Pázmány Péter Katolikus Egyetem

Információs Technológiai és Bionikai Kar

DIPLOMAMUNKA

Camera-Based Pulse Monitoring Using Deep Learning Tools

Kamera alapú pulzusfigyelés mélytanuló módszerek használatával

Terbe Dániel

Info-Bionika MSc.

2020

Témavezető: Dr. Zarándy Ákos

Master Thesis Declaration of Authenticity

I, undersigned Terbe Dániel, student of the Faculty of Information Technology and Bionics at the Pázmány Péter Catholic University, hereby certify that this thesis was written without any unauthorized help, solely by me and I used only the referenced sources. Every part, which is quoted exactly or in a paraphrased manner, is indicated clearly with a reference. I have not submitted this thesis anywhere else.

_____

Terbe Dániel

Abstract

Lately, it has been shown [1] that an average color camera can detect the subtle color variations of the skin (caused by the cardiac cycle) – enabling us to monitor the pulse remotely in a non-contact manner with a camera. Since then, the field of remote photoplethysmography (rPPG) has been formed and advanced quickly in order the overcome its main limitations, namely: motion robustness and low signal quality. Most recently, deep learning (DL) methods have also appeared in the field – but applied only to adults so far. In this work, we utilize DL approaches for long-term, continuous premature infant monitoring in the Neonatal Intensive Care Unit (NICU).

The technology used in NICU for monitoring vital signs of infants has hardly changed in the past 30 years (i.e., ECG and pulse-oximetry). Even though these technologies have been of great importance for the reliable measurement of essential vital signs (like heart-rate, respiration-rate, and blood oxygenation), they also have considerable disadvantages – originating from their contact nature. The skin of premature infants is fragile, and contact sensors may cause discomfort, stress, pain, and even injuries – thus can harm the early development of the neonate [2], [3]. For the well-being of not exclusively newborns, but also every patient or subject who requires long-term monitoring (e.g., elders) or for whom contact sensors are not applicable (e.g., burn patients), it would be beneficial to replace contact-based technologies with non-contact alternatives without significantly sacrificing accuracy. Therefore, the topic of this study is camera-based (remote) pulse monitoring – utilizing DL methods – in the specific use-case of infant monitoring in the NICU.

First of all, as there is no publicly available infant database for rPPG purposes currently to our knowledge, it had to be collected for Deep Neural Network (DNN) training and evaluation. Video data from infants were collected in the *I*st Dept. of Neonatology of Pediatrics, Dept. of Obstetrics and Gynecology, Semmelweis University, Budapest, Hungary and a database was created for DNN training and evaluation with a total length of around 1 day.

Two state-of-the-art DNNs were implemented (and trained on our data) which were developed specifically for the task of pulse extraction from video, namely DeepPhys [4] and PhysNet [5]. Besides, two classical algorithms were implemented, namely POS [6] and FVP [7], to be able to compare the two approaches: in our dataset DL methods outperform classical ones. A novel data augmentation technique is introduced for rPPG DNN training, namely frequency augmentation, which is essentially a temporal resampling of a video and corresponding label segment (while keeping the original camera sampling rate parameter unchanged) resulting in a modified pulse-rate. This method significantly improved the generalization capability of the DNNs.

In case of some external condition, the efficacy of remote sensing the vital signs are degraded (e.g., inadequate illumination, heavy subject motion, limited visible skin surface, etc.). In these situations, the prediction of the methods might be inaccurate or might give a completely wrong estimate blindly without warning – which is undesirable, especially in medical applications. To solve this problem, the technique of Stochastic Neural Networks (SNNs) is proposed which yields a probability distribution over the whole output space instead of a single point estimate. In other words, SNNs associate a certainty/confidence/quality measure to their prediction, therefore we know how reliable an estimate is. In the spirit of this, a probabilistic neural network was designed for pulse-rate estimation, called RateProbEst, fused and trained together with PhysNet. This method has not been applied in this field before to our knowledge. Each method was evaluated and compared with each other on a large benchmark dataset.

Finally, the feasibility of rPPG DNN applications in a resource-limited environment is inspected on an NVIDIA Jetson Nano embedded system. The results demonstrate that the implemented DNNs are capable of (quasi) real-time inference even on limited hardware.

Contents

Chapter 1

Introduction

Photoplethysmography (PPG) is a non-invasive, optical technique that measures the blood volume variations in the living skin. The underlying phenomenon is that at every heartbeat the heart pumps blood into the vasculature (situated in the superficial layers of the skin), therefore its volume increases, which in turn results in more light absorption. In consequence, the absorption and scattering of light change synchronously with the cardiac-cycle which can be detected with a photodetector. Utilizing this, different physiological variables can be derived such as pulse-rate, respiration-rate[1] or pulse-rate variability. In addition, with the analysis of the measured PPG-waveform different cardiovascular conditions can be obtained, for example, arterial diseases like arterial stiffness and aging. Furthermore, a fortunate optical property of the blood enables us to measure its oxygen-saturation (SpO2) level, that is the oxygenated and deoxygenated blood absorbs light differently at different wavelengths. The widely used pulse-oximeter devices are based on these principles.

An average pulse-oximeter consists of two light-emitting diodes (LED) and a photodetector, those are in contact with the skin. The ideal wavelengths for SpO2 measurement are in the visible red (650-750 nm) and in the near-infrared (NIR) (850-1000 nm) range – as these wavelengths have the largest contrast regarding the blood in oxygenated or deoxygenated state. In those commercial wristbands which measure only the pulse-rate, green light (510-560 nm) is applied for illumination because this yields the largest PPG-signal. There are two possible setups for such contact devices: they can be either in transmissive or in reflective mode. In the former one, the LED and the photodetector are on the opposite side of the skin (attached for example to the finger or to the earlobe) and the photodetector measures the transmitted light through the skin. In contrast in

[1] Light absorption is also modulated by respiration.

reflective mode, the light source and the detector is on the same side of the skin (for example, attached to the wrist or to the fingertip) and the detector records the reflected light from the skin.

