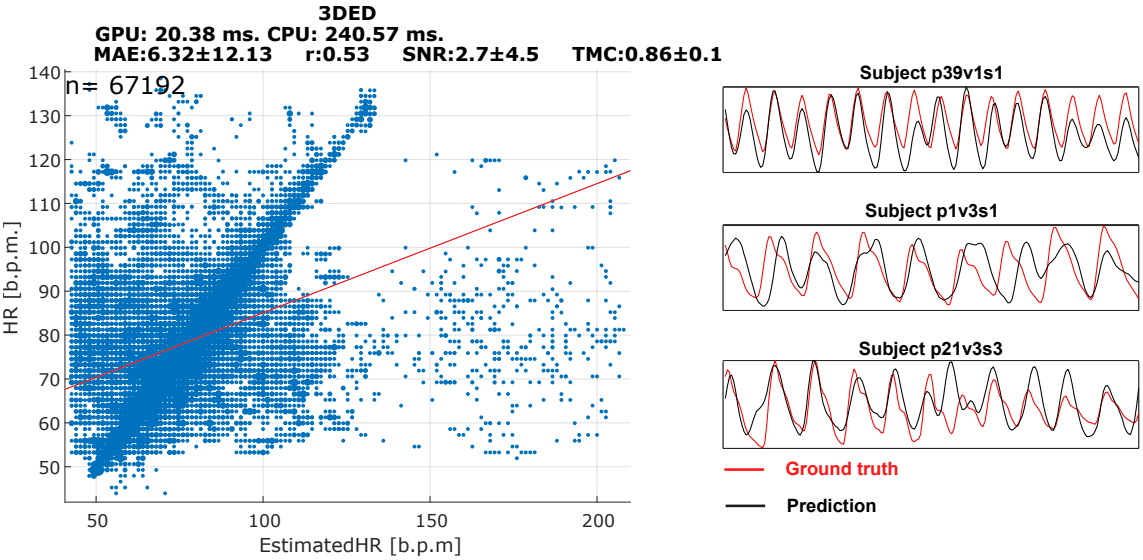

Figure 6.6: 3DED results. On the left side, pulse rate and signal quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

Therefore we are motivated to improve the inference speed and the MAE, r, SNR, and TMC metrics. To this end, we present below a strategy to improve the overall performance of the 3DED network.

6.4.2/ Network optimization

In this section we propose several experiments to acquire the best compromise between real-time, signal quality and pulse rate measurement precision.

Figure 6.7 depicts the experiments proposed in the ablation study. To cope with decreasing input sizes, we changed the pooling layers while applying the same convolutional operations. These changes only happen in the E encoder. We will refer to the network configurations as 3DED*input size-color channel-loss function*, *e.g.* 3DED8-RGB-NP is the 3DED network with input RGB, 8x8 pixels, and NP as loss function.

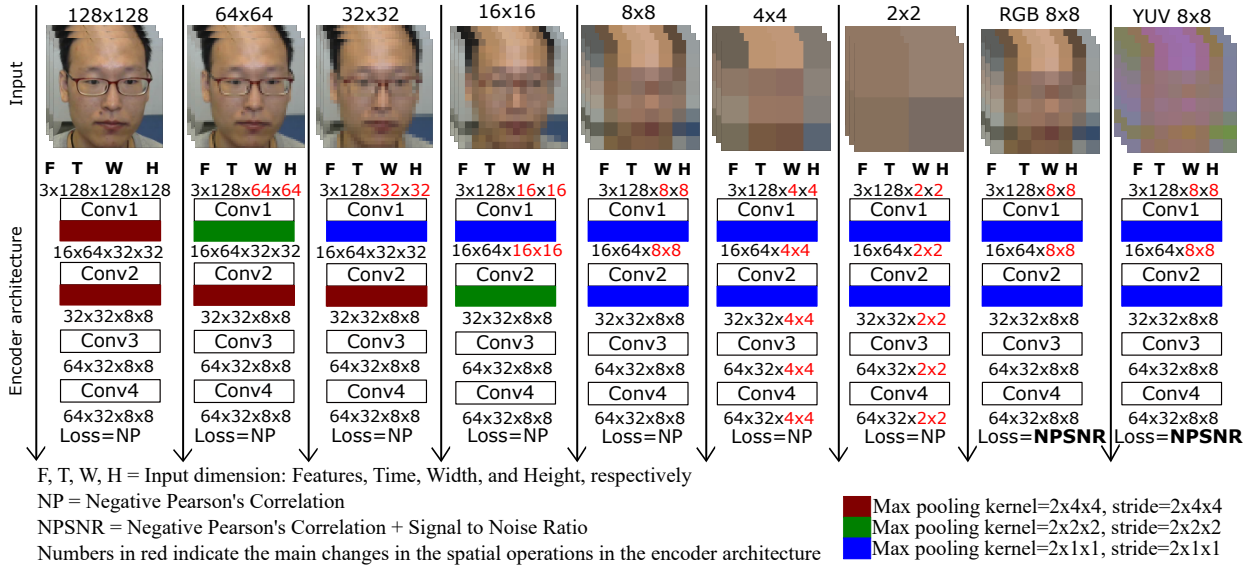

Figure 6.7: Network optimization. We use the 3DCNN baseline 3DED to gradually decrease the input resolution. We also change the loss function and the input color space.

6.4.2.1/ INPUT SIZE

The inference time depends on the hardware used, the network size, and the number of operations required to carry out the forward propagation. Decreasing the size of the input image decreases the number of operations during forward propagation and, therefore, also the inference time. In the first approach we gradually decrease the spatial dimensions of the input frames bxb into six different input sizes: 64x64 (3DED64-RGB-NP), 32x32 (3DED32-RGB-NP), 16x16 (3DED16-RGB-NP), 8x8 (3DED8-RGB-NP), 4x4 (3DED4-RGB-NP), and 2x2 (3DED2-RGB-NP).

Figure 6.8 depicts the results obtained in this experiment by decreasing the spatial size of the input video. In the first row the GPU and CPU inference times; we marked the real-time threshold of 33 ms with a blue horizontal line. In the second row are the pulse rate metrics MAE and r, and in the last row are the signal quality metrics SNR and TMC. The experiments names in bold indicate that they are suitable for real-time in GPU and CPU.

Using the lower resolution of W=2 and H=2, the inference speed is improved in GPU by 90%, going from 20.38 ms to 1.96 ms. In CPU, the improvement is 98%, from 240.57 ms to 3.96. However, when contrasting the pulse rate metrics, we see that the MAE value decreases by 23% in this configuration, going from 6.32 bpm to 7.83 bpm The r metric decreases by 15% from 0.53 to 0.45. Regarding the signal quality metrics, the results are similar, SNR decreases by 44% from 2.72 dB to 1.5 dB, and TMC decreases by 4.6% from 0.86 to 0.82. Interestingly, when using 3ED2-RGB-NP we get higher performance in inference speed but lower performance in pulse rate and signal quality metrics compared to the baseline network. Nevertheless, the metrics are not as bad as one might think when using such a small image. Still, one particular configuration is optimal among all the others.

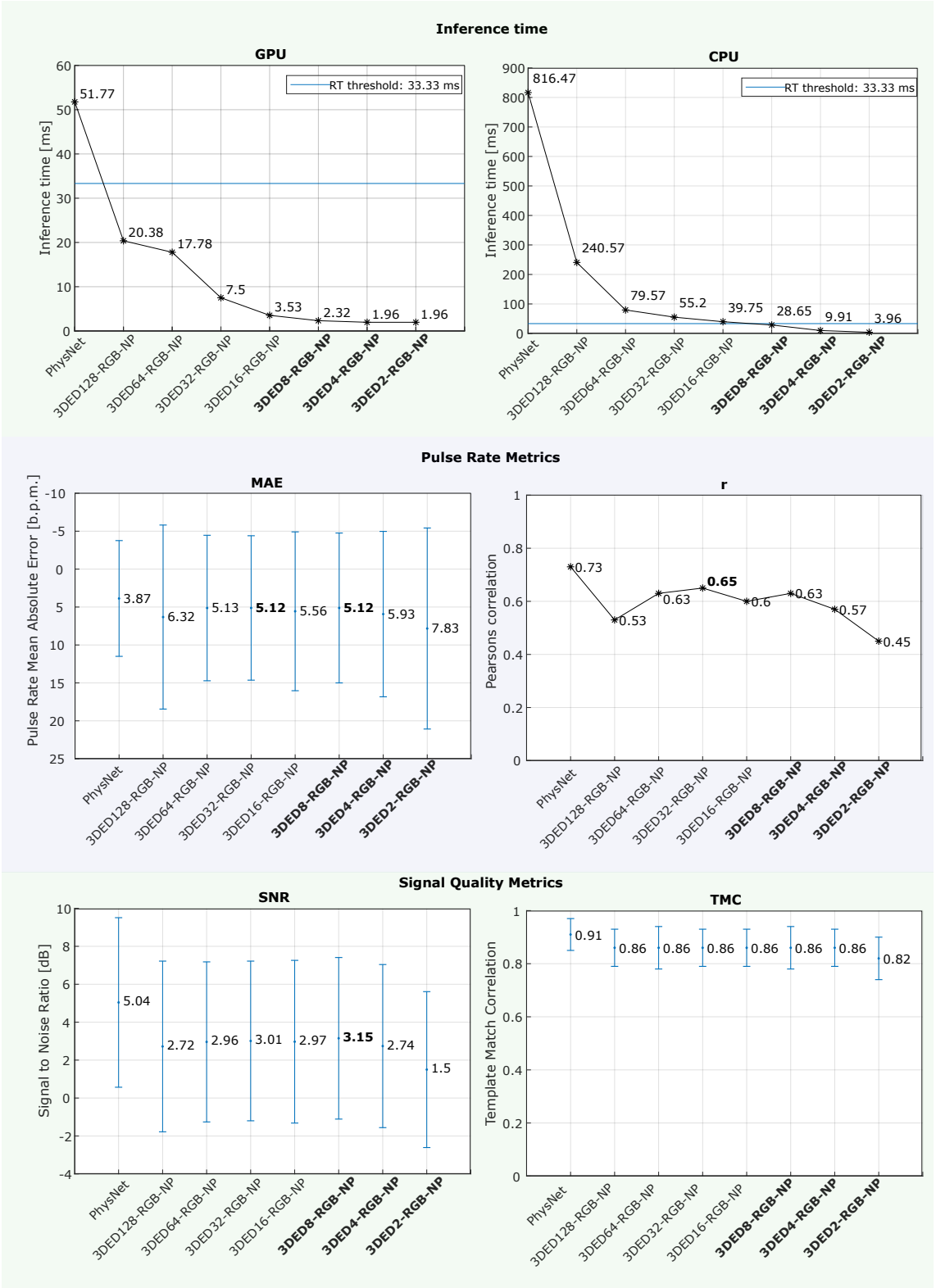

Figure 6.8: 3DED-based network results when decreasing the input size. In the first row the GPU and CPU inference time, in the second row the pulse rate metrics, and in the third one, the signal quality metrics. The experiments names in bold indicate that they are suitable for real-time in GPU and CPU.

All 3DED configurations can work in real-time using the GPU. However, our purpose is that it can also be used in a real-time context and eventually in low-end devices, so we only consider configurations that are also real-time on the CPU. Thus, we are left with three networks: 3DED8-RGB-NP (28.65 ms on CPU and 2.32 ms on GPU), 3DED4-RGB-NP (9.91 ms on CPU and 1.96 ms on GPU), 3DED2-RGB-NP (3.96 ms on CPU and 1.96 ms on GPU). Among the three resulting configurations, it is easy to see that 3DED8-RGB-NP is the optimal one, improving the inference time by 88% for GPU and CPU, the pulse rate metrics MAE by 19% from 6.32 bpm to 5.12 bpm, and r by 19% increasing from 0.53 to 0.63. The signal quality metric SNR is also improved by 16%, from 2.72 to 3.15, and TMC is stable at 0.86.

Figure 6.8 also shows a particular behavior in pulse rate and signal quality metrics compared with the baseline 3DED. Regarding the inference speed, it increases in GPU and CPU when decreasing the video input size. These results were as expected. However, we also expected a decrease in the metrics, assuming that by reducing the spatial information, the quality of the measured RPPG signal would also decrease. Interestingly, except for 3DED2-RGB-NP, the opposite happens. The other five configurations show better results in pulse rate and signal quality metrics than 3DED128-RGB-NP. Indeed, the input resizing process could be understood as a spatial filtering procedure. An extension of the encoder module, precisely, a weights-fixed-layer. The interpolation process reduces the input frame by encoding the pixels with a defined algorithm (*cv.INTER_AREA*[3] in our case). In the smoothed resized input, it is easier to extract the RPPG signal. When reaching a minimal dimension such as W=4 and H=4, or W=2 and H=2, the RPPG signal starts to disappear.

In the following experiment, we will evaluate the temporal-frequency-based loss function NPSNR in the best configuration found so far, 3DED8-RGB-NP.

6.4.2.2/ LOSS FUNCTION

In this experiment we use the 3DED8 architecture with color channel RGB. We propose to replace the temporal-based Negative Pearson's Correlation loss function with the temporal-frequency-based NPSNR loss function presented in subsection 6.3, using $\kappa = 10.9$ (Equation 6.1). To train the 3DED8-RGB-NPSNR network, we used 15 epochs and the Adam optimizer. Then, we performed hyperparameter tuning to find the learning rate and lambda values of 0.00044 and 1.32, respectively.

Table 6.1: 3DED loss comparison, NP vs NPSNR

Network	**MAE** [bpm]	**r**	**SNR** [dB]	**TMC**
3DED128-RGB-NP (baseline)	6.32±12.13	0.53	2.70±4.5	0.86±0.1
3DED8-RGB-NP	5.12±9.88	0.63	3.15±4.26	0.86±0.08
3DED8-RGB-NPSNR	4.37±8.97	0.68	4.24±4.47	0.88±0.07

Table 6.1 shows the results obtained by comparing the two loss functions. When comparing the 3DED8-RGB-NPSNR architecture with the baseline 3DED128-RGB-NP net-

[3]https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html

work, we note that the MAE pulse rate metrics are improved by 31% (12% more than 3DED8-RGB-NP), going from 6.32 bpm to 4.37 bpm. The r metric increases by 28% (9% more than 3DED8-RGB-NP). The signal quality metrics also increase, SNR by 28% (12% more than 3DED8-RGB-NP) and TMC by 2%. These results demonstrate that our temporal-frequency-based loss function is more relevant for acquiring RPPG signals. In the following experiment, we will evaluate different color channels.

6.4.2.3/ COLOR CHANNEL

Finally, we evaluate the performance by changing the RGB color space to Lab, Luv, YCbCr, and YUV. Figure 6.9 depicts the results obtained in this experiment by changing the color channel of the input video. In the first row the GPU and CPU inference times; we marked the real-time threshold of 33 ms with a blue horizontal line. In the second row are the pulse rate metrics MAE and r, and in the last row are the signal quality metrics SNR and TMC. The experiments names in bold indicate that they are suitable for real-time in GPU and CPU.

So far, we optimized the input size and chose a more appropriate loss function, thanks to which we improved the performance of the 3DED128-RGB-NP baseline network by MAE 30.8%, r 22%, SNR 35.8%, and TMC 2.2%. When testing different channel colors, the changes in pulse rate and signal quality performance are slight. On the other hand, looking at the results presented in Figure 6.9, YUV color channels narrowly overperform RGB, Lab, Luv, and YCbCr. Increasing MAE by 6% (3.99 bpm), r by 5.4% (0.73), SNR by 4.9% (4.59 dB), and TMC by 1.1% (0.89).

6.4.2.4/ RTRPPG VS PHYSNET

Thus, with this last series of experiments, we have a 3D convolutional network that acquires RPPG signals with performance comparable to state-of-the-art, fast enough to be used in real-time on GPU and CPU hardware. The best configuration 3DED8-YUV-NPSNR is re-named to Real-Time Remote Photoplethysmography: RTRPPG. On the left side of Figure 6.10, the pulse rate and signal quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

Comparing RTRPPG with PhysNet, one of the best 3DCNNs in the literature, we find that RTRPPG is comparable in RPPG measurement performance but better in inference speed. Our MAE of 3.99 bpm is close to the value given by PhysNet, 3.87 bpm (only 3% lower), an equivalent r of 0.73. The signal quality is also close, having an SNR of 4.59 dB, close to 5.04 dB (only 9% lower), and a TMC of 0.89, close to 0.91 (only 2% lower). Nevertheless, the most important result is the improvement in GPU inference speed by 96% from 51.77 ms to 2.32 ms, and in CPU by 96% from 816.47 ms to 28.65. In this way, we achieved our objective, to propose a 3DCNN network capable of measuring RPPG in an accurate and fast way. Next subsection compares the training time required by the networks implemented.

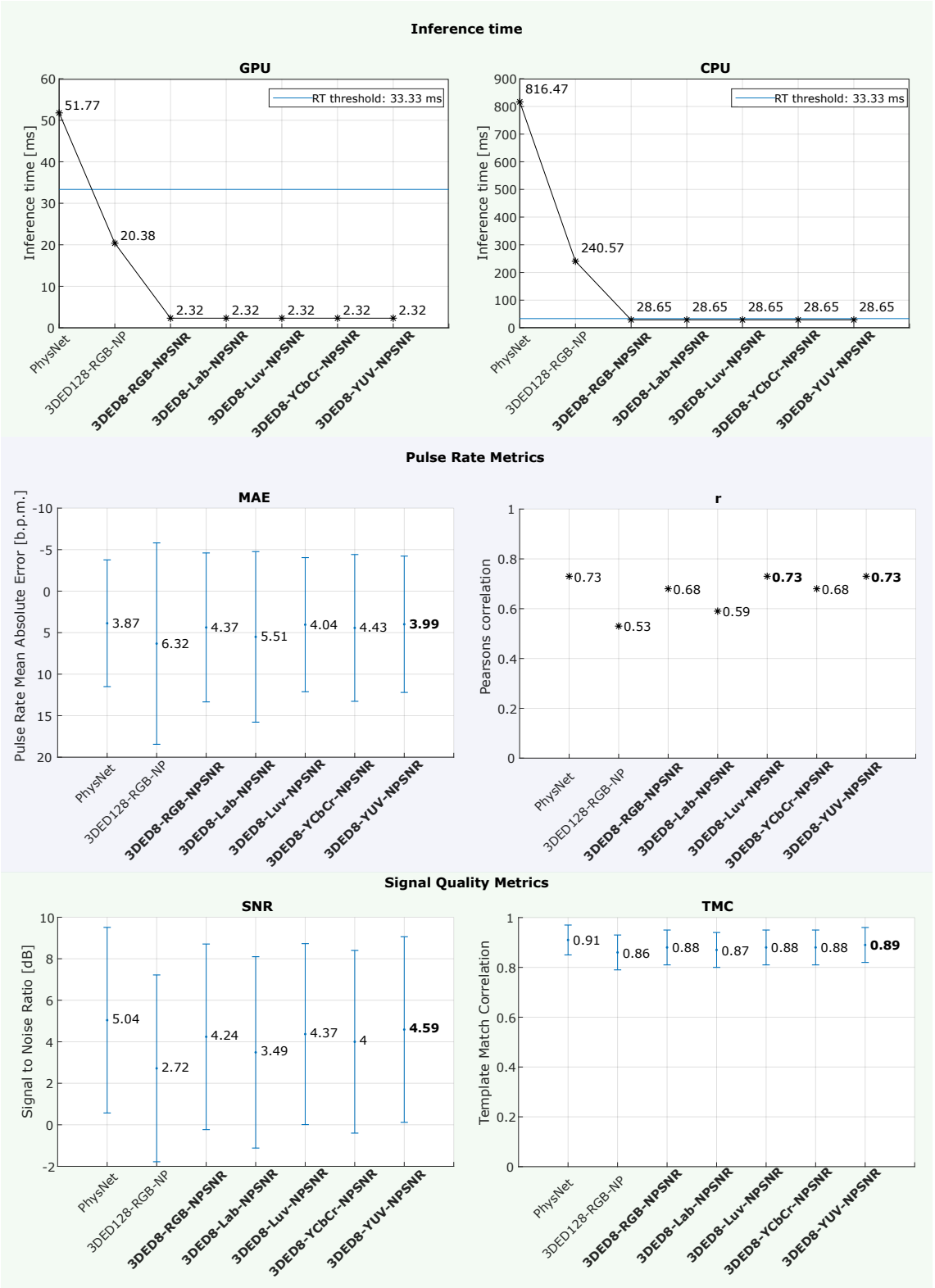

Figure 6.9: 3DED8-NPSNR network results when changing the color channel. In the first row the GPU and CPU inference time, in the second row the pulse rate metrics, and in the third one, the signal quality metrics. The experiments names in bold indicate that they are suitable for real-time in GPU and CPU.