Another common technique with which we can detect cardiac events is electrocardiography (ECG) which measures the electrical activity of the heart – and is considered to be the ground-truth for heart-rate measurement. Although they correlate most of the time (and often used interchangeably), in this study we will distinguish between the ECG derived heart-rate (HR) and the PPG derived pulse-rate (PR) for consistency and clarity. HR stands for the rate of the electrical activity of the heart and PR stands for the rate of blood volume changes in the vascular system. The distinction is important, for example in case of certain heart problems when the heart muscles cannot effectively push enough blood through the cardiac-system at each contraction, thus the pulse-rate will be less than the heart-rate – or the PR might be even totally absent while the HR shows normal activity. Despite the fact that it is widely used, cheap and reliable, the problem with ECG devices is that they are even more obtrusive than PPG devices: they require adhesive electrodes to be attached to the chest.

These technologies have hardly changed in the past 30 years, since the advent of these methods because both ECG and PPG are generally reliable and inexpensive techniques. However, these devices have also considerable disadvantages originating mainly from their skin-contact nature. The wires do not allow free movements which can be uncomfortable during long-term continuous monitoring (e.g. sleep monitoring, Intensive Care Unit (ICU)). Furthermore, the sensor is in contact with the skin which may be even damaging if the patient has fragile skin – especially in the case of newborns in the Neonatal Intensive Care Unit (NICU) or burn patients. Therefore in many situations, it would be beneficial to replace contact-based technologies with non-contact alternatives.

There are five fundamentally different approaches that enable remote monitoring of cardiac activity without skin contact: ballistocardiography (BCG) which is based on the measurement of skin or body displacement caused by heartbeats; capacitive electrocardiography which is the unobtrusive alternative of ECG; infrared thermography which utilizes the thermal changes of the body; electromagnetic induction based monitoring which measures the biological impedance caused by thoracic volume variation; and finally, remote-photoplethysmography (rPPG) which is the remote, non-contact counterpart of PPG. Most of these methods are not practical because they are expensive and/or bulky except rPPG, which can operate with an average RGB-camera (e.g. webcam, mobile

camera) – therefore this is the most active and quickly advancing field. It also serves as the topic of this thesis.

Compared to the pulse-oximeter – which is the most common contact-PPG device – the rPPG setup differs in building blocks and in the geometric arrangement of these elements but fundamentally relies on the same principles: in the rPPG system the LED is replaced with an ambient or dedicated widefield lightsource (e.g. sunlight, fluorescent lamp, incandescent lamp, halogen lamp, etc.) and the detector is replaced with an "array of photodetectors" aka camera – hence the title of this thesis "Camera-based pulse monitoring...". These elements are placed on the same side of the skin, but of course away from it, thus it is non-contact. This falls into the previously mentioned reflective category because the camera detects the reflected light from the skin, but in this case, it is illuminated and monitored remotely (in contrast with conventional PPG devices). Although this measurement is very similar to contact-PPG, for the sake of consistency we differentiate between their outputs: the contact device results in the PPG-signal, and the remote setup yields the rPPG-signal. The main difference between them is the origin of the detected photons: the contact device collects light that has traveled through relatively deep vasculature (arterioles), while the remote camera records the light that has traveled through much shallower tissue depths (mainly capillaries). Finally, the Signal-to-Noise Ratio (SNR) of the rPPG-signal is much smaller and also more sensitive to environmental noise (e.g. body motion, illumination variation), therefore it is more challenging to design such algorithms that can cope with these factors. Nonetheless, it may be possible to achieve a similar performance with an rPPG setup.

For such a contactless system – besides clinical healthcare applications which were already mentioned – there are many areas where it could be employed. For example, it could be integrated into already existing home healthcare monitoring systems designed to monitor elderly or infants with cameras. The ability to estimate the pulse-rate would be a useful, important feature of the system. Remote sleep monitoring is also feasible with an infra-camera [8] mounted above the bed to analyze sleep quality. Another potential application would be a vehicle-driver monitoring system which could recognize tiredness (so could alert the driver not to fall asleep) or predict heart-attack and could warn her/him to pull off the road, thus preventing possible traffic accidents.

In this study, we focus on the specific application case of neonate monitoring in the NICU and apply deep learning tools in order the attain rPPG-signal and/or pulse-rate from video recordings.

1.1. Problem Statement

Monitoring of vital signs is an essential part of standard care for neonates in the NICU, where pulse-rate (PR) is one of the most critical parameters (besides respiration-rate and blood oxygenation). Current non-invasive methods require sensors connected to the skin surface. For example with ECG, adhesive electrodes are attached to the chest and PPG also requires contact. Despite the fact that these technologies have been of great importance for the reliable measurement of essential vital signs, they also have considerable disadvantages – originating from their contact nature. The skin of premature infants is really fragile and sensitive, therefore direct contact should be minimized. For example, allergic reactions can occur that can lead to severe damage to the skin of newborns. In addition, the ability to move freely is impaired in long-term monitoring. There are also cases when doctors choose not to monitor at all when the neonates are extremely premature and a contact device would cause permanent damage.

In summary, contact methods can cause stress, pain, and discomfort which in turn can also have a negative impact on cognitive development [2] and parent-child bonding [3]. For the well-being of not exclusively newborns but also every patient or subject who requires long-term monitoring, it would be beneficial to replace contact-based technologies with non-contact alternatives without significantly sacrificing accuracy.

In recent years, the field of rPPG has emerged and advanced quickly to address this problem. The involved research community is ever-increasing and proposed several rPPG algorithms [6] which aim to catch up with the traditional contact-based systems regarding performance and reliability, so far unsuccessfully, but with promising results. The main hindrances are the orders of magnitudes lower SNR compared to contact-PPG and the sensitivity for environmental artifacts like body motion and illumination variation. Many methods have been proposed to surpass these limitations, most of them are intricately composed of many distinct (but interconnected) and sometimes computationally expensive algorithmic components [8] (e.g. Region of Interest (RoI) detection, RoI tracking, signal extraction, and combination, etc.) which is undesirable for a real-time application.

In the case of many other computer vision tasks, deep learning (DL) based solutions already exceeded the classical counterparts. Similarly, their application in the field of rPPG could also bring surprises. Just recently, different neural network architectures have been proposed for this task [4], [5], [9] with promising results. The main advantage of a deep neural network (DNN) is that it can learn sophisticated nonlinear relationships.

In addition, it encapsulates each processing step within itself and functions in an elegant end-to-end fashion (which is software-technically much simpler and also more stable). Common arguments against DL are that DNNs are resource-intensive, they require a large amount of labeled training data and they function as black-boxes – their intrinsic mechanism is unknown. Although they are computationally expensive, their internal operations are highly optimized by modern frameworks (e.g. PyTorch, Tensorflow) – it would be important to investigate their inference speed on average hardware. On the other hand, there are also techniques with which the inner properties of a DNN can be inspected or constrained/guided.

Therefore, the main motivation of this thesis is to contribute to the development of deep learning based rPPG systems, particularly for NICU application. The objectives and contributions of this work are presented below.

1.2. Objectives

- The creation of a neonatal database for neural network training and validation.
- The implementation, application, and comparison of existing rPPG neural networks (and algorithms) on the created dataset.
- The proposal of a method with which the reliability of the estimate can be defined.
- The examination of applicability in a resource-limited environment.

1.3. Contributions

The main contributions of this thesis are listed below:

1. The construction of training and benchmark dataset for neonate monitoring in the NICU.
2. The implementation of existing rPPG DNNs (DeepPhys [4] and PhysNet [5]), their evaluation and comparison on our dataset.
3. The proposal of a new kind of augmentation technique, the frequency augmentation for rPPG DNN training.
4. The comparison of DL approach with other algorithmic methods (FVP [7] and POS [6]) on our data.

5. The proposal of Probabilistic Neural Networks as a solution to the problem of estimation reliability. Based on this idea the development of a probabilistic rate estimator network (RateProbEst).

6. The measurement of inference speed of PhysNet, DeepPhys, and RateProbEst on *NVIDIA Jetson Nano* embedded system. In other words, the investigation of the applicability of rPPG DNNs in real-time cost-efficient applications.

1.4. Thesis Structure

The structure of the thesis is presented here.

Chapter 2 In this chapter, the study is put into context by discussing the background. First, a general introduction to the field of rPPG is presented focusing on (1) the fundamental physical phenomenon (light-tissue interaction), (2) the properties of video-camera observation, and (3) the signal processing approaches. After that, the related work is discussed. Finally, a detailed mathematical formulation of the problem follows: (i) a linear and a non-linear skin reflection model is introduced (which model light-tissue interaction), (ii) two classical algorithmic methods[2] are derived based on the linear model and (iii) the deep learning approach is formulated. The latter is followed by the introduction of probabilistic neural networks as a solution for the prediction uncertainty problem.

Chapter 3 In this chapter, the methods which I utilized throughout this study are presented. First the process of data collection, then the different loss functions and augmentations applied for neural network training, then the neural networks themselves, and finally the process of evaluation is described.

Chapter 4 In this chapter, the results are presented and interpreted. First, the created database is discussed. After that, the performance of different neural networks and training techniques are analyzed. Then the DL approach is compared with a standard classical algorithmic approach on our database. Finally, the feasibility of application in a resource-limited environment is inspected.

Chapter 5 In this chapter, a summary of the set goals and final achievements is presented. Furthermore, possible future directions are discussed.

[2]namely PBV [10] and POS [6] core rPPG algorithms

Chapter 2

Background

It was first shown by Verkruysse et al. [1] in 2008 that periodic blood volume changes can be detected with an average color camera under normal ambient lighting conditions – which initiated the field referred to as remote-PPG (rPPG). Although these changes in skin color are subtle and imperceptible to the naked human eye, a camera is able to detect them. They also observed that the green color channel contains the strongest rPPG-signal. Blood absorbs blue light the most, followed by green and red, but their depth of penetration (into the skin) follows just the opposite order. Considering these two factors, the green camera channel (the golden mean) happens to contain the most pulse information. Later on, Corral et al. [11] obtained consistent results with this, they measured the backscattered rPPG-signals from the human forehead for wavelengths ranging from 380 nm to 980 nm (see Figure 2.1).

If we look more closely at the physical and biological origin of the rPPG-signal, we find that it is highly controversial, even today. The most basic and obvious idea is that the detected light which is coming from inside the skin is getting modulated by the blood volume changes in the vasculature due to the cardiac-cycle, but Kamshilin et al. [12], [13] argued that the green light – which has the largest pulsatility – do not reach subcutaneous vasculature and proposed a new physiological model of light-tissue interaction. According to this model, the pulsatile pressure of the arteries compresses and decompresses the density of capillaries in the dermis, thus modulating the blood volume in the capillary bed, which in turn modulates the backscattered green light. The latest study [14] supports the (former) volumetric hypothesis with living skin experiments and Monte Carlo simulations of rPPG-amplitude in visible light and infrared. They found that green wavelengths probe dermal arterioles while red and IR wavelengths also reach subcutaneous vasculature.