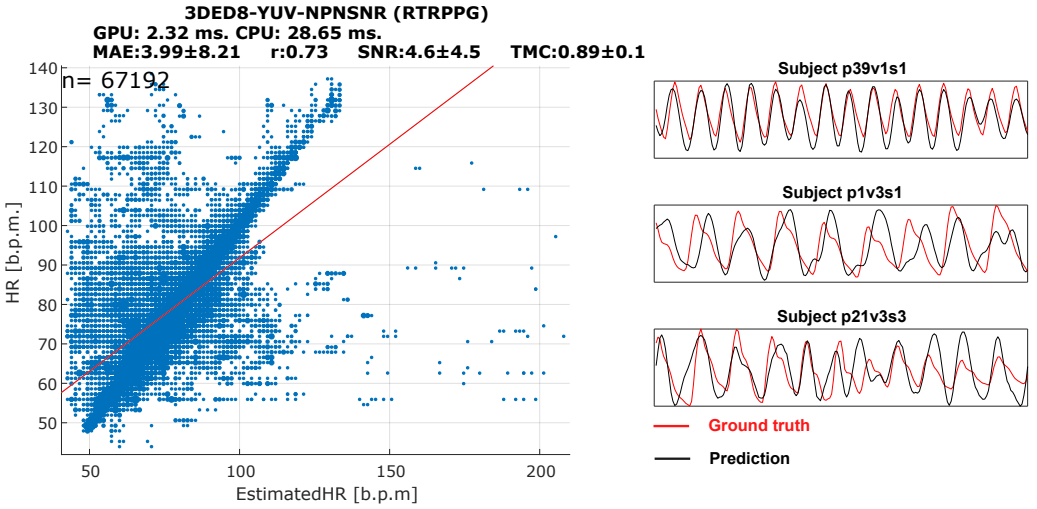

Figure 6.10: 3DED8-YUV-NPSNR Results. On the left side, pulse rate and signal quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

6.4.3/ TRAINING TIME

By decreasing the input size of the 3DED128 network, it was also necessary to modify the network size from 3DED128 to 3DED8 (3DED4 and 3DED2 share the same 3DED8 architecture). This process increases the inference speed and decreases the training time. This subsection compares the training time required on the evaluated networks. We settled to use our hardware (NVIDIA GeForce RTX 2070 GPU) instead of Google Colaboratory sessions because these sessions might not use the same hardware and because the inference time was also measured on our GPU. In this experiment, we trained the 100% VIPL-HR with a batch of 4 during 15 epochs (a batch of 8 was not possible for 128x128 videos). Figure 6.11 depicts the training time required in minutes and hours.

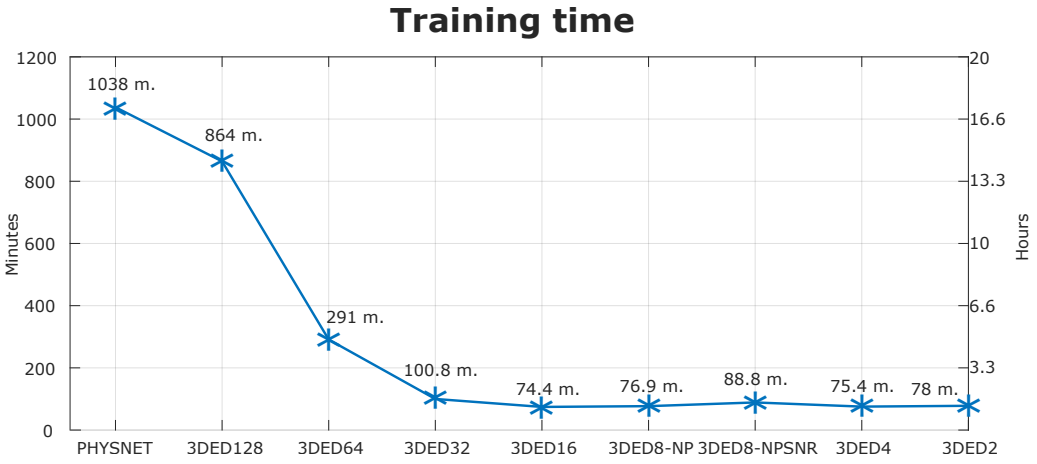

Figure 6.11: Networks training time comparison on NVIDIA GeForce RTX 2070 GPU.

PhysNet being the most robust network requires 1038 minutes (17.3 hours) of training, followed by our baseline 3DED128 network which requires 864 minutes (14.4 hours). As the complexity of the network decreases, the training time decreases considerably until

reaching the 3DED32 network, which takes 100.8 minutes (1.68 hours). The architecture used by RTRPPG (3DED8-NPSNR) decreases the training time by about 90%. In addition, 3DED8-NPSNR takes 15% more time (around 11.9 minutes more) than 3DED8-NP because the NPSNR loss function needs more computation time than NP. The following section will evaluate the RTRPPG network in the cross-dataset protocol.

6.5/ CROSS-DATASET

The RTRPPG network is comparable to PhysNet in an intra-dataset scenario using the VIPL-HR database. Now, we will evaluate the same pre-trained network in a cross-dataset strategy. Table 6.2 presents the results of testing RTRPPG on two small and easy databases (little movement and light changes), UBFC-rPPG and MMSE-HR. In addition, one moderate database (light changes and video compression) COHFACE, and one challenging database (light changes and abrupt movements) ECG-Fitness.

Table 6.2: RTRPPG vs PhysNet in cross-dataset scenario (Trained in VIPL-HR)

Network:Dataset	**MAE**[bpm]	**r**	**SNR**[dB]	**TMC**
PhysNet:UBFC-rPPG	0.67±1.0	1	9.28±2.9	0.97±0.01
RTRPPG:UBFC-rPPG	0.66±0.9	1	8.88±2.7	0.96±0.02
PhysNet:MMSE-HR	1.33±3.4	0.97	8.90±3.5	0.96±0.03
RTRPPG:MMSE-HR	2.18±5.9	0.90	7.18±4.2	0.93±0.05
PhysNet:COHFACE	12.94±13.4	0.09	0.51±5.1	0.84±0.04
RTRPPG:COHFACE	12.40±18.7	0.21	1.08±4.2	0.84±0.04
PhysNet:ECGFitness	21.37±21.9	0.31	-3.43±4.6	0.82±0.05
RTRPPG:ECGFitness	20.14±21.7	0.36	-3.15±4.5	0.80±0.07

The results in a cross-dataset strategy are similar to those in an intra-dataset. PhysNet, a robust but not real-time network, can obtain good results in easy and moderate databases. In a challenging database, as expected, the metrics start to decrease. What is relevant in these results is that the RTRPPG network allows us to obtain results very similar to PhysNet, and this network gives us a real-time prediction.

So far, we have evaluated RTRPPG on the VIPL-HR database in an intra-dataset and cross-dataset scenario. The pulse rate and signal quality performance are adequate because the VIPL-HR database has many videos and scenarios. Thus, the RTRPPG network can better generalize the RPPG measurement task. However, such a complete database is not always available, and downloading and processing the data can take a long time.

Let us consider three databases much smaller than VIPL-HR (1,144 minutes). An easy one, UBFC-rPPG. One more complicated, COHFACE, and a challenging one, ECG-Fitness, with 44, 166, and 187 minutes of duration, respectively. Table 6.3 shows the results obtained by training the RTRPPG network in an intra-dataset 5-folds cross-validation

scenario. The results show that RTRPPG performs well on the first two databases, UBFC-rPPG and COHFACE. On the other hand, due to the intense movements and large PR range of values, RTRPPG does not present good metrics on a challenging database like ECG-Fitness.

Table 6.3: RTRPPG intra-dataset performance

Dataset	**MAE** [bpm]	**r**	**SNR** [dB]	**TMC**
UBFC-rPPG	0.71±1.0	1	7.75±3.1	0.95±0.04
COHFACE	4.62±7.8	0.67	3.63±4.5	0.82±0.15
ECG-Fitness	27.08±19.5	0.08	-6.74±3.3	0.53±0.07

In order to improve the generalization of the RTRPPG network, including challenging databases like ECG-Fitness, next section proposes to use a data augmentation method by creating synthetic RPPG videos. Thus, we will be capable of measuring RPPG in complex scenarios with specific constrains using a few videos.

6.6/ Synthetic Data Augmentation

This section is composed of two parts. First, we will discuss related works on signal generation and synthetic RPPG video. In the second part, we will use a synthetic RPPG video generation as a fine-tuning strategy to pre-train the RTRPPG network.

6.6.1/ Synthetic data in RPPG

Procedures to create synthetic data are increasingly common in deep-learning applications [123]. Synthetic data is often used as pre-training [61, 105] or for lack of data [75]. In the RPPG measurement task, we also find works using this approach. In [61], authors use ResNet-18 [36] for feature learning in a regression model to measure HR. The training is carried out in three stages. ResNet-18 is pre-trained on the ImageNet dataset [28] in the first stage to obtain network parameter initialization. In the second stage, the network is pre-trained on large-scale synthetic spatial-temporal maps to avoid overfitting. Finally, in the last stage, from the cheek area of the face, real-life spatial-temporal maps are created to perform fine-tuning on the network. To the best of our knowledge, this is the first work to propose a model to create synthetic BVP signals for the HR measurement task. Specifically, the synthetic signals are created from a combination of sinusoidal signals and Gaussian noise. However, these synthetic signals do not consider amplitude and frequency variation related to cardiac variability and preventing the generation of realistic PPG signals.

Another transfer learning strategy applied in a ResNet-18 network was proposed in [105]. Specifically, The ResNet-18 network is pre-trained in ImageNet and then in videos from synthetic RPPG signals. Afterward, the network is fine-tuned in feature maps from real data. The synthetic RPPG is constructed from ECG and BVP signals. First, the ECG/BVP key points are found by peak detection. Second, the resulting points are interpolated

through a modified Akima cubic Hermite method [2] to get a curve. Finally, the curve is re-sampled at a desired sampling rate to get the synthetic RPPG. Nevertheless, the priority of this approach is to preserve the frequency of the signal, while no importance is given to maintaining the characteristic shape of the BVP signal.

In [75], an end-to-end pilot model for measuring pulse rate using 3DCNN is presented. The network acts as an extractor of spatial and temporal features from the input video frame. More specifically, the authors demonstrate the potential of training the network on synthetic videos. They start from a waveform construction using Fourier series. Then, they replicate the waveform to create the BVP signal. Additionally, they add a trend to the signal, and in order to convert signals to video, they repeat the vector in the width and height dimensions. Finally, they add random noise to the video. Thus, we can see that creating synthetic RPPG videos is a good strategy for increasing data in the RPPG measurement task. In the following sections, we will use a method proposed in *ImViA* laboratory to generate synthetic RPPG videos, initially presented in [116, 117].

6.6.2/ SYNTHETIC RPPG VIDEO GENERATION

In order to improve the RTRPPG pulse rate performance in challenging databases as ECG fitness, we use a transfer-learning strategy, training the RTRPPG in synthetic RPPG videos. The methodology for synthetic RPPG video generation is presented in Figure 6.12. An image of a static subject is taken and duplicated multiple times, generating a video clip; the frames share an attention mask with the skin position information. A synthetic PPG signal is added to the video using an attention mask to distribute the signal spatially. Finally, motion is transferred to the subject using a pre-trained first-order motion model[4].

The approach used to generate synthetic PPG is similar to the one proposed in [104], where sinusoidal signals were used to represent the main characteristics of a PPG signal. Additionally, we consider the effect of respiration and cardiac variability on the RPPG signal. This way, the technique used in this section models: 1) pulse rate related to the change of the volume in the blood, along with the frequency of the dicrotic notch 2) respiratory rate with its main component *baseline wander* (*bw*), which is a slow change over time with the main respiratory rhythm 3) Gaussian noise due to the measuring instruments 4) frequency modulation linked to cardiac variability and 5) amplitude modulation associated with respiration.

Similar to [104], we define the synthetic signal $s(t)$ in Equation 6.5:

$$s(t) = p(t) + d(t) + b(t) + n(t). \tag{6.5}$$

Where s is composed of four main components. The pulse rate and its dicrotic notch $p(t)$ and $d(t)$, respectively. The breathing rate $b(t)$ and the Gaussian noise $n(t)$. $p(t)$ and $d(t)$ are defined in Equations 6.6 and 6.7, respectively.

$$p(t) = A \sin(2\pi \int_0^t pr(t)dt + \Phi_{pr}) \tag{6.6}$$

[4] https://github.com/AliaksandrSiarohin/first-order-model

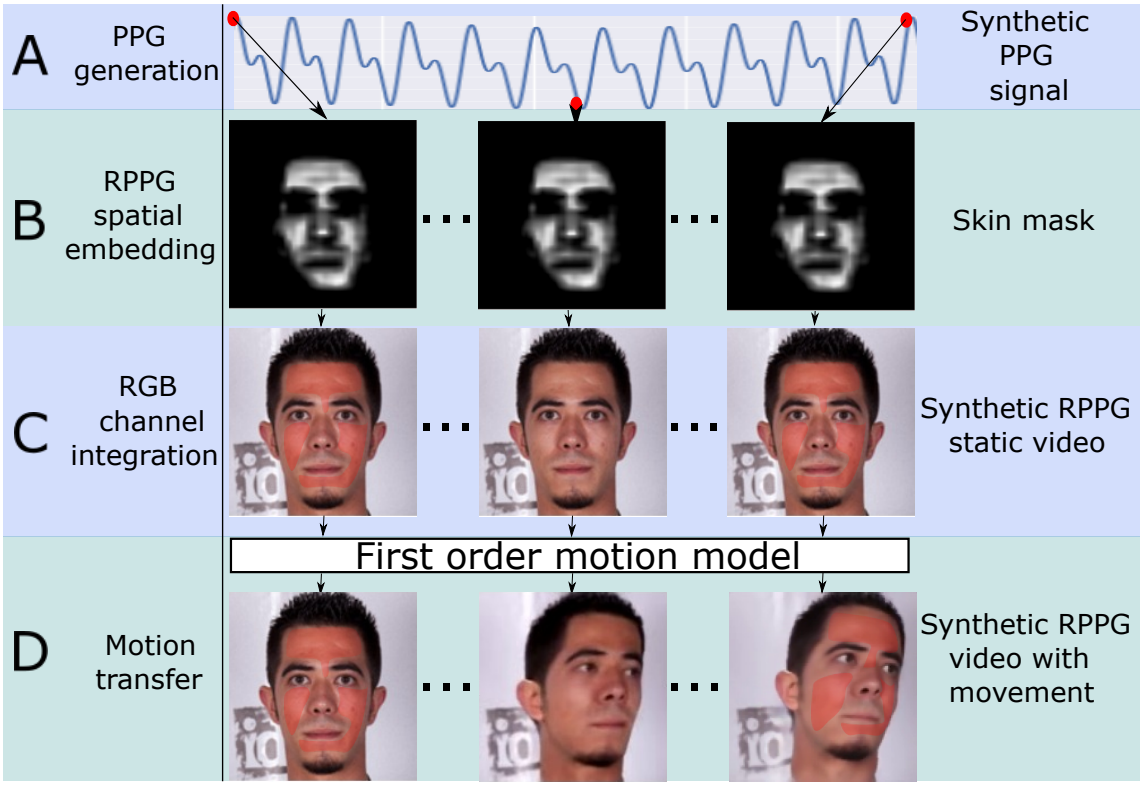

Figure 6.12: Synthetic videos generation. A) Synthetic PPG signal generation. B) Extraction of attention mask from the original image. C) PPG signal integration into static video via RGB channels. D) Adding motion to static video using first-order motion model. (A red color layer was purposely added to some parts of the skin to refer to the PPG value).

$$d(t) = A_2 \sin(4\pi \int_0^t pr(t)dt + \Phi_{pr}) \tag{6.7}$$

where A and A_2 are the signal and its dicrotic notch amplitudes, respectively. pr is the instantaneous pulse rate, and Φ_{pr} is the pulse rate phase. Similarly, The breathing *baseline wander* is given by Equation 6.8:

$$b(t) = B \sin(2\pi \int_0^t br(t)dt + \Phi_{br}) \tag{6.8}$$

with B the *baseline wander* amplitude, and Φ_{br} the breathing rate phase. Thus, we can generate simple synthetic BVP signals like the one presented in Figure 6.13.

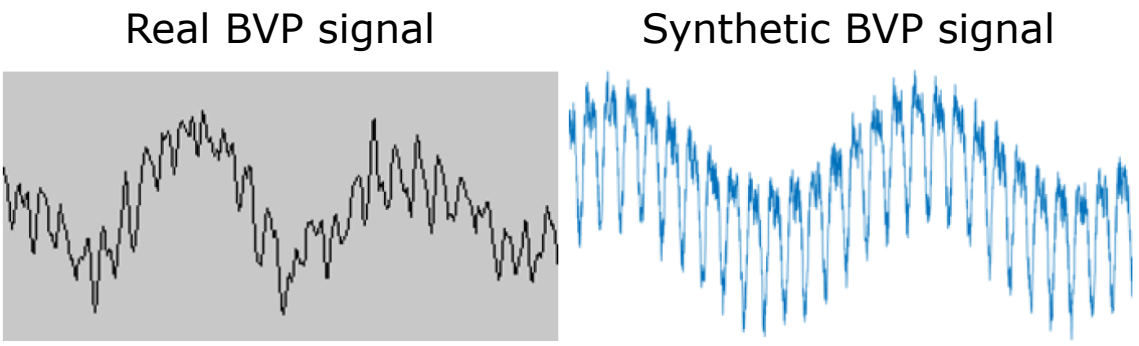

Figure 6.13: Real vs synthetic BVP signals. Adapted from [61]

The current synthetic BVP does not model the cardiac variability behaviour and it can be improved adding amplitude and frequency variations. Therefore, we start the improvement creating a new *baseline wander* $\hat{b}(t) = C_1 b(t)$. Then, we create the amplitude modulation signal $am(t) = C_2 b(t)$, and the frequency modulation by computing the new instantaneous pulse rate as: $\hat{pr} = pr(x) + C_3 b(t)$. C_1, C_2, and C_3 are constants. Finally, we create a new pulse rate signal $\hat{p}(t)$ with its dicrotic notch $\hat{d}(t)$ by adding $am(t)$ in Equations 6.6 and 6.7. Thus, $\hat{p}(t)$ and $\hat{d}(t)$ are defined in Equations 6.9 and 6.10, respectively:

$$p(t) = (A + am(t)) \sin(2\pi \int_0^t \hat{pr}(t)dt + \Phi_{pr}) \tag{6.9}$$

$$d(t) = (A_2 + am(t)) \sin(4\pi \int_0^t \hat{pr}(t)dt + 2\Phi_{pr}) \tag{6.10}$$

Thus, our resulting BVP signals can be defined as $BVP(t) = \hat{p}(t) + \hat{d}(t) + \hat{b}(t) + n(t)$. Constants C_1, C_2, and C_3 were found empirically, being $C_1 = 0.05$, $C_2 = 0.01$, and $C_3 = 0.15$. Based on [104], A is defined between 0.2 and 0.7, A_2 between 0 and 0.3, and B between 0.3 and 2. Finally, the frequency range for pulse rate is set between 0.7 and 3 Hz, and the breathing rate between 0.2 and 0.4 Hz. Figure 6.14 depicts some examples of the final synthetic BVP signals.