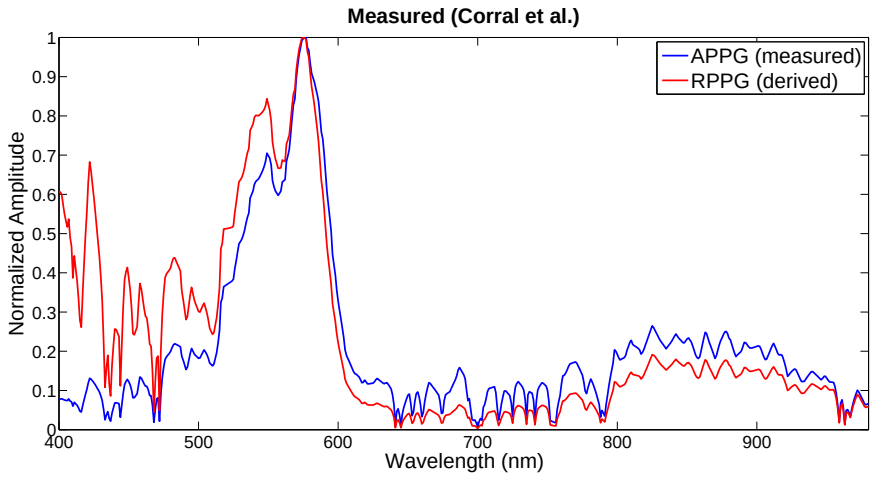

Figure 2.1: [11] The pulsatile amplitude along the spectrum for a rPPG setup, the green light is the optimal. The absolute rPPG amplitude (the AC component of the pulse signal) denoted with blue color and the relative rPPG amplitude (AC over DC (AC/DC) component) denoted with red.

After discussing the fundamental phenomenon, let us turn our attention to the processing of the detected data. The camera-signal measured from the skin results in the raw rPPG-data (i.e. sequence of uncompressed images) which is the input of the rPPG-function which calculates the rPPG-signal, or less formally the pulse-signal. The input (rPPG-data) is 4-dimensional: spatial dimensions (H x W, height and width); temporal dimension (D, number of frames); and color channel dimension (C, most often red, green, and blue, i.e. C=3). Another important property is the time distance between frames, i.e. the frame rate (aka frames per second, FPS) of the recording. The sampling frequency has to be at least twice the frequency of the signal to be observed, this is the minimal condition to be able to measure a periodic signal with a given frequency, also known as the Nyquist–Shannon sampling theorem.

Let's assume an RGB-camera for the sake of simplicity. In this case, the rPPG-data can be considered as a spatio-temporal RGB-signal. This is contaminated with environmental and sensor noise, the former arises mainly from body motions and illumination variation, the latter is the intrinsic quantization noise of the camera. Environmental noise appears synchronously on each color channel, while sensor noise is independent between channels. In consequence, the sensor noise can be eliminated with spatial averaging or by temporal filtering, while environmental noise can be eliminated with the linear combination of the color channels. In general, with n number of independent linear

equations $(n-1)$ terms can be eliminated, thus 2 environmental artifacts can be removed mathematically in the case of an RGB-camera.

The goal of the rPPG-function is to design a projection that separates the pulsatile component from the environmental artifacts. One way is to apply general signal-source separation methods like Principal Component Analysis (PCA) or Independent Component Analysis (ICA) – here we assume that pulse and noise come from independent sources. Another way is the model-based approach where we fabricate biometric signatures utilizing the biophysical fact that pulse has well defined characteristic color variation direction in RGB space – these are introduced based on the (simplistic linear) skin optical reflections model. And finally, we can train DNNs to learn sophisticated, even nonlinear relations in order to extract pulse. In the following, these options and the related work will be discussed in more detail.

2.1. Related Work in rPPG

The main concern with rPPG is the low signal-to-noise ratio (SNR) and its lack of robustness against the subject motion – since motion also induces intensity variations which corrupt the rPPG-signal. Several core rPPG algorithms have been proposed that aim to solve this issue by extracting the pulse-signal information from the color channels. As mentioned before, in the case of three color channels, two artifacts (e.g. motion-induced artifacts, illumination variation artifacts) can be eliminated theoretically.

First, blind source separation techniques were utilized in 2011: PCA-based [15] and ICA-based [16], which assumes that the observed signal is composed of independent sources. After that, numerous more sophisticated (model-based) core algorithms were developed based on prior knowledge of intrinsic color feature of the pulse signal, for example, the "Pulse Blood-volume Vector" PBV [17] and the "Plane Orthogonal to Skin" POS [6] method.

These are only the core algorithms that extract the pulse signal from the color channels, however, there are many other steps: Region of Interest (RoI) detection [18]; RoI tracking; and combining the signals arising from distinct skin locations. Vogels et al. proposed a fully automated framework for sleep monitoring [8] that includes all processing steps and also able to measure blood oxygenation (SpO2). Their algorithm functions well when there is no vigorous movement. An RoI is selected and a tracker is initialized automatically based on an elaborate similarity matrix representing skin-pulsatility – it is a computationally expensive multi-level application.

A more compact and computationally more efficient algorithm was proposed by Wang et al. named Full Video Pulse extraction (FVP) [7] that bypasses RoI selection and tracking, utilizing the color features of the video. The author emphasized its suitability for long-term monitoring in cases when the background is constant – e.g. monitoring infants in an incubator, therefore this algorithm is implemented and tested on our database.

Most recently, the popular deep learning approach appeared also in this field [4], [5], [9], [19], [20]. The first end-to-end convolutional neural network (CNN) was proposed by Chen et al. named DeepPhys [4] which utilizes attention mechanism: it consists of two connected streams, the attention stream – which input is a single frame and responsible for RoI selection – and the motion stream – which input is the normalized frame difference and responsible for pulse signal extraction. Just recently, Zhan et al. analyzed this network [21] and concluded that it learns the wavelength-dependent characteristics of blood absorption color variation to extract physiological signals – as classical algorithms do –, and that the choice and parameters (e.g. phase, spectral content) of the reference-signal may be more crucial than anticipated – i.e. the network learns and performs better if the label is well selected/prepared/aligned.

Spetlík et al. developed two separate convolutional networks [9], one for extracting pulse signal from video and the other for estimating pulse from the previously extracted signal – with the application of a signal-to-noise ratio (SNR) based loss function.

A 3D-CNN network named PhysNet was introduced by Yu et al. [5] which exploits not only spatial but spatio-temporal features. This network is completely end-to-end – there are no pre-processing steps –, its input is a sequence of images and the output is the corresponding pulse signal.

All deep-learning solutions have been applied only on adult facial videos so far. In this study, we focus on neonate monitoring using the PhysNet and DeepPhys deep neural network architectures.

Up to now, only classical algorithms have been used for video-based infant monitoring, these are reviewed in the following. Camera-based non-contact estimation of heart rate in the neonatal care unit was first reported in 2012 [22]. In this work, seven infants were monitored for only 30 seconds with a webcamera 20 cm away from the face and with special illumination. In another study [23], video camera recordings were conducted of 19 infants. The camera was placed on a tripod at approximately 1 meter from the infant and the face region was recorded for up to 5 minutes. RoI was manually selected and heart rate (HR) was estimated from Fast Fourier Transform (FFT) analysis of the

green channel. In 13 of 19 neonates was possible to derive HR estimates for 90% of the time. These studies were conducted on brief video recordings with ideal conditions (e.g. controlled illumination, resting patient). However, it is at most importance to analyze the feasibility of the method in a long-term and not controlled fashion, with real hospital environment and regular light conditions – and most importantly, without affecting patient care. With these in mind, Villarroel et al. [24] directed their study, that contains the analysis of around 25 hours long "valid camera data". The video recording is labeled as "valid camera data" outside the following occurrences: (1) regular interaction between the clinical staff and the baby; (2) clinical interventions; (3) baby taken out of the incubator to be held by the mother (kangaroo care). In addition to HR and RR, the relative changes in SpO2 were also measured. The authors concluded that it is possible to monitor HR, RR, and SpO2 continuously in the neonatal care unit, with clinically useful accuracy. However, they also reported that even during stable periods ("valid camera data") some phenomena prevented the accurate estimation of the vital signs, namely: (1) major changes in lightning conditions; (2) variations in the baby's activity pattern (heavy subject motion); (3) lack of visible skin area (the neonate is covered in sheet). These factors reduced the time for which they could accurately estimate vital sign values from 24.9 to 20.1 hours. For the remaining part, they could estimate HR, to be more specific: the root-mean-square error (RMSE) between their estimate and the ECG-derived value was less than 5 beats-per-minute (BPM). The average RMSE was 3.95 BPM.

The previous methods were not motion-robust and worked only in the daytime (when visible light is present), therefore M. van Gastel et al. developed a system [25] that functions in near darkness (using custom near infra-red (NIR) illumination and monochrome cameras with filters) and utilizes modern motions-robust core rPPG algorithm, namely the PBV [17]. The focus of this study was to compare systems with 2 or 3 wavelengths and to compare different body parts (face and upper torso) for signal extraction (with manually selected and tracked RoI (Region of Interest)). To this end, the authors created a dataset with 7 different subjects and a total duration of 134 minutes. They demonstrated the feasibility of non-contact cardiac monitoring of neonates in NIR and that the upper torso also contains valuable pulse information. Furthermore, the authors reported that accurate results were not only obtained for scenarios without motion but also for common movements of neonates – such as wriggling, turning and respiration induced motion. However, in the proposed system RoI selection is manual and long-term RoI tracking is not stable (and itself is a challenge, i.e. not suitable for continuous monitoring).

Considering all these results and facts, the application of deep neural networks (DNN) on neonate monitoring is promising, as it is capable to solve all common tasks/issues (e.g. RoI selection, tracking, motion compensation) internally, in an end-to-end fashion, and might perform better in motion scenarios (because it can handle non-linear relationships). Furthermore in classical approaches, facial regions of the infants were generally selected as RoI, although skin-pulsatility is only slightly lower in the upper-torso region [25] – which is thus not utilized. In contrast, deep CNNs are capable to learn sophisticated weighted RoI (attention) maps and "track" them precisely frame-by-frame with great stability.

This motivated us to investigate the application of deep learning techniques for continuous premature infant monitoring.

2.2. Mathematical Formulation of rPPG

In this section, we introduce the linear and nonlinear skin reflection model [6], [19], then the classical and DL approach is formulated based on this model. Last but not least, probabilistic neural networks are introduced for the prediction uncertainty problem.

2.2.1 Skin Reflection Model

Here we use the model proposed in [6] which is based on Shafer's Dichromatic Reflection Model (DRM). Assume that the light source has constant spectral composition but the intensity level is allowed to vary. The camera records skin region only (see system setup in Figure 2.2). We can define the RGB intensity values of the p-th skin pixel in time with the following function (which includes physiological, light source and camera sensor features):

$$\mathbf{C}_p(t) = I(t) \cdot [\mathbf{v}_s(t) + \mathbf{v}_d(t)] + \mathbf{v}_n(t) \tag{2.1}$$

where $\mathbf{C}_p(t)$ denotes a vector containing the RGB intensity values observed by the camera; $I(t)$ is the luminance intensity level, which changes (i) with the light source (ii) and with the distance between the light source, skin-tissue and camera. In the DRM model $I(t)$ is modulated by two components (see Figure 2.2): (i) by the specular reflection $\mathbf{v}_s(t)$, which is a mirror-like light reflection from the skin surface; (ii) and by the diffuse reflection $\mathbf{v}_d(t)$, which penetrates into deeper layers of the skin. The last component in the equation, $\mathbf{v}_n(t)$ is the quantization noise of the camera which can be easily eliminated by averaging some pixels (for example averaging a 25x25 pixels subarea) – i.e. downsampling the image, this is the 0th step for each approach:

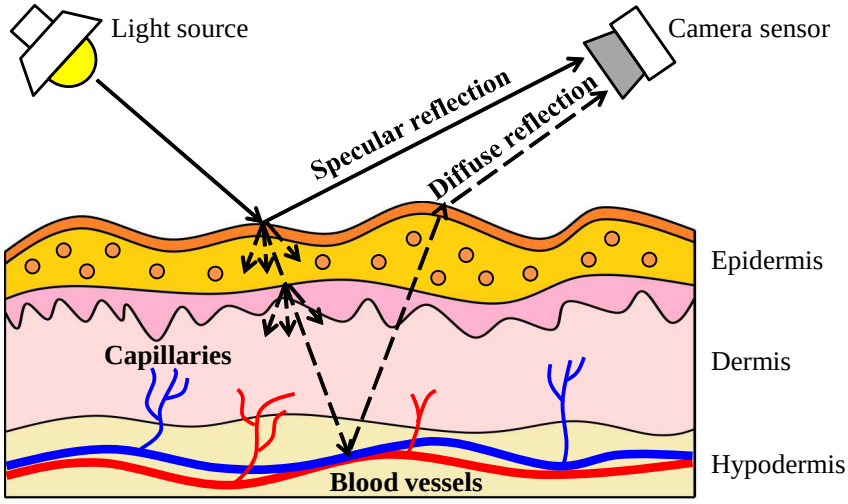

Figure 2.2: [6] Skin reflection model illustration.

$$\mathbf{C}_m(t) = I(t) \cdot [\mathbf{v}_s(t) + \mathbf{v}_d(t)] \tag{2.2}$$

where $\mathbf{C}_m(t)$ contains the RGB values of the downsampled image at the m-th mean-pixel value and at time-point t. The remaining three terms (luminance level I, specular component $\mathbf{v}_s$, diffuse component $\mathbf{v}_d$) can be decomposed into a stationary and a time-varying part:

$$I(t) = I_0 \cdot (1 + i(t)) \tag{2.3}$$

$$\mathbf{v}_s(t) = \mathbf{u}_s \cdot (s_0 + s(t)) \tag{2.4}$$

$$\mathbf{v}_d(t) = \mathbf{u}_d \cdot d_0 + \mathbf{u}_p \cdot p(t) \tag{2.5}$$

where I_0 is the average luminance intensity and $i(t)$ is its variation. In the next equation, $\mathbf{u}_s$ denotes the unit color vector of light spectrum (mirror like reflection), s_0 is the stationary and $s(t)$ is the varying part of the specular reflection (it is induced by body motion and illumination changes). In the last equation, $\mathbf{u}_d$ and d_0 denotes the unit color vector of skin-tissue and the stationary reflection strength which is mainly characterized by the hemoglobin and melanin content of the skin (which determines the skin color). Finally, the diffuse component $\mathbf{v}_d(t)$ is varied by blood volume changes: $\mathbf{u}_p$ denotes the relative pulsatile strengths in RGB channels and $p(t)$ denotes the Blood Volume Pulse (BVP) signal. If we substitute equation 2.3-2.5 back to 2.2 we obtain the following:

$$\mathbf{C}_m(t) = I_0 \cdot (1 + i(t)) \cdot [\mathbf{u}_s \cdot s_0 + \mathbf{u}_s \cdot s(t) + \mathbf{u}_d \cdot d_0 + \mathbf{u}_p \cdot p(t)] \tag{2.6}$$

The stationary parts in specular and diffuse reflection can be combined into one component ($\mathbf{u}_c \cdot c_0 = \mathbf{u}_s \cdot s_0 + \mathbf{u}_d \cdot d_0$) representing the stationary skin reflection (which characterizes the perceptible skin color at given illumination source):

$$\mathbf{C}_m(t) = I_0 \cdot (1 + i(t)) \cdot [\mathbf{u}_c \cdot c_0 + \mathbf{u}_s \cdot s(t) + \mathbf{u}_p \cdot p(t)] \tag{2.7}$$

where $\mathbf{u}_c$ denotes the unit color vector of the skin reflection and c_0 denotes the reflection strength. We can further simplify this equation with performing the multiplication and neglecting the small terms, i.e. which contain products of two AC parts, e.g. $p(t) \cdot i(t)$ or $s(t) \cdot i(t)$:

$$\mathbf{C}_m(t) \approx \mathbf{u}_c \cdot I_0 \cdot c_0 + \mathbf{u}_c \cdot I_0 \cdot c_0 \cdot i(t) + \mathbf{u}_s \cdot I_0 \cdot s(t) + \mathbf{u}_p \cdot I_0 \cdot p(t) \tag{2.8}$$

The approximation holds because the AC-modulation terms are orders of magnitudes smaller than the DC terms, thus the product modulation terms are so small that they are probably undetectable by the camera.

Equation 2.8 tells us that the recorded signal is the linear combination of three independent source-signals: $i(t)$, $s(t)$ and $p(t)$. The goal of the classical algorithms is to define such linear projections with which the pulse-signal $p(t)$ can be extracted from $\mathbf{C}_m(t)$.

If the task is SpO2 measurement, then the terms containing $\mathbf{u}_c$ and $\mathbf{u}_p$ are used as they depend on, and change with deoxy/oxy-hemoglobin concentration variation.