Following the process exposed in Figure 6.12, the BVP signal is not homogeneously distributed through the face, there are some parts of the face where the BVP magnitude is higher, such as the forehead or cheeks. Thus, we use the method developed in [118], where the authors propose a skin mask module to find the magnitude distribution of the BVP signal on the face. The skin mask is a pointwise multiplication between a binary mask by skin detection, and the attention mask generated with the pre-trained DeepPhys network. The BVP signal is embedded in the static video frame through the skin mask. Finally, we used the first-order motion model proposed in [87] to replicate the movements inside the database. Specifically, we took the first frame of one subject from the ECG-Fitness dataset and created the synthetic RPPG static video, then we replicated the motion from the scenarios performed in ECG-Fitness: speaking, rowing, exercising on a stationary bike and exercising on an elliptical trainer. We performed the same procedure with the 17 subjects from ECG-Fitness to generate a synthetic RPPG video database, using the movement scenarios of subject 00. The resulting synthetic database has 500 videos of a 1-minute duration. The pulse rate values are between 40 bpm and 180 bpm. Figure 6.15 depicts the pulse rate histogram of the 500 subjects using a 15-seconds slid-

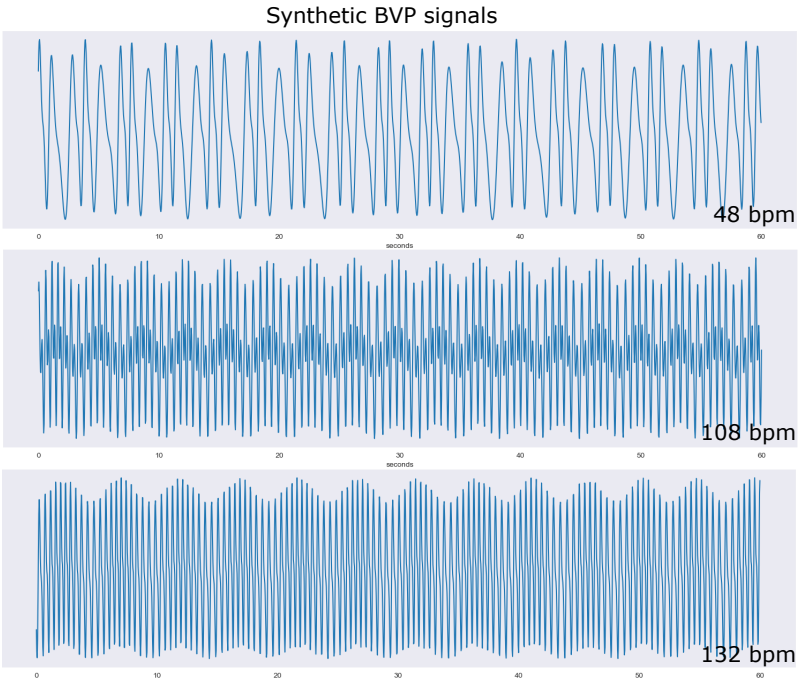

Figure 6.14: Examples of the final synthetic BVP signals.

ing window, 0.5-second step. Next section presents the RTRPPG pre-training on the new synthetic database.

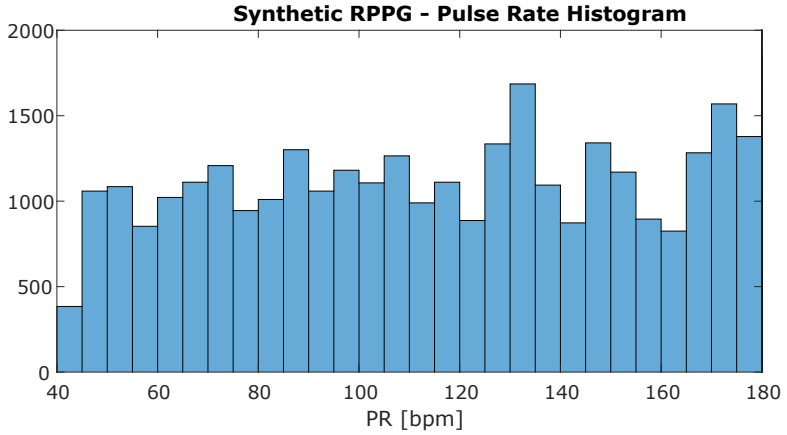

Figure 6.15: Synthetic RPPG database. Pulse rate histogram.

6.6.2.1/ PRE-TRAINING IN SYNTHETIC RPPG VIDEOS

We can perform a pre-training process on the RTRPPG network with the synthetic database and fine-tune it on the authentic databases. The results of this experiment are presented in table 6.4. Indeed, thanks to the pre-training of the RTRPPG network on the RPPG synthetic video network, the network learns the context of the RPPG measurement task, then, when performing fine-tuning on real data, the network is better trained than without the pre-training. The results obtained in UBFC-rPPG and COHFACE with pre-training are slightly better. The most interesting results are in ECG-Fitness, where all

pulse rate and signal quality metrics are greatly improved. Thus, we can conclude that with a suitable data augmentation strategy as the one used in this section, it is possible to increase the pulse rate and signal quality performance of a neural network, even on challenging databases where it is not possible to get a lot of videos with specific environments (specific range of movements, noise, light changes, etc.).

Table 6.4: RTRPPG Intra-dataset fine-tuning comparison

Fine-tuned	**Dataset**	**MAE** [bpm]	**r**	**SNR** [dB]	**TMC**
No	UBFC-rPPG	0.71±1.0	1	7.75±3.1	0.95±0.04
Yes	UBFC-rPPG	0.67±1.0	1	8.71±2.8	0.95±0.03
No	COHFACE	4.62±7.8	0.67	3.63±4.5	0.82±0.15
Yes	COHFACE	2.96±6.5	0.81	5.6±4.2	0.91±0.04
No	ECG-Fitness	27.08±19.5	0.08	-6.74±3.3	0.53±0.07
Yes	ECG-Fitness	8.83±13.5	0.75	0.22±4.2	0.74±0.07

6.7/ RTRPPG, PULSE RATE VARIABILITY METRICS

We study in this section, the potential of the RTRPPG network to be used in applications that require accurate pulse rate variability. To this end, we revisit the experiments performed in the previous section where we pre-trained RTRPPG on synthetic videos and performed fine-tuning on an easy (UBFC-rPPG), a moderate (COHFACE), and a challenging (ECG-Fitness) database, in an intra-dataset evaluation. In Table 6.5, we present the features related to pulse rate variability and compare them with PhysNet. In UBFC-rPPG, all four metrics are very close to those obtained by PhysNet. On the other hand, PhysNet performs better than RTRPPG in COHFACE; this may be related to the fact that PhysNet architecture is more robust to the highly compressed videos. Finally, ECG-Fitness is a more complicated database than COHFACE, not due to video compression, but light changes and subjects' movement. Thus, by pre-training RTRPPG on synthetic videos, this network can outperform PhysNet metrics. In the next section, we will use the LSTMDF filter in the RPPG signals estimated by RTRPPG.

6.8/ COMBINATION OF RTRPPG AND LSTMDF

In this section, we will smooth the output of the RTRPPG network with our LSTMDF filter. For this purpose, we will use the subject-independent 5-fold cross-validation evaluation in the intra-dataset scenario in the VIPL-HR database. RTRPPG is trained on the same subjects as LSTMDF in each fold. RTRPPG will estimate the RPPG signals from the videos, and then the estimated signals will be filtered by LSTMDF. Table 6.6 presents the results acquired by filtering the RPPG signal using the classical methods and the LSTMDF network. Best results are presented in bold.

Table 6.5: RTRPPG pulse rate variability metrics

	UBFC-rPPG			
	E:SDNN	**E:RMSSD**	**E:LF**	**E:HF**
PhysNet	0.020	0.0025	0.43	0.37
RTRPPG	0.022	0.0027	0.48	0.46
	COHFACE			
	E:SDNN	**E:RMSSD**	**E:LF**	**E:HF**
PhysNet	0.032	0.0041	0.30	0.49
RTRPPG	0.060	0.0070	0.53	1.02
	ECG-Fitness			
	E:SDNN	**E:RMSSD**	**E:LF**	**E:HF**
PhysNet	0.080	0.0086	1.17	1.98
RTRPPG	0.056	0.0056	0.69	1.22

Recall: the lower is the metric value, the best is the method

Table 6.6: Combination of RTRPPG and LSTMDF in VIPL-HR

Filtering method	**MAE** [bpm]	**r**	**SNR** [dB]	**TMC**
RTRPPG	3.99±8.21	0.73	4.59±4.47	0.89±0.07
RTRPPG+BP	3.75±7.44	0.76	4.8±4.39	0.89±0.06
RTRPPG+WV	3.73±7.26	0.77	4.7±4.47	0.90±0.06
RTRPPG+SG	3.98±8.48	0.73	4.62±4.57	0.90±0.06
RTRPPG+MTO	**3.66±7.34**	**0.77**	4.59±4.21	0.88±0.06
RTRPPG+MTM	3.81±7.74	0.74	**9.96±6.71**	**0.95±0.04**

Similar to the results in chapter 5, MTO performs better in estimating pulse rate metrics, but the SNR of 4.59 dB and TMC of 0.88 are almost the same as the RTRPPG results without filtering. MTM is the best choice, it improves pulse rate metrics, but also signal quality. Perhaps the most interesting result is the improvement of the signal quality, an SNR of 9.96 dB and TMC of 0.95 is considerably superior compared with the SNR of 4.59 dB and TMC of 0.89 without filtering. Thus, although the unfiltered RTRPPG model achieves state-of-the-art results, it is still possible to improve the signal quality by using a deep-filter. However, the RPPG measure module and the filtering module operate independently. In the conclusions section (7), we will mention future work to add a filtering layer to the end-to-end RPPG measurement model. The following is a summary of the discussion in this chapter.

6.9/ SUMMARY

In this chapter, we proposed a 3DCNN baseline and a series of experiments to find a fast and accurate network for acquiring reliable RPPG signals. The best configuration is referred to as Real-Time RPPG: RTRPPG. We showed that by decreasing the dimension of the input images, the inference speed is improved at the cost of accuracy drop in measuring RPPG signals. We proposed a joint solution showing that a temporal-frequency-based loss function is necessary for the network to learn the fundamental features of the input videos. Likewise, it was also shown that RTRPPG is robust to color representation, where YUV channels perform slightly better than other color channels. Interestingly, when comparing RTRPPG with the state-of-the-art PhysNet, a comparable accuracy to the RPPG signal estimation is achieved. At the same time, compared with PhysNet, our model improves the inference speed by about 96%, from 51.77 ms to 2.32 ms in GPU and from 816.47 ms to 28.65 ms in CPU.

Finally, we used a data augmentation strategy using synthetic RPPG videos to pre-train the RTRPPG network and perform fine-tuning on complex databases. This approach improved the pulse rate and signal quality metrics in the ECG-Fitness database, decreasing the MAE and r metrics from 27.08 bpm and 0.08 to 8.83 bpm and 0.75, respectively. Similarly, the pulse rate variability metrics are comparable and sometimes better than those acquired with a robust network such as PhysNet.

In the following section, we will present some general conclusions and perspectives on the whole of this thesis.

7

CONCLUSIONS AND FUTURE WORK

Remote photoplethysmography is a non-invasive technique to measure the BVP signal from video. We can estimate pulse rate, breathing rate, and pulse rate variability from a BVP signal. High-quality BVP signals allow a more precise physiological data measurement; however, recent RPPG measurements have focused on pulse rate measurement rather than signal quality. In this thesis, we used deep-learning-based methods to measure and filter RPPG signals. We will now present the conclusions found in the contributions of this thesis. We will also discuss some perspectives and future work.

In chapter 5, we proposed a new deep-learning-based framework to filter RPPG signals, Long Short-Term Memory Deep-Filter (LSTMDF). We used multiple protocols to compare the classical filters: bandpass, wavelet, and Savitzky-Golay, with the two sequence-to-sequence versions of the LSTMDF, many-to-many (MTM) and many-to-one (MTO). The experiments were conducted on three public databases, leading to the following conclusions:

1. A relatively low number of signals is enough to train the LSTMDF efficiently, a training set of approximately 90 signals totaling 45 minutes was sufficient.
2. In an intra-dataset scenario, both MTO and MTM overperform the conventional filters, but the MTM approach gives the best results.
3. LSTMDF is stable when filtering RPPG signals estimated from different state-of-the-art RPPG measurement methods.
4. We experimentally observed a dependence of the LSTMDF filter on the RPPG signal-to-noise ratio (RPPG-SNR). It is recommended that the RPPG-SNR average of the training set has to be as close as possible to those of the test set; if so, we can expect the LSTMDF to overperform classical filters even in a cross-dataset scenario.
5. Recurrent neural networks are a better choice than classical filtering methods to preserve the characteristic frequency and interestingly the shape of the RPPG signal.

Regarding pulse rate variability metrics, our results show that, in fact, LSTMDF allows estimating a cleaner RPPG. That is, the error is smaller with LSTMDF than with classical filters. Nevertheless, from the values found, it is difficult to conclude whether the improvement is sufficient for stress estimation or atrial fibrillation applications. Therefore, in future

work, we will evaluate the relevance of using the LSTMDF filter in applications where the shape of the BVP signal is critical.

Our method was applied to the RPPG signal filtering task. However, the temporal and frequency characteristics of this type of signal are present in other biomedical signals, e.g. ECG. In future work, we plan to use our filtering method for different periodic signals, such as ECG. On the other hand, it is also hypothesized whether our filtering method could be helpful in non-periodic biomedical signals. Therefore we will also evaluate the LSTMDF filter in EEG signals in future work. Finally, we believe that our filter could be applied even to non-biomedical signals. In future work, we will review other applications where it is crucial to preserve the temporal and shape characteristics of signals.

In chapter 6, we proposed the end-to-end RPPG measurement framework RTRPPG which stands for Real-Time RPPG (Available online[1]). RTRPPG was designed to be used in real-time and maintain the same pulse rate measurement and signal quality accuracy as the end-to-end RPPG measurement methods found in the literature, such as PhysNet. Compared with PhysNet, our model improves the inference speed by about 96%, from 51.77 ms to 2.32 ms in GPU and from 816.47 ms to 28.65 ms in CPU. In the larger database VIPL-HR, in pulse rate metrics, RTRPPG reached an MAE of 3.99 bpm and r of 0.73, while PhysNet gives an MAE of 3.87 bpm and r of 0.73. In signal quality, the SNR and TMC metrics reached by RTRPPG were 4.6 dB and 0.89, respectively. PhysNet allows SNR of 5 dB and TMC of 0.91. Therefore, our framework can be used in real-time applications in low-cost GPU and CPU devices with state-of-the-art pulse rate measurement and signal quality performance. Finally, in the VIPL-HR dataset, we decreased the training time in our GPU hardware from 1038 minutes (PhysNet) to 76.8 minutes (RTRPPG), which gives a lot more flexibility to perform more experiments and tuning.

We used an ablation study to find the optimal configuration of the RTRPPG architecture. We started from a 3DCNN baseline network and experimentally studied different pipeline configurations to acquire a reasonable pulse rate and signal quality performance, along with real-time implementation. Our ablation study is rigorous and can be used to optimize deep-learning-based architectures in other computer vision tasks. The following are the steps and their conclusions about finding the optimal architecture of RTRPPG.

1. Input frames resizing: It can be considered as a spatial filtering procedure. Decreasing the size of the input image decreases the number of operations during forward propagation and, therefore, also the training and inference time. The best input resolution in RTRPPG was 8x8 frames.

2. Custom loss function: A customized loss function can help in the convergence of the neural network. The NPSNR loss function used by RTRPPG evaluates the temporal and frequency characteristics of the BVP signals.

3. Color channel: Some color spaces preserve or highlight important features of the BVP within the scene.

In our experiments with RTRPPG, we demonstrated that an RPPG measurement model that works in real-time on GPU and CPU is possible. As a next step, in future work, we

[1] https://github.com/deividbotina-alv/rtrppg

will implement the RTRPPG network on embedded devices to demonstrate the versatility of our method.

In chapter 6, we also used a data augmentation strategy based on synthetic RPPG video generation. The process consists in taking an image of a static subject, and then it is duplicated multiple times, generating a video clip. A synthetic PPG signal is added to the skin subject within the video. Finally, motion is transferred to the subject using a pre-trained first-order motion model. This method can augment data in specific scenarios with characteristic light changes of movements. We assessed this data augmentation process as a pre-training step for a challenging database, and the results showed a significant improvement. This methodology can potentially be used in any environment. With few images and video data, it is possible to create a challenging RPPG synthetic database.

We used the RTRPPG network and the data augmentation method in the RPPG measurement task. However, in [124], Roy *et al.* showed the potential of 3DCNNs in applications such as deep-fake mask detection. Thus, in future work, we will evaluate the RTRPPG and the RPPG synthetic videos in the deep-fake mask detection task. Likewise, the scientific community has proposed interesting challenges aimed at using benchmark databases to measure heart rate, inter-beat intervals (which require accurate measurement of each individual pulse peak), and respiration rate. Consequently, we will participate in the third challenge on Remote Physiological Signal Sensing (RePSS) [114, 98] in future work. For this purpose, instead of using RTRPPG and LSTMDF independently as we did in section 6.8, we will present a unified model using the RTRPPG architecture with a filtering module based on LSTMDF. The new architecture shall preserve the real-time component.

The objectives set out in the introduction of this thesis have been fulfilled. During this thesis, we have developed two deep-learning-based frameworks which add value to the filtering and estimation of RPPG signals tasks. The research related to these methods has been validated and published in one international journal and three international conferences and workshops, a list which is presented in the Annexes section A.1. We hope and believe that the research undertaken in this thesis will be a valuable addition to the scientific community, not only in the domain of RPPG signal estimation and filtering but also in other artificial vision tasks and generic biomedical signal analysis.

BIBLIOGRAPHY

[1] NASA. **Ballistocardiography: A bibliography**. Tech. Rep. FAA-AM-65-15, National Aeronautics and Space Administration, September 1965.

[2] AKIMA, H. **A new method of interpolation and smooth curve fitting based on local procedures**. *Journal of the ACM (JACM) 17*, 4 (1970), 589–602.

[3] RUMELHART, D. E., DURBIN, R., GOLDEN, R., AND CHAUVIN, Y. **Backpropagation: The basic theory**. *Backpropagation: Theory, architectures and applications* (1995), 1–34.

[4] HOCHREITER, S., AND SCHMIDHUBER, J. **Long short-term memory**. *Neural computation 9*, 8 (1997), 1735–1780.

[5] HYVARINEN, A. **Fast and robust fixed-point algorithms for independent component analysis**. *IEEE transactions on Neural Networks 10*, 3 (1999), 626–634.

[6] HYVÄRINEN, A., AND OJA, E. **Independent component analysis: algorithms and applications**. *Neural networks 13*, 4-5 (2000), 411–430.

[7] ALLEN, J. **Photoplethysmography and its application in clinical physiological measurement**. *Physiological measurement 28*, 3 (2007), R1.

[8] CONAIRE, C. O., O'CONNOR, N. E., AND SMEATON, A. F. **Detector adaptation by maximising agreement between independent data sources**. In *2007 IEEE Conference on Computer Vision and Pattern Recognition* (2007), IEEE, pp. 1–6.

[9] VERKRUYSSE, W., SVAASAND, L. O., AND NELSON, J. S. **Remote plethysmographic imaging using ambient light.** *Optics express 16*, 26 (2008), 21434–21445.

[10] POH, M.-Z., MCDUFF, D. J., AND PICARD, R. W. **Advancements in noncontact, multiparameter physiological measurements using a webcam**. *IEEE transactions on biomedical engineering 58*, 1 (2010), 7–11.

[11] POH, M.-Z., MCDUFF, D. J., AND PICARD, R. W. **Non-contact, automated cardiac pulse measurements using video imaging and blind source separation.** *Optics express 18*, 10 (2010), 10762–10774.

[12] LEWANDOWSKA, M., RUMINSKI, J., KOCEJKO, T., AND NOWAK, J. **Measuring pulse rate with a webcam—a non-contact method for evaluating cardiac activity**. In *2011 federated conference on computer science and information systems (FedCSIS)* (2011), IEEE, pp. 405–410.

[13] SOLEYMANI, M., LICHTENAUER, J., PUN, T., AND PANTIC, M. **A multimodal database for affect recognition and implicit tagging**. *IEEE transactions on affective computing 3*, 1 (2011), 42–55.

[14] TSOURI, G. R., KYAL, S., DIANAT, S. A., AND MESTHA, L. K. **Constrained independent component analysis approach to nonobtrusive pulse rate measurements**. *Journal of biomedical optics 17*, 7 (2012), 077011.

[15] WU, H.-Y., RUBINSTEIN, M., SHIH, E., GUTTAG, J., DURAND, F., AND FREEMAN, W. **Eulerian video magnification for revealing subtle changes in the world**. *ACM transactions on graphics (TOG) 31*, 4 (2012), 1–8.

[16] BOUSEFSAF, F., MAAOUI, C., AND PRUSKI, A. **Continuous wavelet filtering on webcam photoplethysmographic signals to remotely assess the instantaneous heart rate**. *Biomedical Signal Processing and Control 8*, 6 (2013), 568–574.

[17] BOUSEFSAF, F., MAAOUI, C., AND PRUSKI, A. **Continuous wavelet filtering on webcam photoplethysmographic signals to remotely assess the instantaneous heart rate**. *Biomedical Signal Processing and Control 8*, 6 (2013), 568–574.

[18] DE HAAN, G., AND JEANNE, V. **Robust pulse rate from chrominance-based rppg**. *IEEE Transactions on Biomedical Engineering 60*, 10 (2013), 2878–2886.

[19] CHO, K., VAN MERRIËNBOER, B., BAHDANAU, D., AND BENGIO, Y. **On the properties of neural machine translation: Encoder-decoder approaches**. *arXiv preprint arXiv:1409.1259* (2014).

[20] DE HAAN, G., AND VAN LEEST, A. **Improved motion robustness of remote-ppg by using the blood volume pulse signature**. *Physiological measurement 35*, 9 (2014), 1913.

[21] ESTEPP, J. R., BLACKFORD, E. B., AND MEIER, C. M. **Recovering pulse rate during motion artifact with a multi-imager array for non-contact imaging photoplethysmography**. In *2014 IEEE international conference on systems, man, and cybernetics (SMC)* (2014), IEEE, pp. 1462–1469.

[22] LI, X., CHEN, J., ZHAO, G., AND PIETIKAINEN, M. **Remote heart rate measurement from face videos under realistic situations**. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2014), pp. 4264–4271.

[23] ORPHANIDOU, C., BONNICI, T., CHARLTON, P., CLIFTON, D., VALLANCE, D., AND TARASSENKO, L. **Signal-quality indices for the electrocardiogram and photoplethysmogram: Derivation and applications to wireless monitoring**. *IEEE journal of biomedical and health informatics 19*, 3 (2014), 832–838.

[24] STRICKER, R., MÜLLER, S., AND GROSS, H.-M. **Non-contact video-based pulse rate measurement on a mobile service robot**. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication* (2014), IEEE, pp. 1056–1062.

[25] KAMSHILIN, A. A., NIPPOLAINEN, E., SIDOROV, I. S., VASILEV, P. V., EROFEEV, N. P., PODOLIAN, N. P., AND ROMASHKO, R. V. **A new look at the essence of the imaging photoplethysmography**. *Scientific reports 5*, 1 (2015), 1–9.

[26] LAM, A., AND KUNO, Y. **Robust heart rate measurement from video using select random patches**. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 3640–3648.

[27] McDuff, D. J., Estepp, J. R., Piasecki, A. M., and Blackford, E. B. **A survey of remote optical photoplethysmographic imaging methods**. In *2015 37th annual international conference of the IEEE engineering in medicine and biology society (EMBC)* (2015), IEEE, pp. 6398–6404.

[28] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., and others. **Imagenet large scale visual recognition challenge**. *International journal of computer vision 115*, 3 (2015), 211–252.

[29] Sun, Y., and Thakor, N. **Photoplethysmography revisited: from contact to noncontact, from point to imaging**. *IEEE Transactions on Biomedical Engineering 63*, 3 (2015), 463–477.

[30] Tsouri, G. R., and Li, Z. **On the benefits of alternative color spaces for noncontact heart rate measurements using standard red-green-blue cameras**. *Journal of biomedical optics 20*, 4 (2015), 048002.

[31] Allard, U. C., Nougarou, F., Fall, C. L., Giguère, P., Gosselin, C., Laviolette, F., and Gosselin, B. **A convolutional neural network for robotic arm guidance using semg based frequency-features**. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2016), IEEE, pp. 2464–2470.

[32] Atzori, M., Cognolato, M., and Müller, H. **Deep learning with convolutional neural networks applied to electromyography data: A resource for the classification of movements for prosthetic hands**. *Frontiers in neurorobotics 10* (2016), 9.

[33] Bhandare, A., Bhide, M., Gokhale, P., and Chandavarkar, R. **Applications of convolutional neural networks**. *International Journal of Computer Science and Information Technologies 7*, 5 (2016), 2206–2215.

[34] Chen, W., Hernandez, J., and Picard, R. W. **Non-contact physiological measurements from near-infrared video of the neck**. *Submitted to Biomedical Optic Express* (2016).

[35] Geng, W., Du, Y., Jin, W., Wei, W., Hu, Y., and Li, J. **Gesture recognition by instantaneous surface emg images**. *Scientific reports 6*, 1 (2016), 1–8.

[36] He, K., Zhang, X., Ren, S., and Sun, J. **Deep residual learning for image recognition**. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778.

[37] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. **Ssd: Single shot multibox detector**. In *European conference on computer vision* (2016), Springer, pp. 21–37.

[38] Park, K.-H., and Lee, S.-W. **Movement intention decoding based on deep learning for multiuser myoelectric interfaces**. In *2016 4th international winter conference on brain-computer Interface (BCI)* (2016), IEEE, pp. 1–2.

[39] Ruder, S. **An overview of gradient descent optimization algorithms**. *arXiv preprint arXiv:1609.04747* (2016).

[40] WAND, M., AND SCHMIDHUBER, J. **Deep neural network frontend for continuous emg-based speech recognition.** In *Interspeech* (2016), pp. 3032–3036.

[41] WANG, W., DEN BRINKER, A. C., STUIJK, S., AND DE HAAN, G. **Algorithmic principles of remote ppg**. *IEEE Transactions on Biomedical Engineering 64*, 7 (2016), 1479–1491.

[42] ZHANG, Z., GIRARD, J. M., WU, Y., ZHANG, X., LIU, P., CIFTCI, U., CANAVAN, S., REALE, M., HOROWITZ, A., YANG, H., AND OTHERS. **Multimodal spontaneous emotion corpus for human behavior analysis**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 3438–3446.

[43] CHEN, W., AND PICARD, R. W. **Eliminating physiological information from facial videos**. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)* (2017), IEEE, pp. 48–55.

[44] HASSAN, M. A., MALIK, A. S., FOFI, D., SAAD, N., KARASFI, B., ALI, Y. S., AND MERIAUDEAU, F. **Heart rate estimation using facial video: A review**. *Biomedical Signal Processing and Control 38* (2017), 346–360.

[45] HEUSCH, G., ANJOS, A., AND MARCEL, S. **A reproducible study on remote heart rate measurement**. *arXiv preprint arXiv:1709.00962* (2017).

[46] HSU, G.-S., AMBIKAPATHI, A., AND CHEN, M.-S. **Deep learning with time-frequency representation for pulse estimation from facial videos**. In *2017 IEEE international joint conference on biometrics (IJCB)* (2017), IEEE, pp. 383–389.

[47] HURTER, C., AND MCDUFF, D. **Cardiolens: remote physiological monitoring in a mixed reality environment**. In *ACM siggraph 2017 emerging technologies*. Association for Computing Machinery, 2017, pp. 1–2.

[48] SHAFFER, F., AND GINSBERG, J. P. **An overview of heart rate variability metrics and norms**. *Frontiers in public health* (2017), 258.

[49] WANG, W., STUIJK, S., AND DE HAAN, G. **Living-skin classification via remote-ppg**. *IEEE Transactions on biomedical engineering 64*, 12 (2017), 2781–2792.

[50] WU, B.-F., HUANG, P.-W., TSOU, T.-Y., LIN, T.-M., AND CHUNG, M.-L. **Camera-based heart rate measurement using continuous wavelet transform**. In *2017 International Conference on System Science and Engineering (ICSSE)* (2017), IEEE, pp. 7–11.

[51] ZHAI, X., JELFS, B., CHAN, R. H., AND TIN, C. **Self-recalibrating surface emg pattern recognition for neuroprosthesis control based on convolutional neural network**. *Frontiers in neuroscience 11* (2017), 379.

[52] BAI, G., HUANG, J., AND LIU, H. **Real-time robust noncontact heart rate monitoring with a camera**. *IEEE Access 6* (2018), 33682–33691.

[53] BENEZETH, Y., LI, P., MACWAN, R., NAKAMURA, K., GOMEZ, R., AND YANG, F. **Remote heart rate variability for emotional state monitoring**. In *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)* (2018), IEEE, pp. 153–156.

[54] CHEN, W., AND MCDUFF, D. **Deepphys: Video-based physiological measurement using convolutional attention networks**. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 349–365.

[55] FAUST, O., HAGIWARA, Y., HONG, T. J., LIH, O. S., AND ACHARYA, U. R. **Deep learning for healthcare applications based on physiological signals: A review**. *Computer methods and programs in biomedicine 161* (2018), 1–13.

[56] LEE, J., SUN, S., YANG, S. M., SOHN, J. J., PARK, J., LEE, S., AND KIM, H. C. **Bidirectional recurrent auto-encoder for photoplethysmogram denoising**. *IEEE journal of biomedical and health informatics 23*, 6 (2018), 2375–2385.

[57] LI, X., ALIKHANI, I., SHI, J., SEPPANEN, T., JUNTTILA, J., MAJAMAA-VOLTTI, K., TULPPO, M., AND ZHAO, G. **The obf database: A large face video database for remote physiological signal measurement and atrial fibrillation detection**. In *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)* (2018), IEEE, pp. 242–249.

[58] LIU, S., HUANG, D., AND OTHERS. **Receptive field block net for accurate and fast object detection**. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 385–400.

[59] MACWAN, R., BENEZETH, Y., AND MANSOURI, A. **Remote photoplethysmography with constrained ica using periodicity and chrominance constraints**. *Biomedical engineering online 17*, 1 (2018), 1–22.

[60] MACWAN, R., BOBBIA, S., BENEZETH, Y., DUBOIS, J., AND MANSOURI, A. **Periodic variance maximization using generalized eigenvalue decomposition applied to remote photoplethysmography estimation**. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2018), pp. 1332–1340.

[61] NIU, X., HAN, H., SHAN, S., AND CHEN, X. **Synrhythm: Learning a deep heart rate estimator from general to specific**. In *2018 24th International Conference on Pattern Recognition (ICPR)* (2018), IEEE, pp. 3580–3585.

[62] NIU, X., HAN, H., SHAN, S., AND CHEN, X. **Vipl-hr: A multi-modal database for pulse estimation from less-constrained face video**. In *Asian Conference on Computer Vision* (2018), Springer, pp. 562–576.

[63] OH, S. L., NG, E. Y., SAN TAN, R., AND ACHARYA, U. R. **Automated diagnosis of arrhythmia using combination of cnn and lstm techniques with variable length heart beats**. *Computers in biology and medicine 102* (2018), 278–287.

[64] PO, L.-M., FENG, L., LI, Y., XU, X., CHEUNG, T. C.-H., AND CHEUNG, K.-W. **Block-based adaptive roi for remote photoplethysmography**. *Multimedia Tools and Applications 77*, 6 (2018), 6503–6529.

[65] ŠPETLÍK, R., FRANC, V., AND MATAS, J. **Visual heart rate estimation with convolutional neural network**. In *Proceedings of the british machine vision conference, Newcastle, UK* (2018), pp. 3–6.

[66] TRUMPP, A., LOHR, J., WEDEKIND, D., SCHMIDT, M., BURGHARDT, M., HELLER, A. R., MALBERG, H., AND ZAUNSEDER, S. **Camera-based photoplethysmography in an intraoperative setting**. *Biomedical engineering online 17*, 1 (2018), 1–19.

[67] TSIOURIS, K. M., PEZOULAS, V. C., ZERVAKIS, M., KONITSIOTIS, S., KOUTSOURIS, D. D., AND FOTIADIS, D. I. **A long short-term memory deep learning network for the prediction of epileptic seizures using eeg signals**. *Computers in biology and medicine 99* (2018), 24–37.

[68] XIA, P., HU, J., AND PENG, Y. **Emg-based estimation of limb movement using deep learning with recurrent convolutional neural networks**. *Artificial organs 42*, 5 (2018), E67–E77.

[69] YILDIRIM, Ö. **A novel wavelet sequence based on deep bidirectional lstm network model for ecg signal classification**. *Computers in biology and medicine 96* (2018), 189–202.

[70] ZHANG, Q., ZHOU, Y., SONG, S., LIANG, G., AND NI, H. **Heart rate extraction based on near-infrared camera: Towards driver state monitoring**. *IEEE Access 6* (2018), 33076–33087.

[71] BAZAREVSKY, V., KARTYNNIK, Y., VAKUNOV, A., RAVEENDRAN, K., AND GRUNDMANN, M. **Blazeface: Sub-millisecond neural face detection on mobile gpus**. *arXiv preprint arXiv:1907.05047* (2019).

[72] BELAICHE, R., MEZIATI SABOUR, R., MIGNIOT, C., BENEZETH, Y., GINHAC, D., NAKAMURA, K., GOMEZ, R., AND YANG, F. **Emotional state recognition with micro-expressions and pulse rate variability**. In *International Conference on Image Analysis and Processing* (2019), Springer, pp. 26–35.

[73] BIAN, M., PENG, B., WANG, W., AND DONG, J. **An accurate lstm based video heart rate estimation method**. In *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)* (2019), Springer, pp. 409–417.

[74] BOBBIA, S., MACWAN, R., BENEZETH, Y., MANSOURI, A., AND DUBOIS, J. **Unsupervised skin tissue segmentation for remote photoplethysmography**. *Pattern Recognition Letters 124* (2019), 82–90.

[75] BOUSEFSAF, F., PRUSKI, A., AND MAAOUI, C. **3d convolutional neural networks for remote pulse rate measurement and mapping from facial video**. *Applied Sciences 9*, 20 (2019), 4364.

[76] CHAICHULEE, S., VILLARROEL, M., JORGE, J., ARTETA, C., MCCORMICK, K., ZISSERMAN, A., AND TARASSENKO, L. **Cardio-respiratory signal extraction from video camera data for continuous non-contact vital sign monitoring using deep learning**. *Physiological measurement 40*, 11 (2019), 115001.

[77] DA POIAN, G., LETIZIA, N. A., RINALDO, R., AND CLIFFORD, G. D. **A low-complexity photoplethysmographic systolic peak detector for compressed sensed data**. *Physiological measurement 40*, 6 (2019), 065007.

[78] DVORNIK, N., MAIRAL, J., AND SCHMID, C. **On the importance of visual context for data augmentation in scene understanding**. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).

[79] KARNAN, H., SIVAKUMARAN, N., AND MANIVEL, R. **An efficient cardiac arrhythmia onset detection technique using a novel feature rank score algorithm**. *Journal of medical systems 43*, 6 (2019), 1–8.

[80] KOPELIOVICH, M., MIRONENKO, Y., AND PETRUSHAN, M. **Architectural tricks for deep learning in remote photoplethysmography**. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops* (2019), pp. 0–0.

[81] MEYES, R., LU, M., DE PUISEAU, C. W., AND MEISEN, T. **Ablation studies in artificial neural networks**. *arXiv preprint arXiv:1901.08644* (2019).

[82] NIU, X., SHAN, S., HAN, H., AND CHEN, X. **Rhythmnet: End-to-end heart rate estimation from face via spatial-temporal representation**. *IEEE Transactions on Image Processing 29* (2019), 2409–2423.

[83] NIU, X., ZHAO, X., HAN, H., DAS, A., DANTCHEVA, A., SHAN, S., AND CHEN, X. **Robust remote heart rate estimation from face utilizing spatial-temporal attention**. In *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)* (2019), IEEE, pp. 1–8.

[84] NOWARA, E., AND MCDUFF, D. **Combating the impact of video compression on non-contact vital sign measurement using supervised learning**. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops* (2019), pp. 0–0.

[85] PIERRART, P. **Etude et implémentation du deep-learning pour la photopléthysmographie sans contact**. M.s. thesis, Laboratoire ImVia – Université de Bourgogne- Franche-Compté, 64 Rue Sully, Dijon, France, 2019.

[86] RASCHKA, S., AND MIRJALILI, V. **Python Machine Learning, 3rd Ed.**, 3 ed. Packt Publishing, Birmingham, UK, 2019.

[87] SIAROHIN, A., LATHUILIÈRE, S., TULYAKOV, S., RICCI, E., AND SEBE, N. **First order motion model for image animation**. In *Conference on Neural Information Processing Systems (NeurIPS)* (December 2019).

[88] SLAPNICAR, G., DOVGAN, E., CUK, P., AND LUSTREK, M. **Contact-free monitoring of physiological parameters in people with profound intellectual and multiple disabilities**. In *Proceedings of the IEEE International Conference on Computer Vision Workshops* (2019), pp. 0–0.

[89] VERMA, S. **Understanding 1d and 3d convolution neural network — keras**, September 2019. Reviewed on 10th August 2022.

[90] YU, Z., LI, X., AND ZHAO, G. **Remote photoplethysmograph signal measurement from facial videos using spatio-temporal networks**. *arXiv preprint arXiv:1905.02419* (2019).

[91] YU, Z., PENG, W., LI, X., HONG, X., AND ZHAO, G. **Remote heart rate measurement from highly compressed facial videos: an end-to-end deep learning solution with video enhancement**. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 151–160.

[92] YUEN, P. C., LIU, S., ZHANG, S., AND ZHAO, G. **3d mask face anti-spoofing with remote photoplethysmography**, Aug. 13 2019. US Patent 10,380,444.

[93] BOTINA-MONSALVE, D., BENEZETH, Y., MACWAN, R., PIERRART, P., PARRA, F., NAKAMURA, K., GOMEZ, R., AND MITERAN, J. **Long short-term memory deep-filter in remote photoplethysmography**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2020), pp. 306–307.

[94] CHAN, H.-P., HADJIISKI, L. M., AND SAMALA, R. K. **Computer-aided diagnosis in the era of deep learning**. *Medical physics 47*, 5 (2020), e218–e227.

[95] FINŽGAR, M., AND PODRŽAJ, P. **Feasibility of assessing ultra-short-term pulse rate variability from video recordings**. *PeerJ 8* (2020), e8342.

[96] HUANG, B., CHANG, C.-M., LIN, C.-L., CHEN, W., JUANG, C.-F., AND WU, X. **Visual heart rate estimation from facial video based on cnn**. In *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)* (2020), IEEE, pp. 1658–1662.

[97] LEE, E., CHEN, E., AND LEE, C.-Y. **Meta-rppg: Remote heart rate estimation using a transductive meta-learner**. In *European Conference on Computer Vision* (2020), Springer, pp. 392–409.

[98] LI, X., HAN, H., LU, H., NIU, X., YU, Z., DANTCHEVA, A., ZHAO, G., AND SHAN, S. **The 1st challenge on remote physiological signal sensing (repss)**. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops* (June 2020).

[99] LIU, S.-Q., AND YUEN, P. C. **A general remote photoplethysmography estimator with spatiotemporal convolutional network**. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)* (2020), IEEE, pp. 481–488.

[100] LIU, X., FROMM, J., PATEL, S., AND MCDUFF, D. **Multi-task temporal shift attention networks for on-device contactless vitals measurement**. *Advances in Neural Information Processing Systems 33* (2020), 19400–19411.

[101] MIRONENKO, Y., KALININ, K., KOPELIOVICH, M., AND PETRUSHAN, M. **Remote photoplethysmography: Rarely considered factors**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2020), pp. 296–297.

[102] NIU, X., YU, Z., HAN, H., LI, X., SHAN, S., AND ZHAO, G. **Video-based remote physiological measurement via cross-verified feature disentangling**. In *European Conference on Computer Vision* (2020), Springer, pp. 295–310.

[103] NOWARA, E. M., MARKS, T. K., MANSOUR, H., AND VEERARAGHAVAN, A. **Near-infrared imaging photoplethysmography during driving**. *IEEE Transactions on Intelligent Transportation Systems* (2020).

[104] PEREPELKINA, O., ARTEMYEV, M., CHURIKOVA, M., AND GRINENKO, M. **Heartrack: Convolutional neural network for remote video-based heart rate monitoring**. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2020), pp. 288–289.

[105] SONG, R., ZHANG, S., LI, C., ZHANG, Y., CHENG, J., AND CHEN, X. **Heart rate estimation from facial videos using a spatiotemporal representation with convolutional neural networks**. *IEEE Transactions on Instrumentation and Measurement 69*, 10 (2020), 7411–7421.

[106] TSOU, Y.-Y., LEE, Y.-A., HSU, C.-T., AND CHANG, S.-H. **Siamese-rppg network: Remote photoplethysmography signal estimation from face videos**. In *Proceedings of the 35th annual ACM symposium on applied computing* (2020), pp. 2066–2073.

[107] YANG, S., YU, X., AND ZHOU, Y. **Lstm and gru neural network performance comparison study: Taking yelp review dataset as an example**. In *2020 International workshop on electronic communication and artificial intelligence (IWECAI)* (2020), IEEE, pp. 98–101.

[108] YU, Z., LI, X., NIU, X., SHI, J., AND ZHAO, G. **Autohr: A strong end-to-end baseline for remote heart rate measurement with neural searching**. *IEEE Signal Processing Letters 27* (2020), 1245–1249.

[109] ZHAN, Q., WANG, W., AND DE HAAN, G. **Analysis of cnn-based remote-ppg to understand limitations and sensitivities**. *Biomedical Optics Express 11*, 3 (2020), 1268–1283.

[110] CHOW, J. Y., VADAKKEN, M. E., WHITLOCK, R. P., KOZIARZ, A., AINSWORTH, C., AMIN, F., MCINTYRE, W. F., DEMERS, C., AND BELLEY-CÔTÉ, E. P. **Pulmonary artery catheterization in patients with cardiogenic shock: a systematic review and meta-analysis**. *Canadian Journal of Anesthesia/Journal canadien d'anesthésie* (2021), 1–19.

[111] DE DEUS, L. F., SEHGAL, N., AND TALUKDAR, D. **Evaluating visual photoplethysmography method**. *medRxiv* (2021).

[112] GIDEON, J., AND STENT, S. **The way to my heart is through contrastive learning: Remote photoplethysmography from unlabelled video**. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 3995–4004.