A more realistic model proposed by the creators of DeepPhys DNN architecture [4] also considers the appearance of BVP effect in the specular and intensity terms:

$$i(t) = \Psi(m(t), p(t)) \tag{2.9}$$

$$s(t) = \Phi(m(t), p(t)) \tag{2.10}$$

$$\mathbf{C}_m(t) \approx \mathbf{u}_c \cdot I_0 \cdot c_0 + \mathbf{u}_c \cdot I_0 \cdot c_0 \cdot \Psi(m(t), p(t)) + \mathbf{u}_s \cdot I_0 \cdot \Phi(m(t), p(t)) + \mathbf{u}_p \cdot I_0 \cdot p(t) \tag{2.11}$$

where $m(t)$ includes all the non-physiological variations (e.g. flickering of light source, complex body motions) and $p(t)$ is the already discussed blood volume pulse (BVP). Consequently, $i(t)$, $s(t)$ and $p(t)$ are no longer independent source signals. Note that the BVP has also mechanical effects not just optical (e.g. periodic movement of head or shirt collar, the small displacement of skin surface, i.e. ballistocardiographic effect), this is the reason why $p(t)$ appears in Equation 2.9 and 2.10. The interaction between physiological and non-physiological motion, that is inside $\Phi(\cdot)$ and $\Psi(\cdot)$, are usually complex nonlinear functions. Equation 2.8 assumes linear connection between $\mathbf{C}_n$ and $p(t)$, which generally holds when $m(t)$ is small (i.e. in ideal case when the subject is still, motionless), but in

most real-life situation the linear assumption will harm the measurement performance. From another perspective, in this nonlinear model we consider not only the rPPG-signal (optical in nature), but also the BCG-signal (mechanical in nature) which may result in higher performance and yields the so called remote BVP-signal (rBVP), but for the sake of simplicity we will refer to this simply as the pulse-signal.

So far we have only discussed how to extract pulse-signal from one averaged skin pixel $\mathbf{C}_m$, but we have generally hundreds of them and they (the averaged pixels) act like independent sensors, therefore they can be effectively combined to increase the signal-to-noise ratio of the final pulse-signal. Furthermore, the tracking of these averaged skin pixels is also important, because subject motion result in skin pixel displacement which corrupts the calculation. A huge advantage of a DNN is that it can learn sophisticated weight-maps for signal combination and solve all algorithmic steps internally in a compact and efficient way. In the next section, some core rPPG algorithms are presented which input is one averaged skin pixel $\boldsymbol{C}_m$ and their output is the extracted pulse-signal $p(t)$.

2.2.2 Classical Algorithmic Approaches

All of the classical algorithms are based on the linear skin reflection model, eq. 2.8. In the mathematical formulation of classical rPPG core methods, we mainly rely on the work [6].

Blind Source Separation (ICA/PCA) Equation 2.8 suggest that Blind Source Separation (BBS) may be ideal for pulse retrieval from $\mathbf{C}(t)$. These kind of methods (eg.: ICA, PCA) can be expressed as follows:

$$\mathbf{S}(t) = \mathbf{W} \cdot \mathbf{C}(t) \tag{2.12}$$

where $\mathbf{S}(t)$ denotes the source-signals which consists the pulse and artefacts; $\mathbf{W}$ denotes the de-mixing matrix obtained from PCA or ICA. After the BBS operation we have to select the most periodic signal from the source-signals $\mathbf{S}(t)$ as the pulse-signal. Consequently, periodic movements distort the results of these methods. In addition, BSS techniques are statistical solutions for signal-processing problems which do not utilize the unique properties inherent in the specific rPPG task. In the following paragraph, we show some of these methods which exploit the optical features of skin reflection.

Model-based methods: PBV Model-based methods – in contrast with BSS – make use of color channel properties. One common step in many of these methods is to eliminate

the DC-level from $\mathbf{C}(t)$, which can be done by temporal normalization:

$$\mathbf{C}_n(t) = \mathbf{N} \cdot \mathbf{C}(t) = \mathbf{1} \cdot (1 + i(t)) + \mathbf{N} \cdot \mathbf{u}_s \cdot I_0 \cdot s(t) + \mathbf{N} \cdot \mathbf{u}_p \cdot I_0 \cdot p(t) \tag{2.13}$$

where $\mathbf{N}$ is a diagonal matrix that temporally normalizes $\mathbf{C}(t)$ and $\mathbf{1} = [1, 1, 1]^T$. The DC removed (centered) signal is defined as:

$$\mathbf{C}_c(t) = \mathbf{C}_n(t) - \mathbf{1} = \mathbf{1} \cdot i(t) + \mathbf{N} \cdot \mathbf{u}_s \cdot I_0 \cdot s(t) + \mathbf{N} \cdot \mathbf{u}_p \cdot I_0 \cdot p(t) \tag{2.14}$$

The PBV method projects $\mathbf{C}_c(t)$ directly to the hand-crafted pulsatile direction:

$$p(t) \propto \mathbf{C}_c(t)^T \cdot \mathbf{z} \tag{2.15}$$

where $\mathbf{z}$ is a vector defining the direction of projection:

$$\mathbf{z} \propto \Sigma^{-1} \cdot \mathbf{u}_{pbv} \tag{2.16}$$

where $\Sigma = \overline{[\mathbf{C}_c(t) \cdot \mathbf{C}_c(t)^T]}$ is a 3x3 covariance matrix estimated from the video content, $\overline{[.]}$ denotes temporal averaging and $\mathbf{u}_{pbv}$ is the prior-known[1] pulse blood volume vector. This method assumes that $i(t)$ and $s(t)$ is not correlated with $p(t)$ in eq. 2.8.