[113] KIM, S.-E., YU, S.-G., KIM, N. H., SUH, K. H., AND LEE, E. C. **Restoration of remote ppg signal through correspondence with contact sensor signal**. *Sensors 21*, 17 (2021), 5910.

[114] LI, X., SUN, H., SUN, Z., HAN, H., DANTCHEVA, A., SHAN, S., AND ZHAO, G. **The 2nd challenge on remote physiological signal sensing (repss)**. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 2404–2413.

[115] LUGUERN, D., MACWAN, R., BENEZETH, Y., MOSER, V., DUNBAR, L. A., BRAUN, F., LEMKADDEM, A., AND DUBOIS, J. **Wavelet variance maximization: a contactless respiration rate estimation method based on remote photoplethysmography**. *Biomedical Signal Processing and Control 63* (2021), 102263.

[116] PERCHE, S. **Expérimentations et développement logiciel pour la mesure rppg**. M.s. thesis, Laboratoire ImVia – Université de Bourgogne- Franche-Compté, 64 Rue Sully, Dijon, France, 2021.

[117] PERCHE, S., BOTINA, D., BENEZETH, Y., NAKAMURA, K., GOMEZ, R., AND MITERAN, J. **Data-augmentation for deep learning based remote photoplethysmography methods**. In *2021 International Conference on e-Health and Bioengineering (EHB)* (2021), IEEE, pp. 1–4.

[118] SABOKROU, M., POURREZA, M., LI, X., FATHY, M., AND ZHAO, G. **Deep-hr: Fast heart rate estimation from face video under realistic conditions**. *Expert Systems with Applications 186* (2021), 115596.

[119] SPETH, J., VANCE, N., FLYNN, P., BOWYER, K., AND CZAJKA, A. **Unifying frame rate and temporal dilations for improved remote pulse detection**. *Computer Vision and Image Understanding 210* (2021), 103246.

[120] VO, K., NAEINI, E. K., NADERI, A., JILANI, D., RAHMANI, A. M., DUTT, N., AND CAO, H. **P2e-wgan: Ecg waveform synthesis from ppg with conditional wasserstein generative adversarial networks**. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing* (2021), pp. 1030–1036.

[121] YU, Z., LI, X., WANG, P., AND ZHAO, G. **Transrppg: Remote photoplethysmography transformer for 3d mask face presentation attack detection**. *IEEE Signal Processing Letters 28* (2021), 1290–1294.

[122] ZHANG, P., LI, B., PENG, J., AND JIANG, W. **Multi-hierarchical convolutional network for efficient remote photoplethysmograph signal and heart rate estimation from face video clips**. *arXiv preprint arXiv:2104.02260* (2021).

[123] HORVATH, B. **Synthetic data for deep learning**. *Quantitative Finance 22*, 3 (2022), 423–425.

[124] ROY, R., JOSHI, I., DAS, A., AND DANTCHEVA, A. **3d cnn architectures and attention mechanisms for deepfake detection**. In *Handbook of Digital Face Manipulation and Detection*. Springer, Cham, 2022, pp. 213–234.

LIST OF FIGURES

List of Tables

III

ANNEXES

A

ADDITIONAL INFORMATION

A.1/ LIST OF PUBLICATIONS

A.1.1/ INTERNATIONAL JOURNALS

BOTINA-MONSALVE Deivid, Benezeth Yannick, Miteran Johel (2022). Performance Analysis of Remote Photoplethysmography Deep Filtering Using Long Short-Term Memory Neural Network. In BioMedical Engineering OnLine (BMEO).

A.1.2/ INTERNATIONAL CONFERENCES AND WORKSHOPS

BOTINA-MONSALVE Deivid, Benezeth Yannick, Miteran Johel. (2022). RTrPPG: An Ultra Light 3DCNN for Real-Time Remote Photoplethysmography. In. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop (CVPR). (pp. 2146–2154).

Perche Simon, **BOTINA-MONSALVE Deivid**, Benezeth Yannick, Nakamura Keisuke, Gomez Randy, Miteran Johel. (2021). Data-Augmentation for Deep Learning Based Remote Photoplethysmography Methods. In. 2021 International Conference on e-Health and Bioengineering (EHB). (pp. 1–4).

BOTINA-MONSALVE Deivid, Benezeth Yannick, Macwan Richard, Pierrart Paul, Parra Federico, Nakamura Keisuke, Gomez Randy, Miteran Johel. (2020). Long short-term memory deep-filter in remote photoplethysmography. In. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop (CVPR). (pp. 306–307).

A.2/ DATABASE DESCRIPTION PER FOLD

This section presents the signal duration histograms and the pulse-rate histogram of the ground truth signals for all the folds in all the databases used. This information is complementary to that presented in Section 4.2.2.

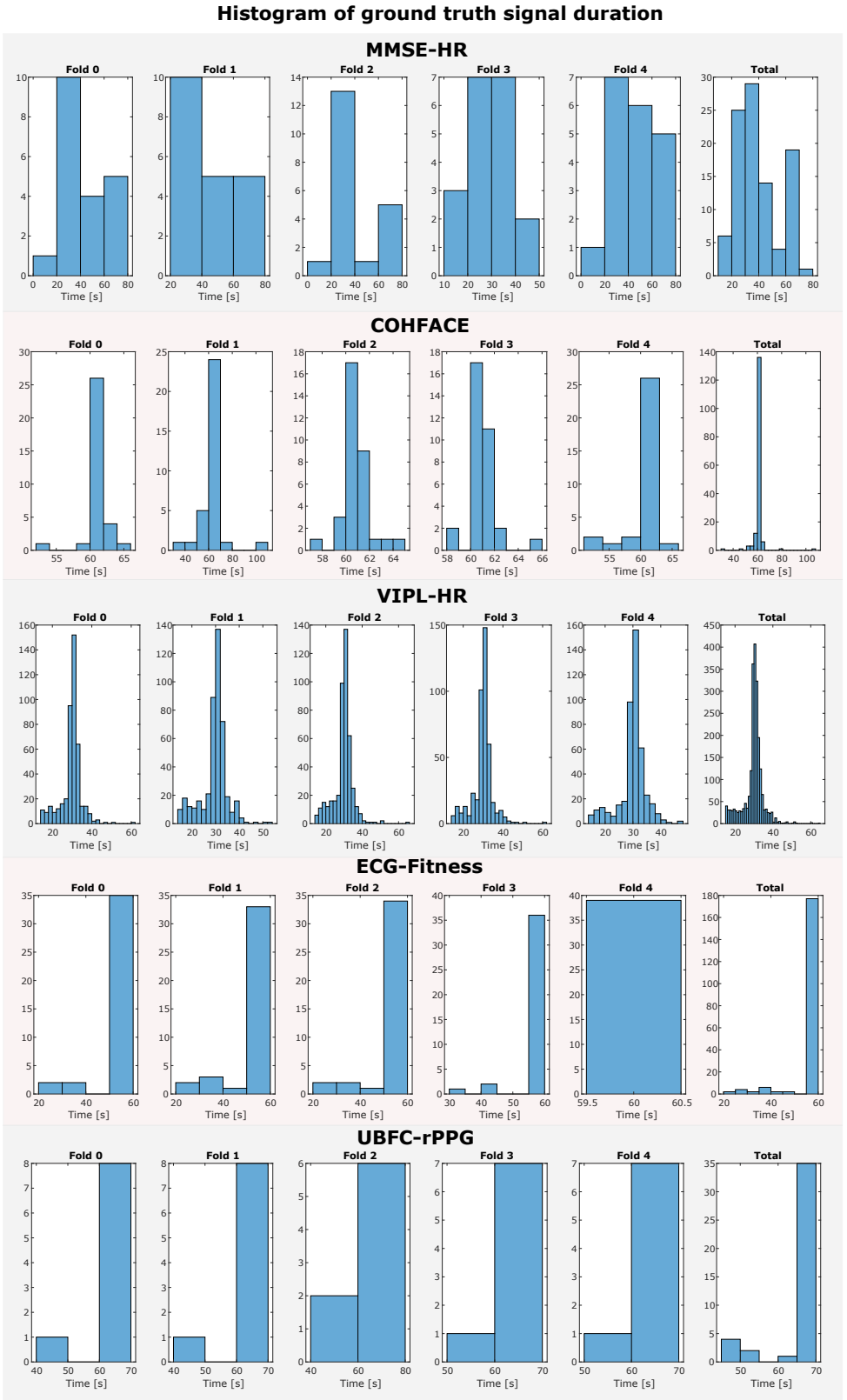

Figure A.1: Histograms with the duration of ground truth signals in each database for each fold after pre-processing.

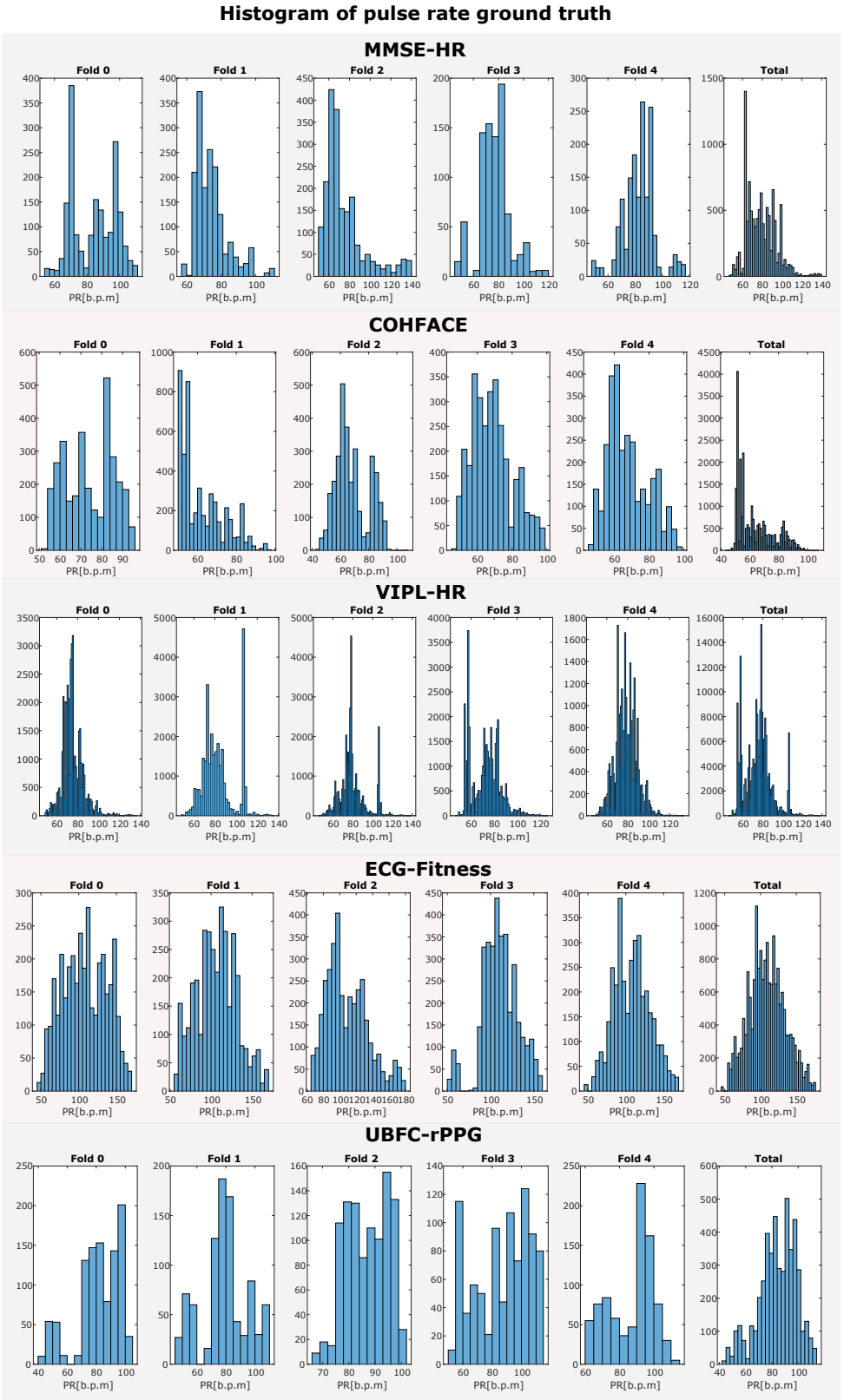

Figure A.2: Histograms with the pulse rate measured from the ground truth signals in each database for each fold after pre-processing.

A.3/ LSTMDF ARCHITECTURE: NUMBER OF UNITS AND LAYERS

This section presents the experiments performed to find the number of units and layers of the LSTMDF. These experiments were performed only on the MMSE-HR database, where the MSE loss function was used as the only metric. The following experiments were originally presented in the master thesis of Paul Pierrart [85] that took place in our research group. Eventually, it was published in [93].

Table A.1: Influence in number of LSTM layers

LSTM layers	1	2	3	4	5	6	7	8	9	10
MSE(E-03)	181	176	174	176	173	177	195	205	205	205

Table A.2: Influence in number of LSTM units

LSTM units	10	20	30	40	50	60	70	80	90	100
MSE(E-03)	188	184	181	178	182	180	178	177	183	178
LSTM units	110	120	130	140	150	160	170	180	190	200
MSE(E-03)	180	181	180	179	190	184	181	184	182	182
LSTM units	250	300	350	400	450	500				
MSE(E-03)	184	187	184	188	188	186				

A.4/ LSTMDF PULSE-RATE VARIABILITY METRICS

This section presents the results of the pulse-rate variability metrics of the experiments developed in Chapter 5. Specifically, in the protocols, *intra-dataset* first and second part (section 5.4.2.1), *cross-dataset* (section 5.4.2.2), *Amount of training data* (section 5.4.2.3), *RPPG-SNR dependence* first and second part (section 5.4.2.4).

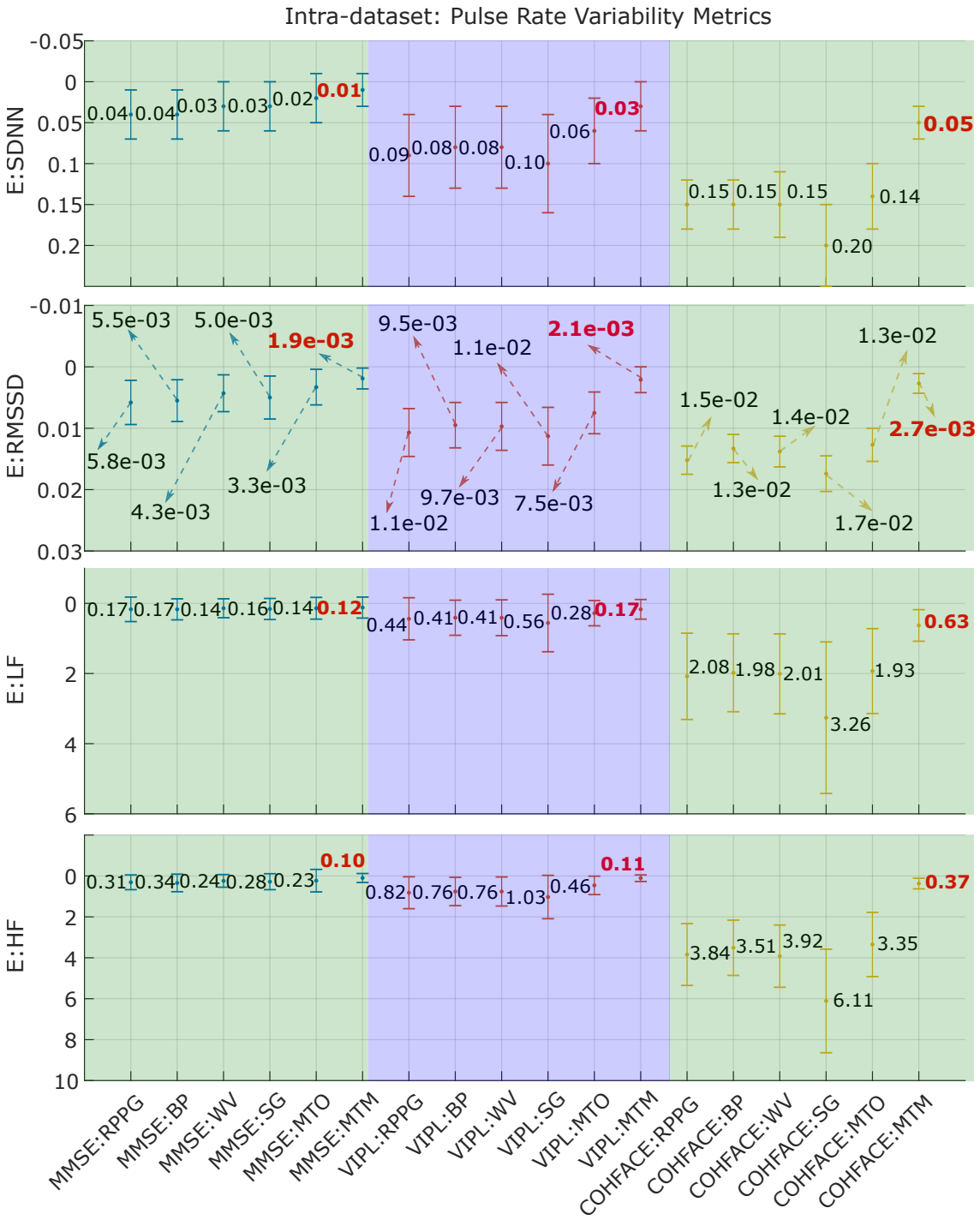

Figure A.3: Intra-dataset pulse-rate-variability metrics. Results of the filters: bandpass (BP), wavelet (WV), Savitzky-Golay (SG), LSTMDF MTO and MTM. Best values are presented in bold red.

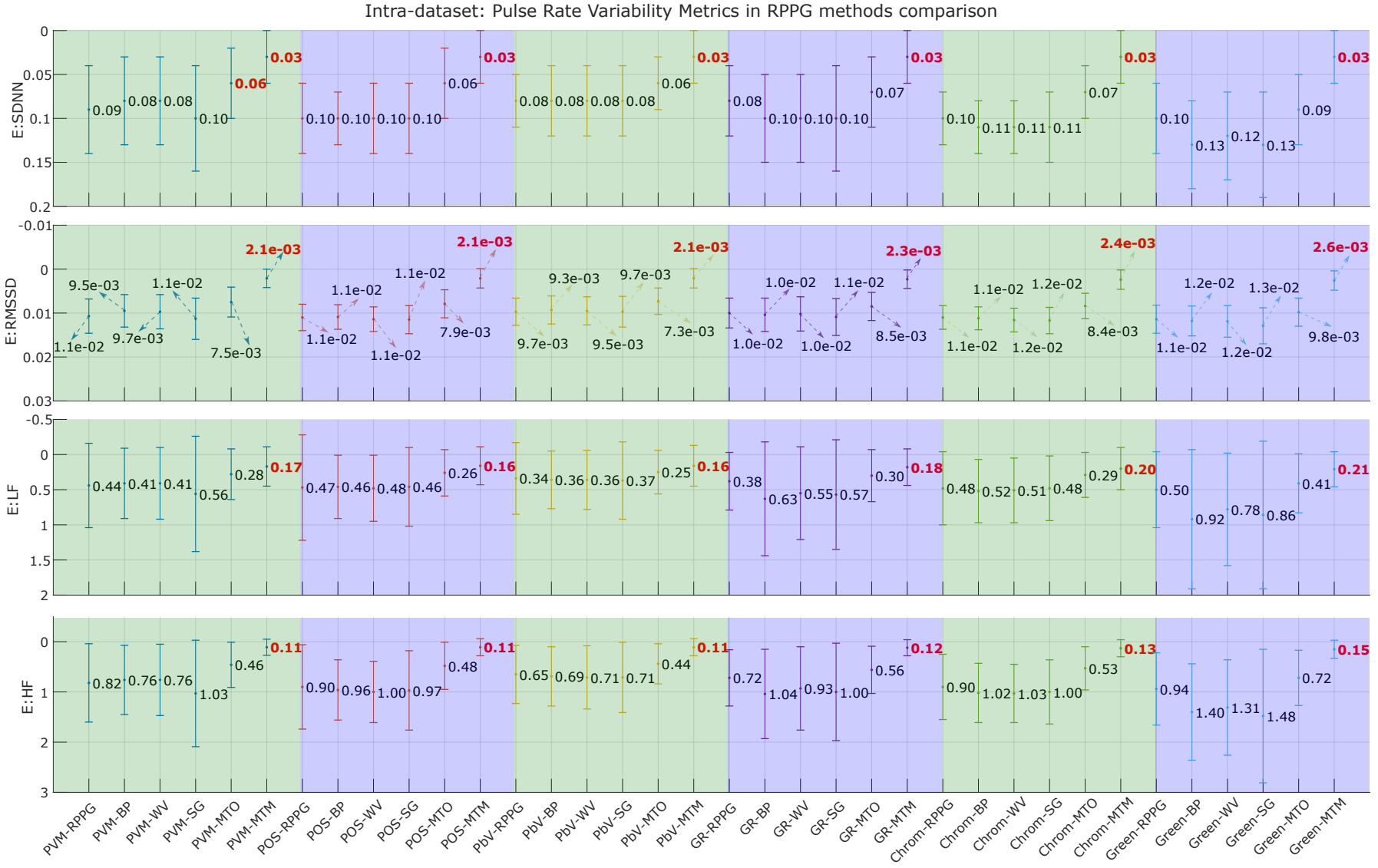

Figure A.4: RPPG methods comparison. Results of pulse-rate-variability metrics: E:SDNN, E:RMSSD, E:LF, and E:HF in the *intra-dataset* scenario.

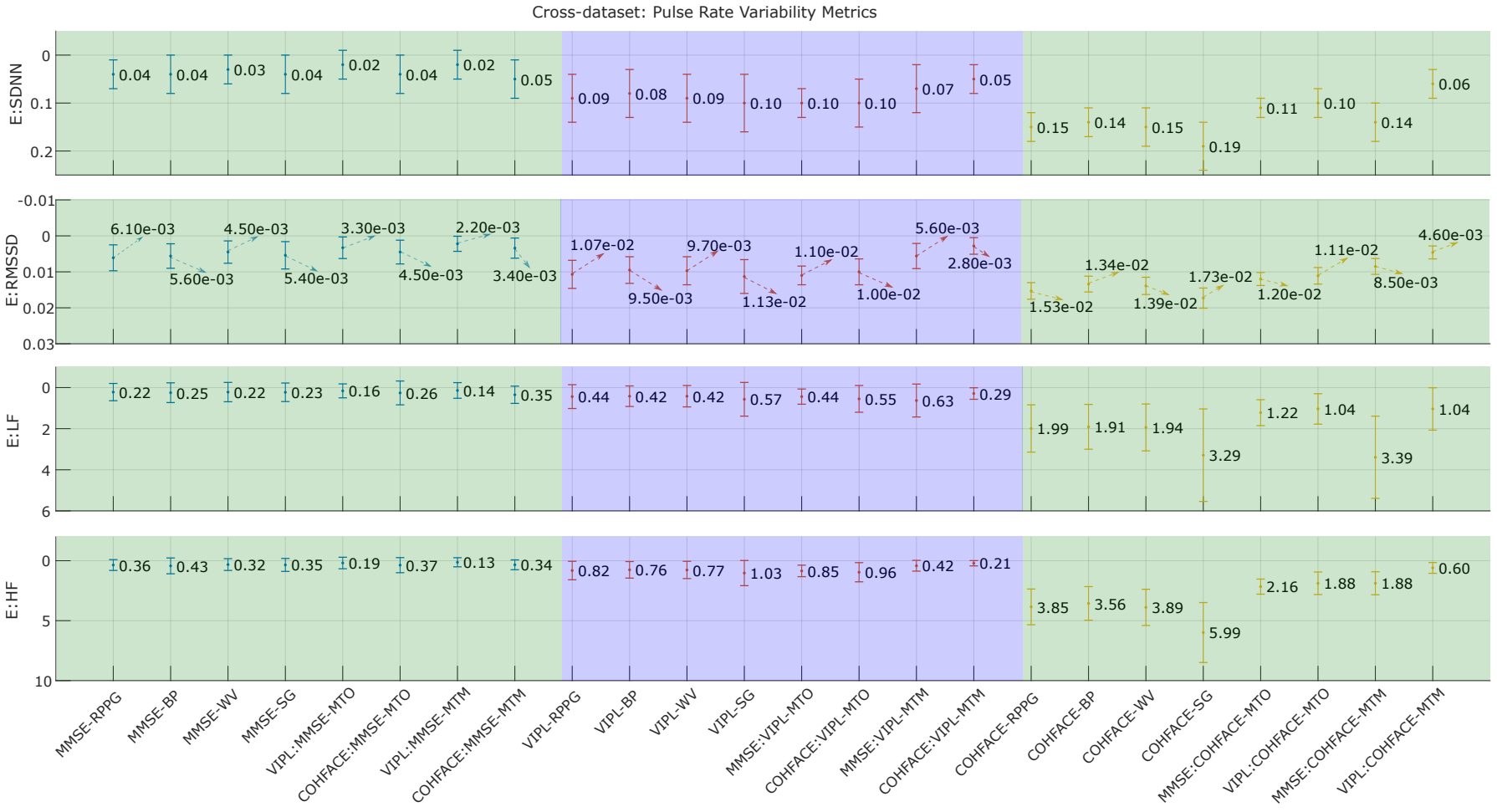

Figure A.5: Cross-dataset. Results of pulse-rate-variability metrics: E:SDNN, E:RMSSD, E:LF, and E:HF in the *cross-dataset* experiment. Experiments on the x-axis are listed as ”Training database: Test database - Filtering method” except for classical filters

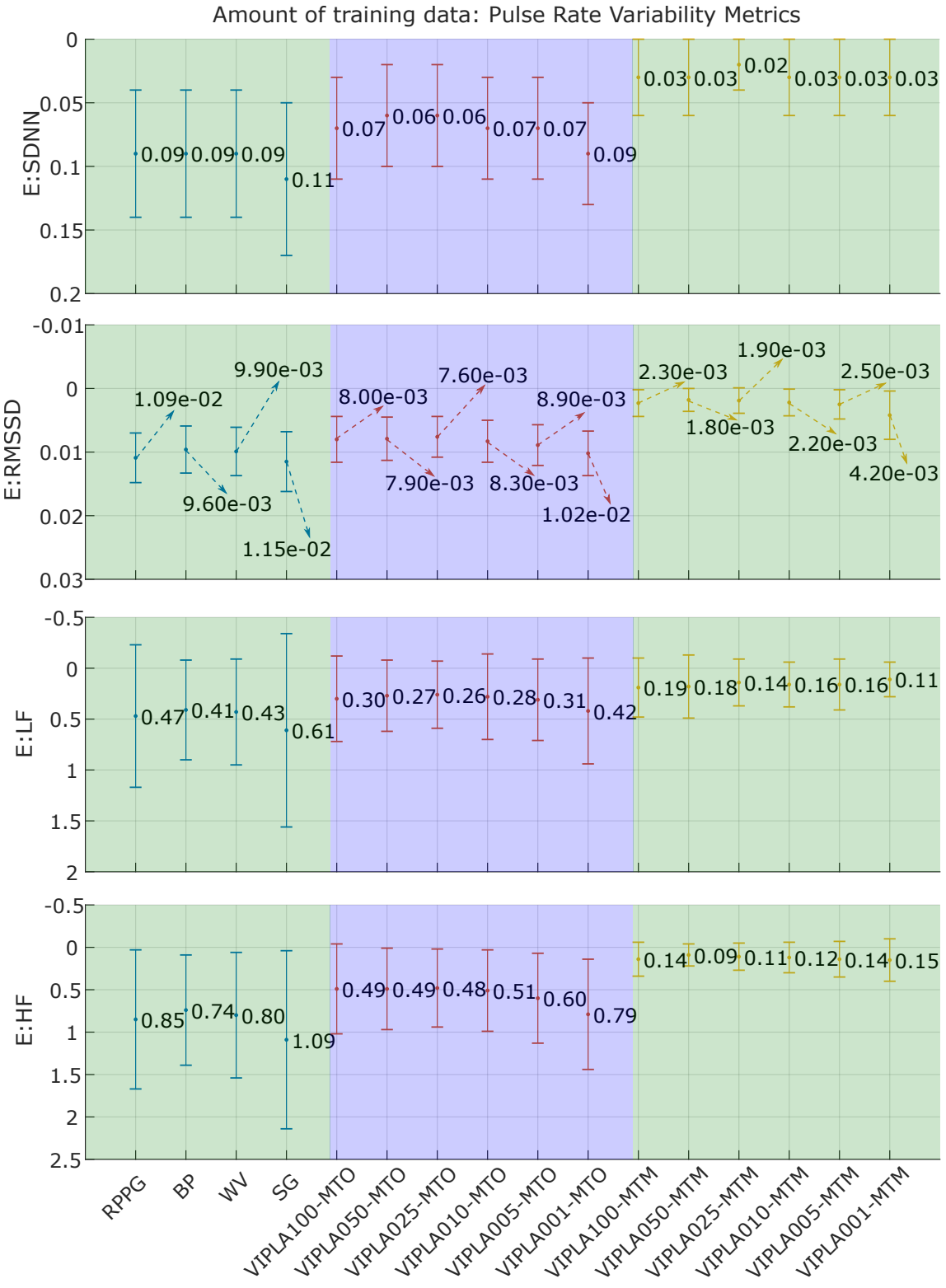

Figure A.6: Amount of training data. Results of pulse-rate-variability metrics: E:SDNN, E:RMSSD, E:LF, and E:HF in the *Amount of training data* experiment.

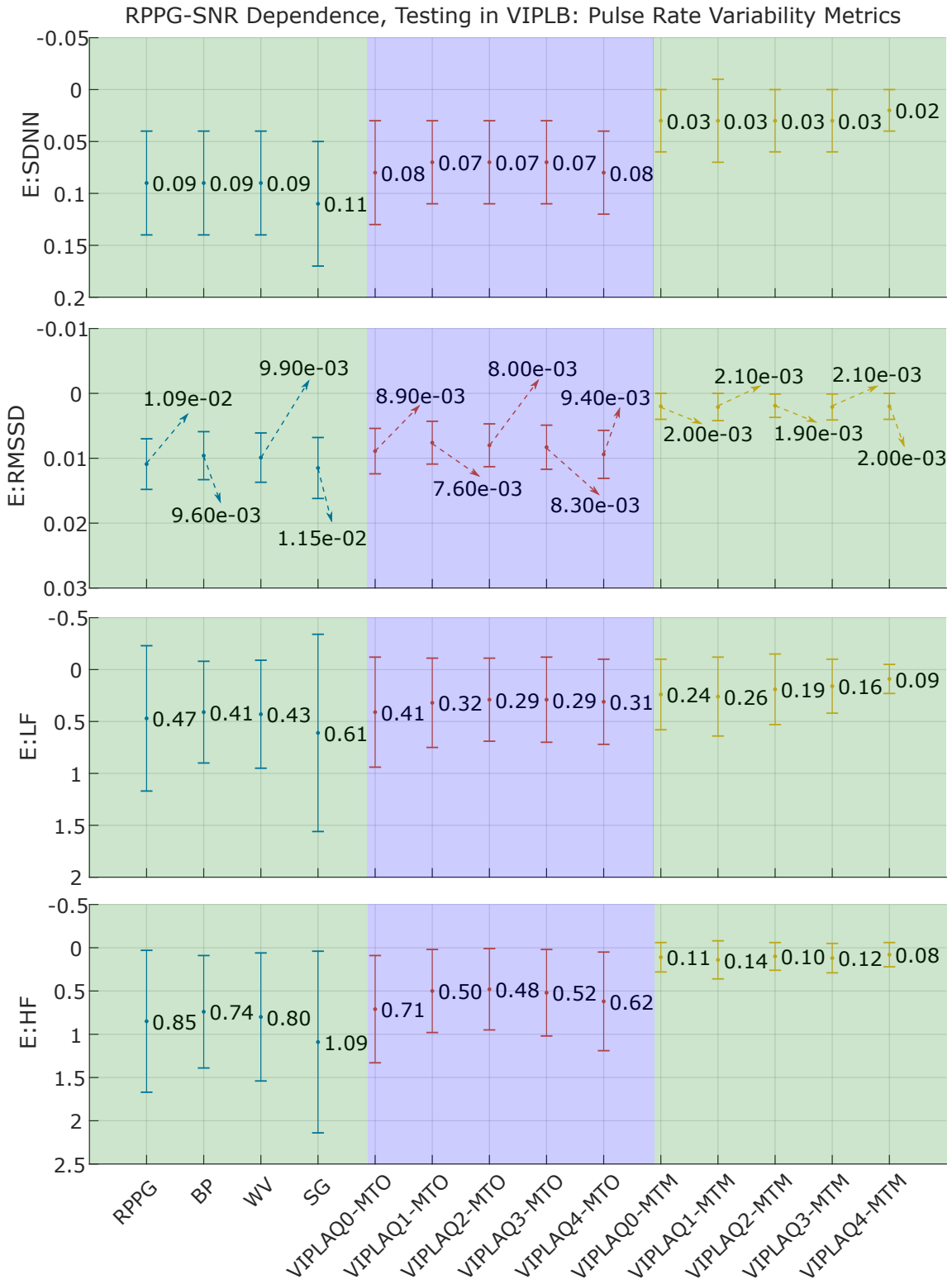

Figure A.7: RPPG-SNR dependence during training. Results of pulse-rate-variability metrics : E:SDNN, E:RMSSD, E:LF, and E:HF in the *RPPG-SNR dependence* experiment. Training in VIPLAQ0, VIPLAQ1, VIPLAQ2, VIPLAQ3, and VIPLAQ4, and testing in VIPLB.

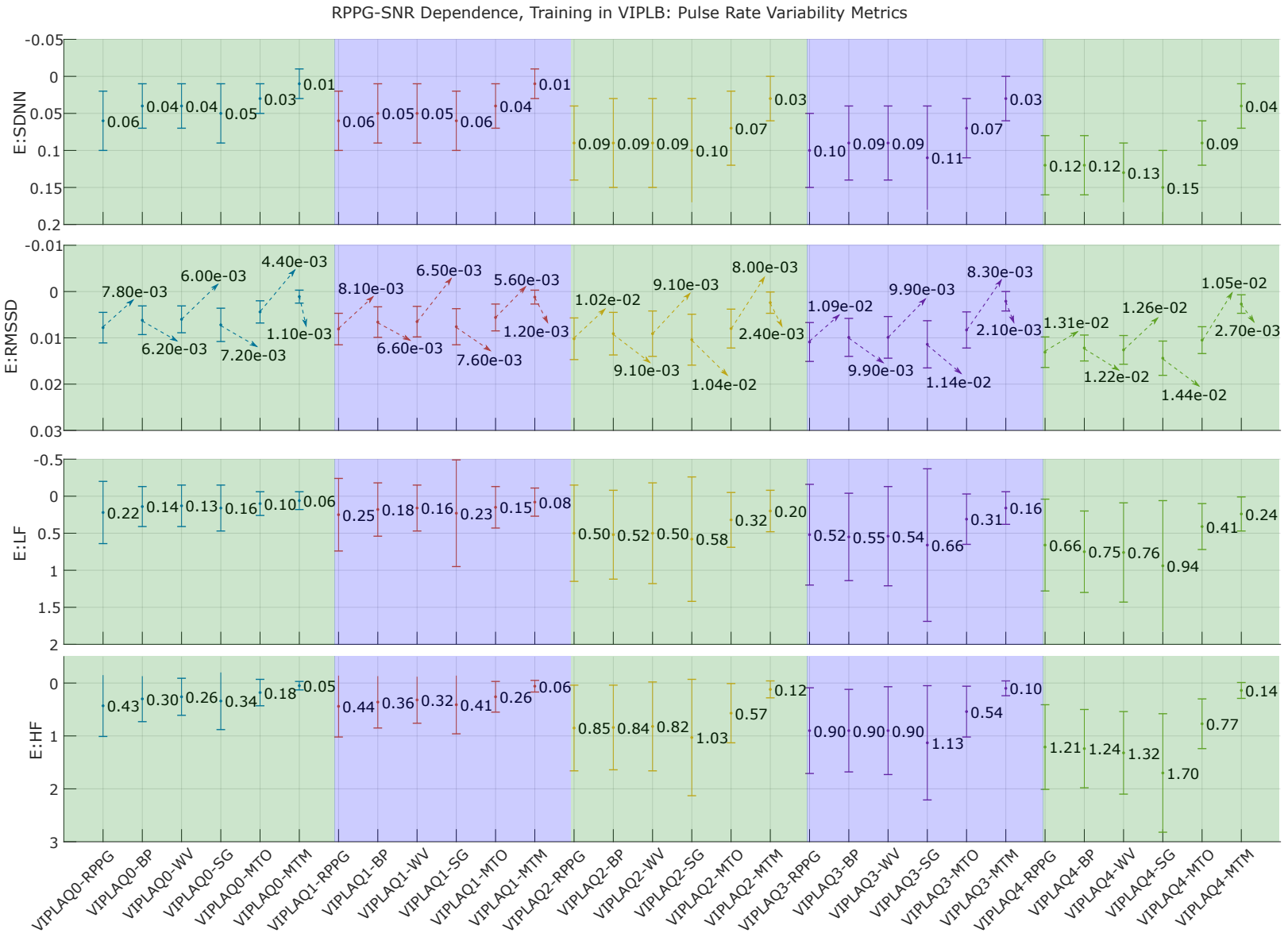

Figure A.8: RPPG-SNR dependence during testing. Results of pulse-rate-variability metrics: E:SDNN, E:RMSSD, E:LF, and E:HF in the *RPPG-SNR dependence* experiment. Training in VIPLB and testing in VIPLAQ0, VIPLAQ1, VIPLAQ2, VIPLAQ3, and VIPLAQ4.

A.5/ 3DCNN-BASED NETWORKS: TRAINING HISTORY LOSS

The training histories of the neural networks used in the chapter 6 are presented below.

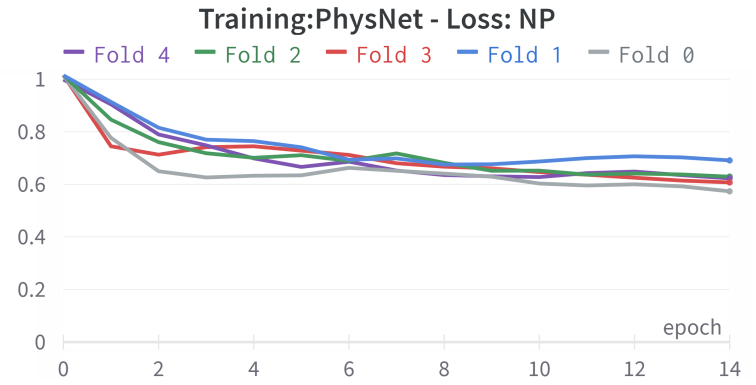

Figure A.9: PhysNet training history using the NP loss function. Using RGB channels.

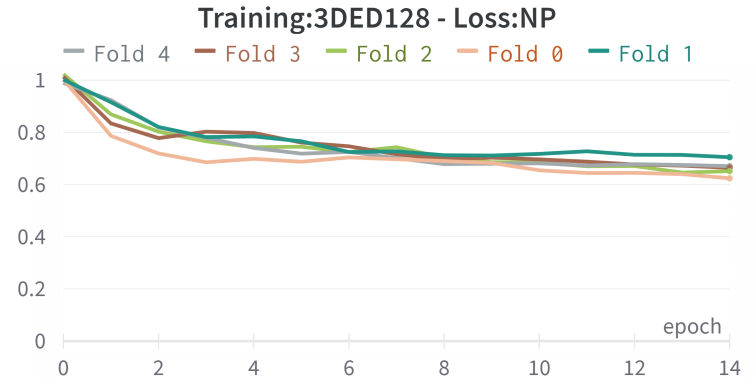

Figure A.10: 3DED128 training history using the NP loss function. Using RGB channels.

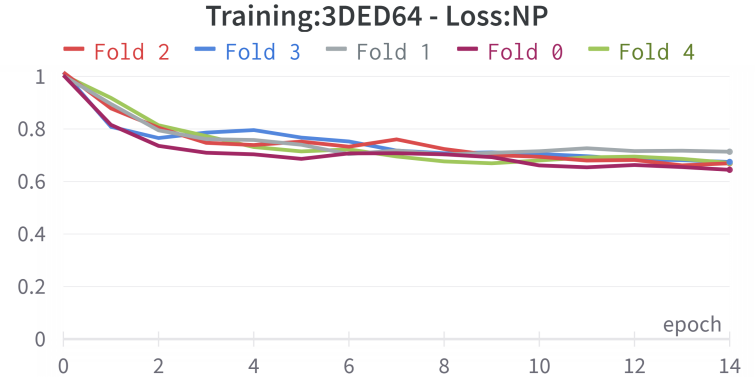

Figure A.11: 3DED64 training history using the NP loss function. Using RGB channels.

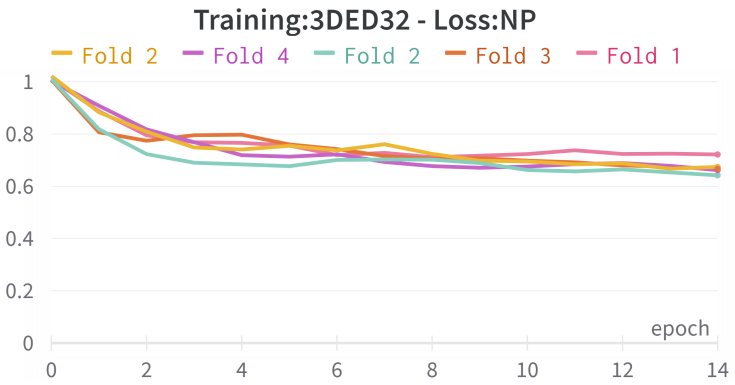

Figure A.12: 3DED32 training history using the NP loss function. Using RGB channels.

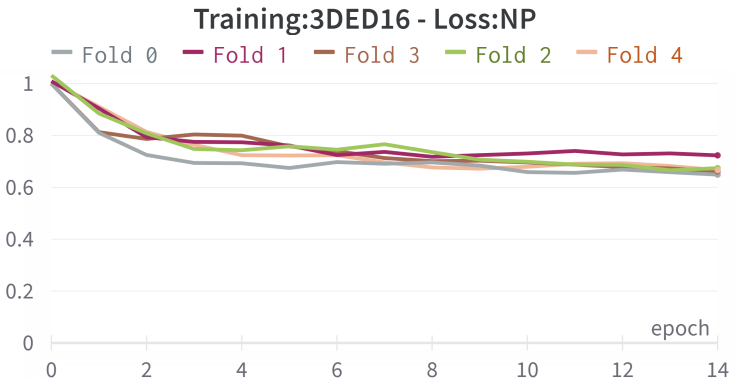

Figure A.13: 3DED16 training history using the NP loss function. Using RGB channels.

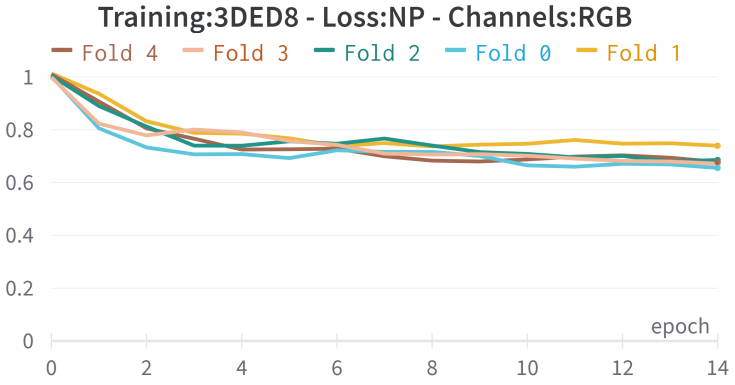

Figure A.14: 3DED8-RGB-NP training history using the NP loss function. Using RGB channels.

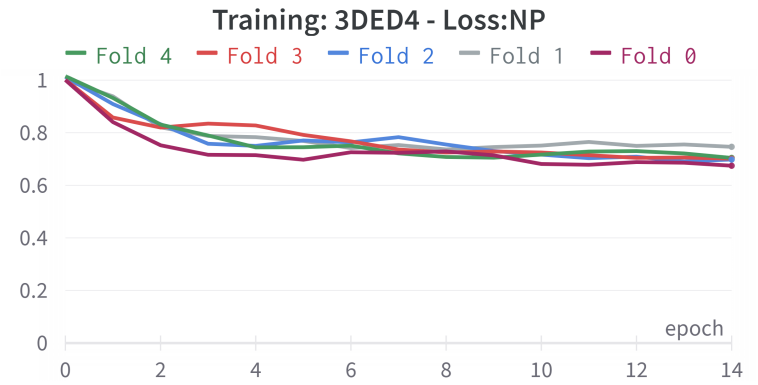

Figure A.15: 3DED4 training history using the NP loss function. Using RGB channels.

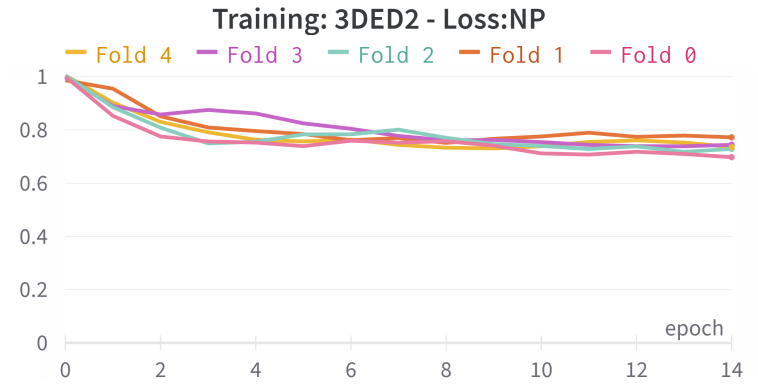

Figure A.16: 3DED2 training history using the NP loss function. Using RGB channels.

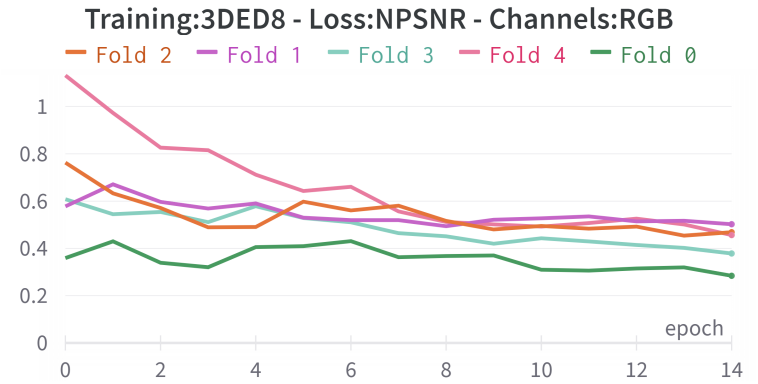

Figure A.17: 3DED8-RGB-NPSNR training history using the NPSNR loss function. Using RGB input channels.

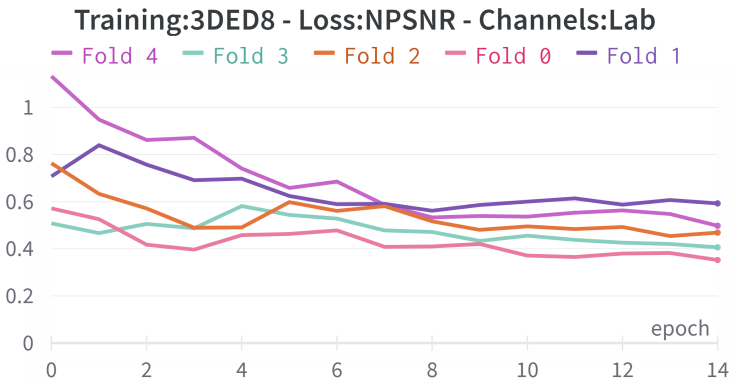

Figure A.18: 3DED8-Lab-NPSNR training history using the NPSNR loss function. Using Lab input channels.

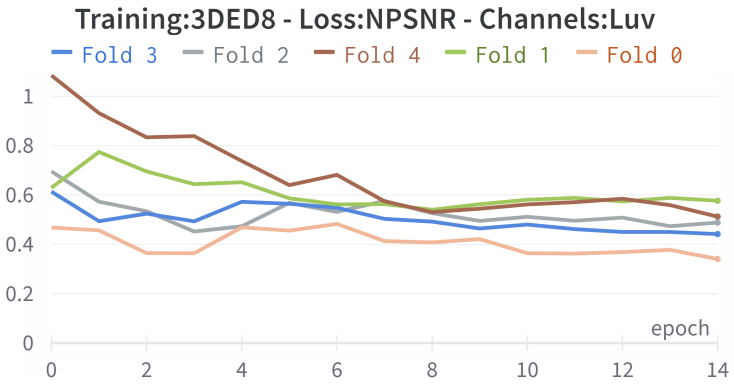

Figure A.19: 3DED8-Luv-NPSNR training history using the NPSNR loss function. Using Luv input channels.

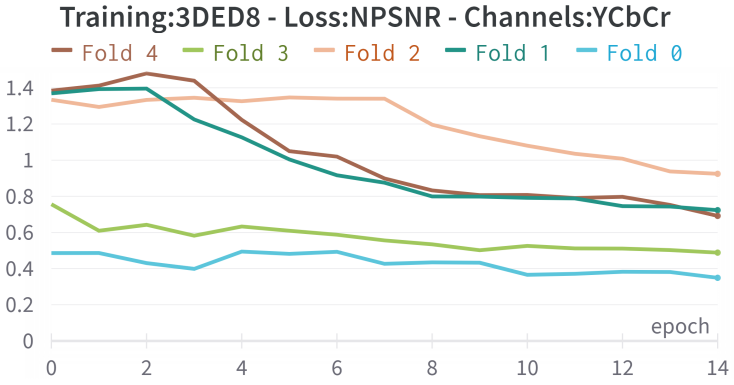

Figure A.20: 3DED8-YCbCr-NPSNR training history using the NPSNR loss function. Using YCbCr input channels.

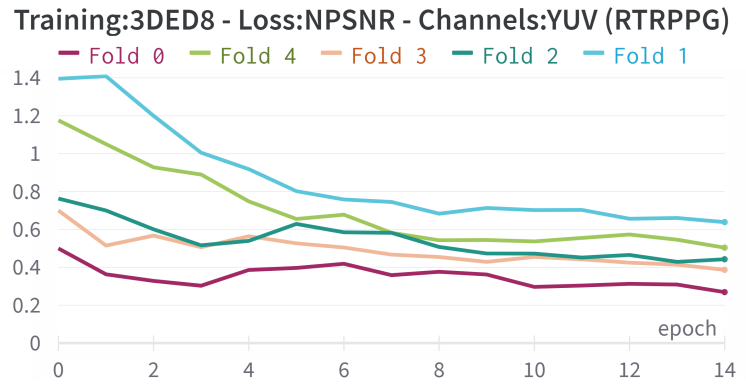

Figure A.21: 3DED8-YUV-NPSNR (RTRPPG) training history using the NPSNR loss function. Using YUV input channels.

A.6/ 3DED-BASED NETWORKS: ARCHITECTURES

The different architectures of 3DED-based networks used in chapter 6 are presented below.

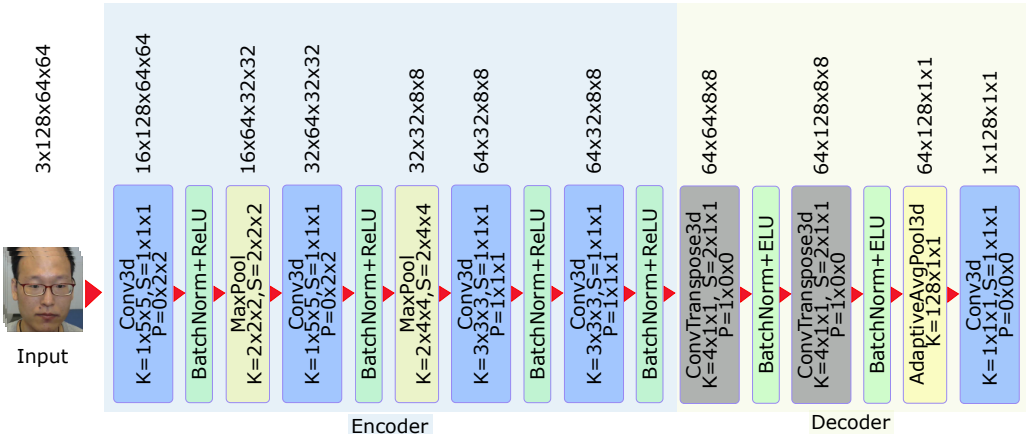

Figure A.22: 3DED64-RGB-NP Architecture. Inference time of GPU: 17.78 ms, CPU: 79.57 ms.

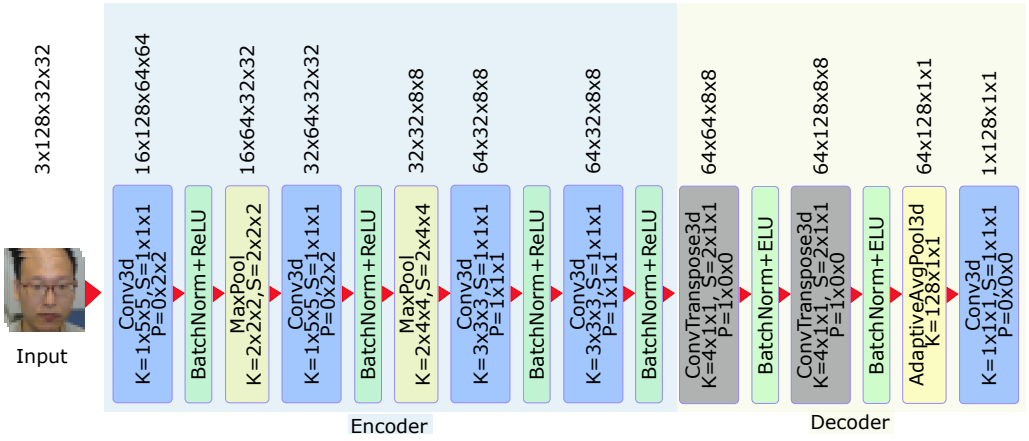

Figure A.23: 3DED32-RGB-NP Architecture. Inference time of GPU: 7.5 ms, CPU: 55.2 ms.

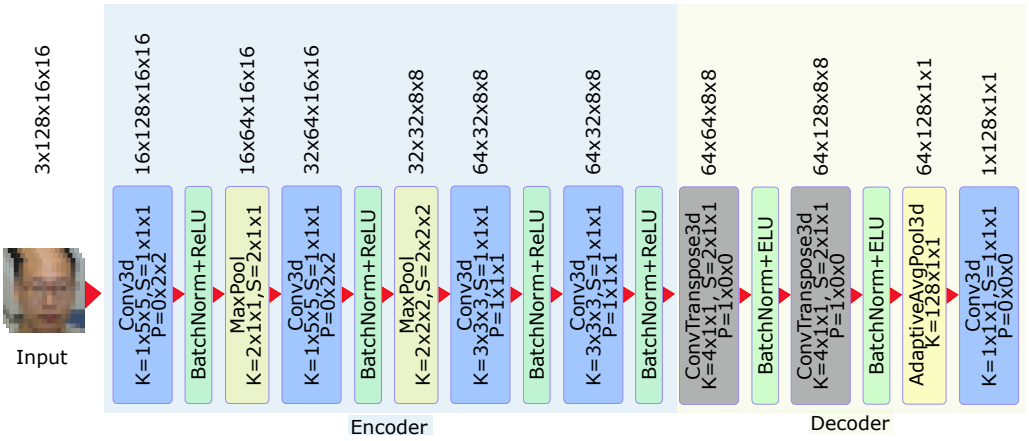

Figure A.24: 3DED16-RGB-NP Architecture. Inference time of GPU: 3.53 ms, CPU: 39.75 ms.

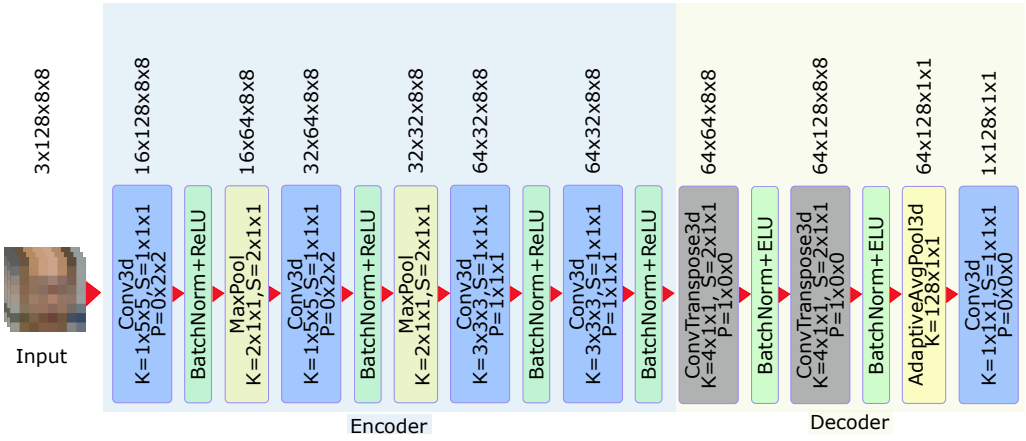

Figure A.25: 3DED8-RGB-NP Architecture. Inference time of GPU: 2.32 ms, CPU: 28.65 ms.

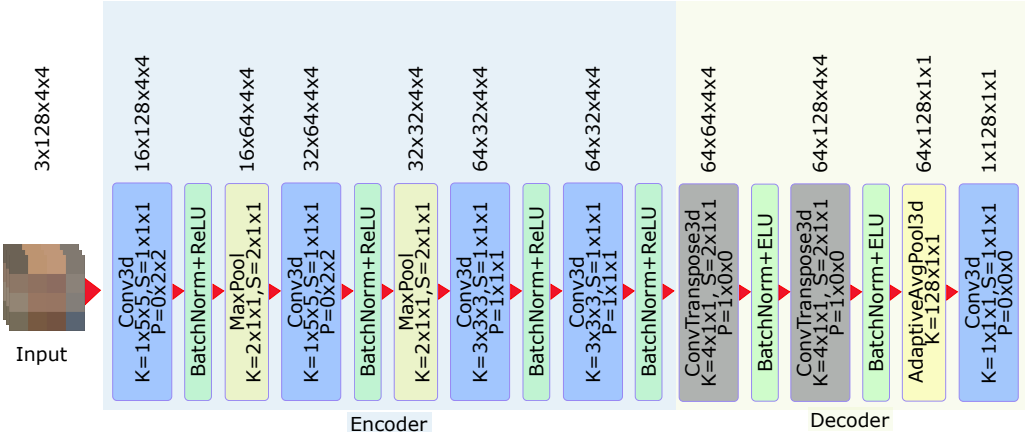

Figure A.26: 3DED4-RGB-NP Architecture. Inference time of GPU: 1.96 ms, CPU: 9.91 ms.

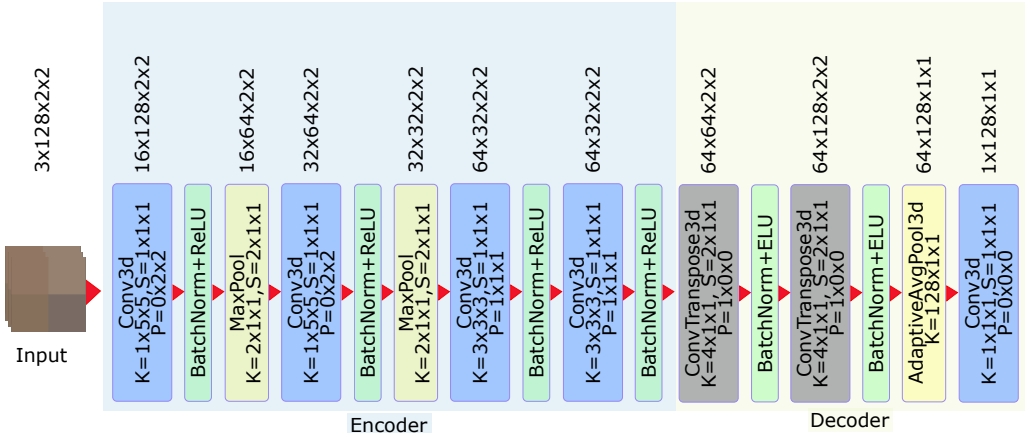

Figure A.27: 3DED2-RGB-NP Architecture. Inference time of GPU: 1.96 ms, CPU: 3.96 ms.

A.7/ 3DED-BASED NETWORKS: RESULTS

The results of the different 3DED-based network architectures are presented below.

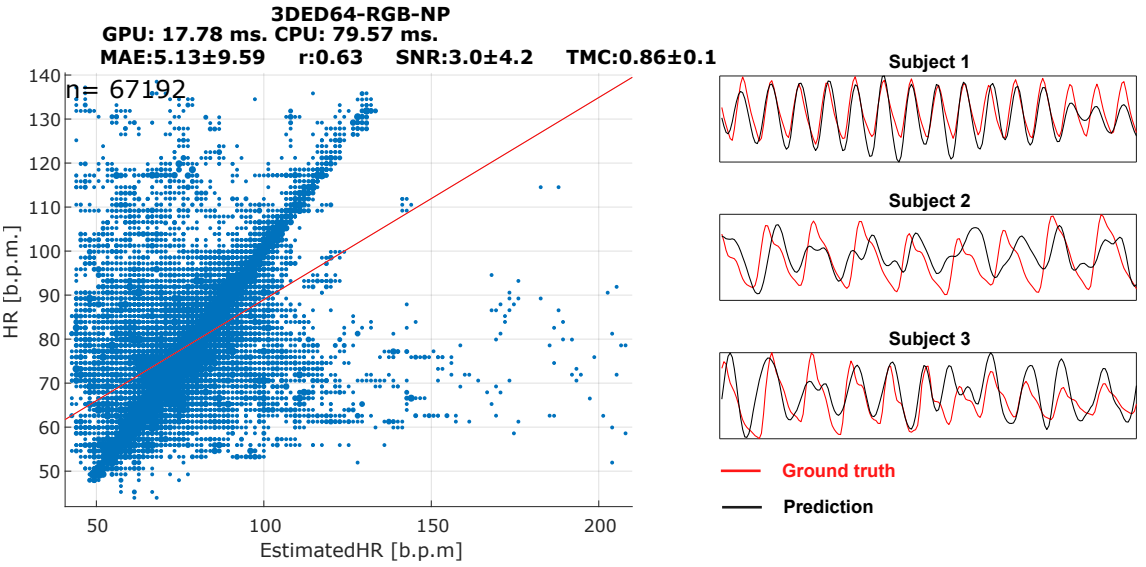

Figure A.28: 3DED64-RGB-NP results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

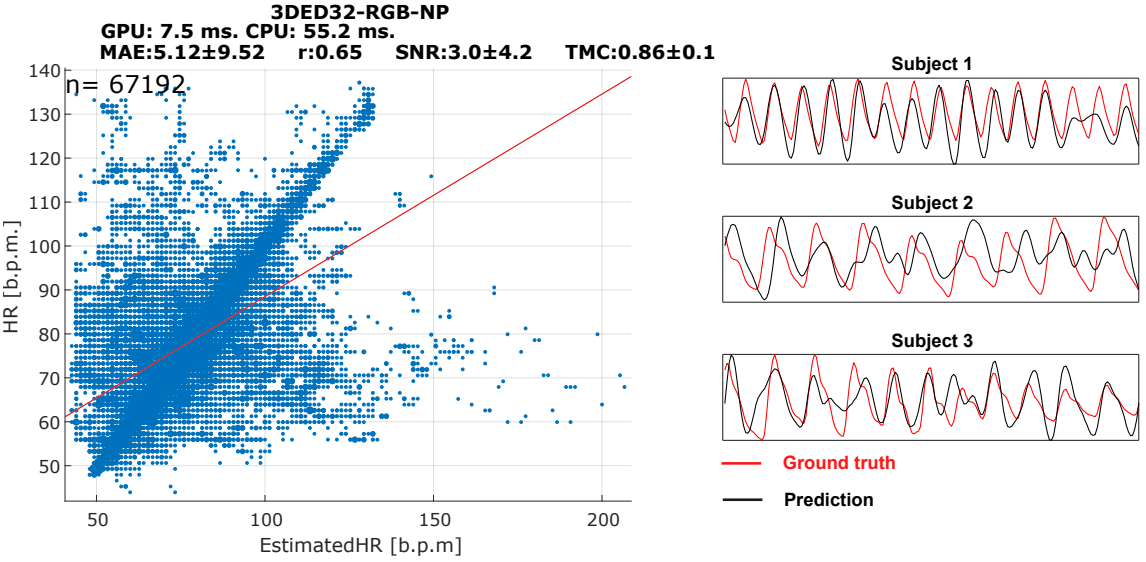

Figure A.29: 3DED32-RGB-NP results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

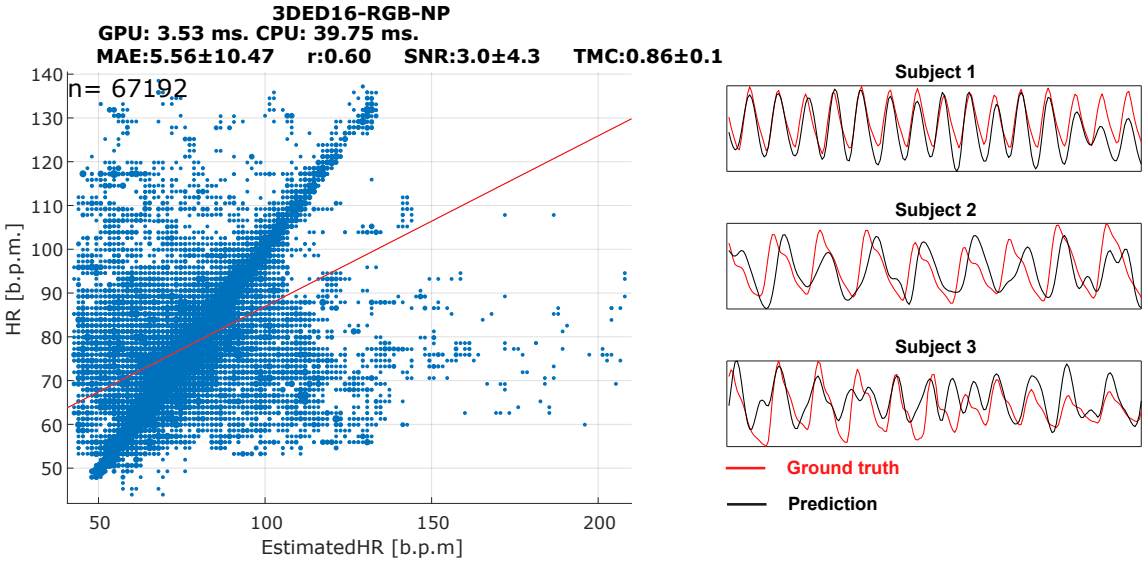

Figure A.30: 3DED16-RGB-NP Results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

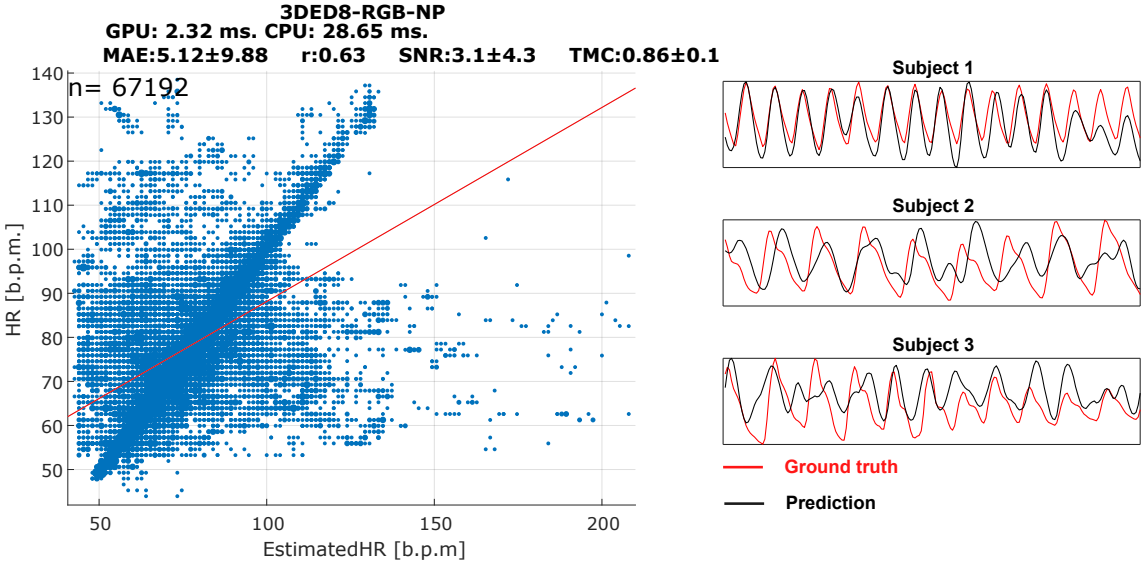

Figure A.31: 3DED8-RGB-NP Results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

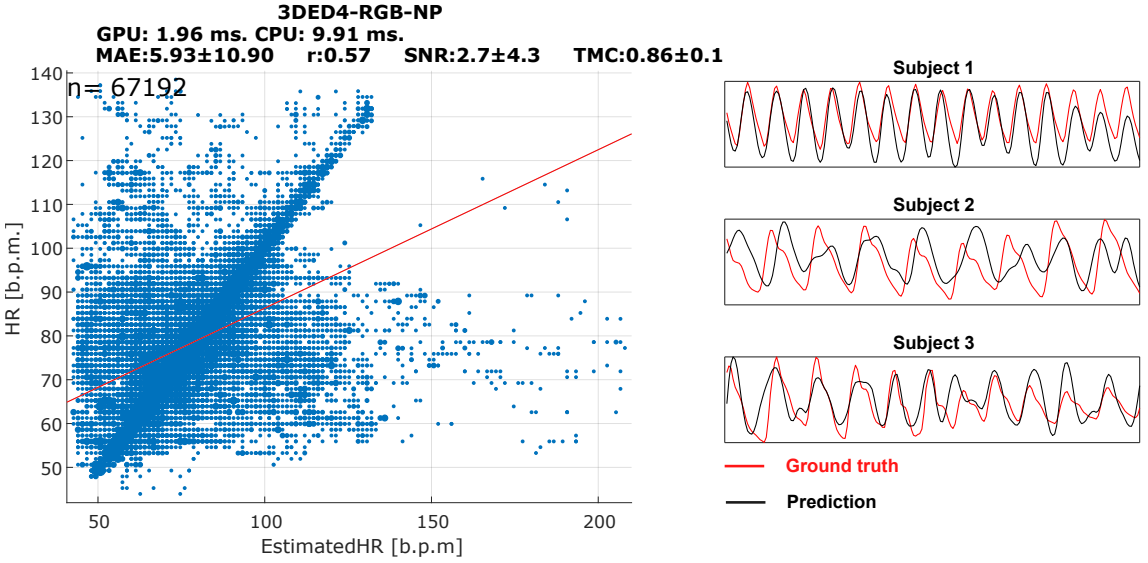

Figure A.32: 3DED4-RGB-NP Results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

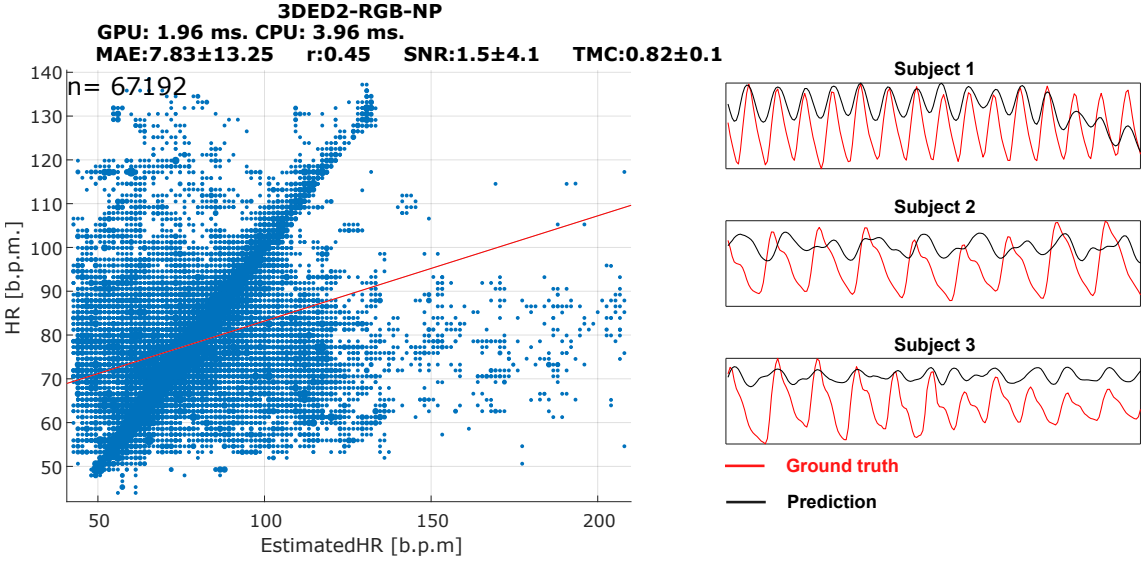

Figure A.33: 3DED2-RGB-NP Results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

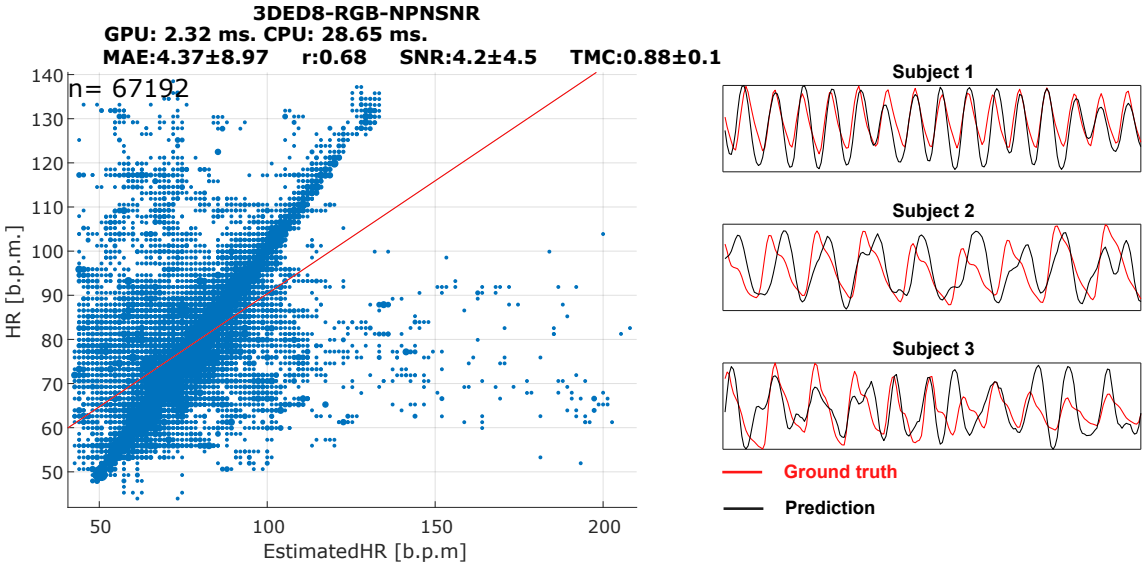

Figure A.34: 3DED8-RGB-NPSNR Results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

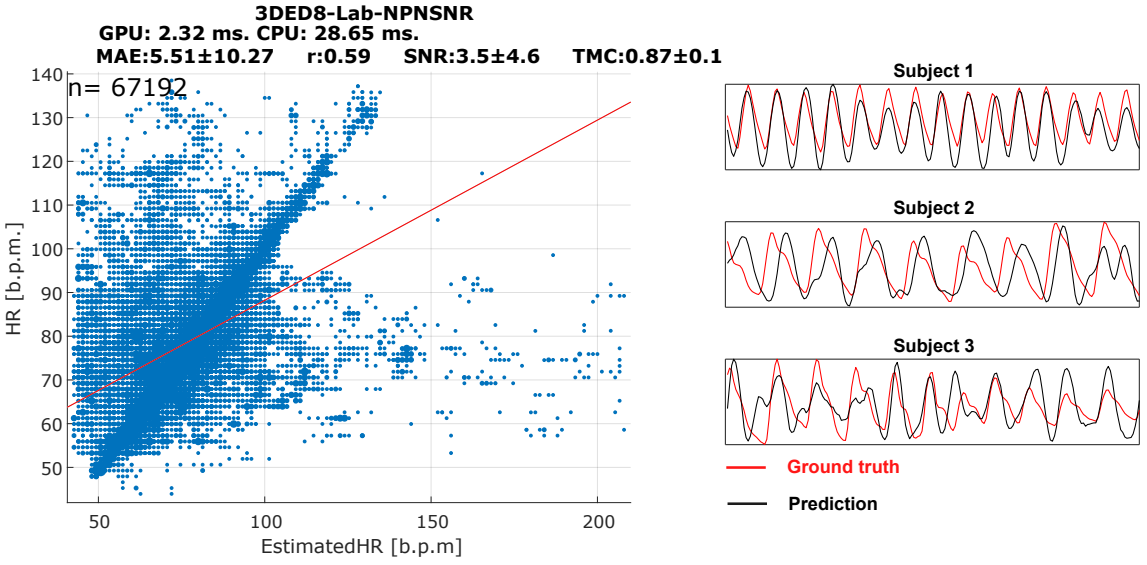

Figure A.35: 3DED8-Lab-NPSNR Results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

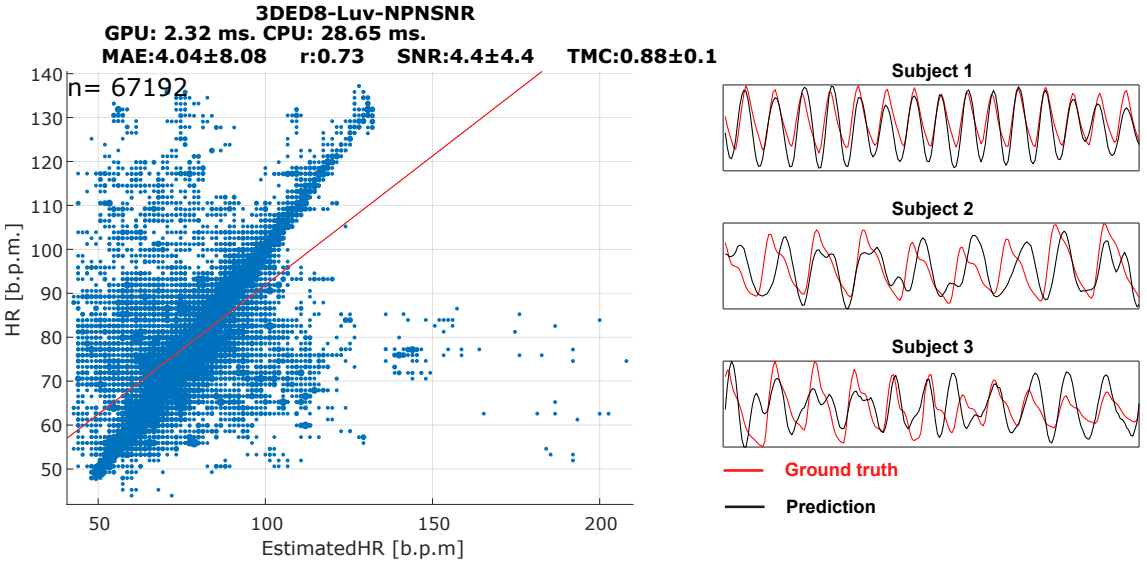

Figure A.36: 3DED8-Luv-NPSNR Results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

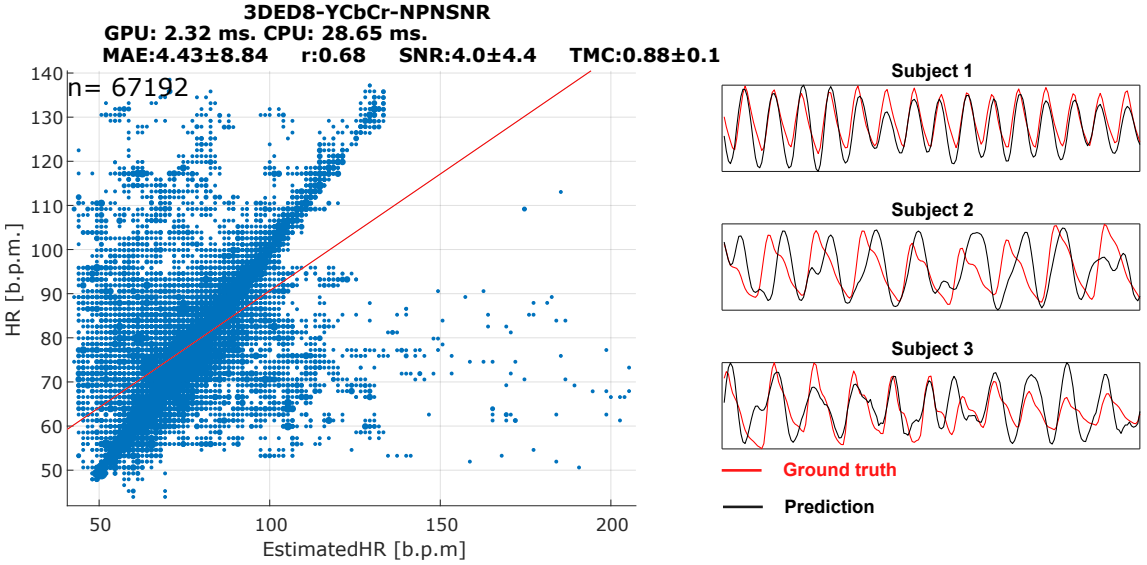

Figure A.37: 3DED8-YCbCr-NPSNR Results. On the left side, pulse-rate and signal-quality metrics are presented with the correlation plot, and on the right side, three different subject predictions.

Document generated with LaTeX and:
the LaTeX style for PhD Thesis created by S. Galland — http://www.multiagent.fr/ThesisStyle
the tex-upmethodology package suite — http://www.arakhne.org/tex-upmethodology/