Model-based methods: POS Here we project $\mathbf{C}_n(t)$ to the plane orthogonal to $\mathbf{1}$, which can be written as:

$$\mathbf{S}(t) = \mathbf{P}_p \cdot \mathbf{C}_n(t) \approx \mathbf{P}_p \cdot \mathbf{N} \cdot \mathbf{u}_s \cdot I_0 \cdot s(t) + \mathbf{P}_p \cdot \mathbf{N} \cdot \mathbf{u}_p \cdot I_0 \cdot p(t) \tag{2.17}$$

where $\mathbf{S}(t) = [S_1(t),\ S_2(t)]^T$ is the projected RGB-signal to a plane which most likely encapsulates the pulse direction and $\mathbf{P}_p$ denotes a 2x3 projection matrix with rows orthogonal to each other and also to $\mathbf{1}$. This method does not require exact knowledge in contrast with PBV, but requires only the order of pulsatile strength of the color channels which in case of an RGB camera is G,B,R. In this case the projection axes (which also satisfies the orthogonality constraints) can be defined as:

$$\mathbf{P}_p = \begin{pmatrix} 0 & 1 & -1 \\ -2 & 1 & 1 \end{pmatrix} \tag{2.18}$$

which combines temporally normalized RGB-signals. Finally, to find the exact projection direction alpha-tuning [10] is applied:

[1]It can be measured, but depends on illumination light spectrum and camera sensor.

$$h(t) = S_1(t) + \frac{\sigma(S_1)}{\sigma(S_2)} \cdot S_2(t) \tag{2.19}$$

where $\sigma(.)$ denotes the standard deviation operator and $h(t)$ is the estimated pulse signal segment.

This is the state-of-the-art core rPPG algorithm with highest overall performance [6], therefore we have chosen (and implemented) this one to compare it with deep learning approaches (see in Results section).

2.2.3 Deep Learning Approaches

General Introduction to Deep Learning

The DL approach can handle non-linearity, thus in theory it is able to extract $p(t)$ from the non-linear model, eq. 2.11. In contrast with the previous methods, in supervised learning we don't have hand-crafted, prior-known features but only large models with learnable parameters and datasets containing input-output pairs on which the optimization of the model-parameters can happen.

The dataset $\mathcal{D}$ is consisted of input-output pairs

$$\mathcal{D} = \mathcal{X} \times \mathcal{Y} = \{(\mathbf{x}_i, \mathbf{y}_i) \mid i = 1,\ 2,\ \ldots\ ,\ N\}$$

where N is the number of samples in the dataset, $\mathcal{X}$ is the input set and $\mathcal{Y}$ is the output domain. We can formalize a deep neural network as a cascade of nonlinear layers:

$$\hat{\mathbf{y}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\Theta}) = \boldsymbol{f}^{(l)}\left(\boldsymbol{f}^{(l-1)}\left(\ \ldots\ \boldsymbol{f}^{(1)}\left(\boldsymbol{x}, \boldsymbol{\Theta}^{(1)}\right)\right)\right) \tag{2.20}$$

where $\hat{\mathbf{y}}$ is the output/prediction, $\boldsymbol{x} \in \mathcal{X}$ is the input, Θ denotes all the model parameters, l is the number of layers, $\boldsymbol{f}^{(i)}(\boldsymbol{z}^{(i)}, \boldsymbol{\Theta}^{(i)})$ is the ith layer corresponding to a nonlinear transformation of intermediate activation $\boldsymbol{z}^{(i)}$ (note that $\boldsymbol{z}^{(1)} = \boldsymbol{x}$) and $\boldsymbol{\Theta}^{(i)}$ denotes the parameters of the ith layer. The function $\boldsymbol{f}$ is a universal function approximator.

We have to define a loss function $\mathcal{L}(.)$ which compares the output of the network with the corresponding label and yields an error measure. Our goal is to tune the model parameters such that the error is minimized:

$$\boldsymbol{\Theta}^* = \min_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{f}, \boldsymbol{\Theta}, \mathcal{X}, \mathcal{Y}) \tag{2.21}$$

where $\boldsymbol{\Theta}^*$ denotes the learned model parameters. The model architecture $\boldsymbol{f}$, the quality and quantity of training dataset $\mathcal{D}$ and the choice of objective function $\mathcal{L}$ is essential and

decides the success of model training. After the model parameters are tuned, it is able to make good predictions for an input which is in the training set (or which is similar to those in the training set):

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\Theta}^*) = \hat{\boldsymbol{y}} \approx \boldsymbol{y} \qquad \boldsymbol{x} \in \mathcal{X},\ \boldsymbol{y} \in \mathcal{Y} \tag{2.22}$$

but it is questionable how well it can generalize for samples which are outside of the training dataset (though there are many generalization techniques to avoid overfitting):

$$\boldsymbol{f}(\boldsymbol{x}', \boldsymbol{\Theta}^*) = \hat{\boldsymbol{y}} \overset{?}{\approx} \boldsymbol{y}' \qquad \boldsymbol{x}' \notin \mathcal{X},\ \boldsymbol{y}' \notin \mathcal{Y} \tag{2.23}$$

where $(\boldsymbol{x}', \boldsymbol{y}')$ is a known input-output pair which is not in the training set and also out of its distribution. For example, the neural network was trained only on white cats and then a black cat is given as input, can it still recognize that it is a cat?

The neural network outputs a point estimate $\hat{\boldsymbol{y}}$ without knowing its certainty which can be a problem in many cases – especially in medical applications. It would be desirable to know how accurate the prediction is. It is discussed in the following section.

Uncertainty Estimation in Deep learning

Probabilistic deep learning is a relatively new but exploding research field [26]–[30] (mainly applied on the problem of self-driving cars [28]) which aims to provide probability instead of simple point estimates – which is crucial in most medical application, also in our case (remote monitoring of vital signs) but not yet utilized.

The source of uncertainty can be divided into two components: (1) *aleatoric* uncertainty which comes from the noise inherent in the observation (e.g. camera noise or motion noise) and cannot be decreased with more training data; (2) *epistemic* or aka model uncertainty which can be reduced if more data were to be collected. We focus on the former which can handle situations when vigorous movements corrupt the observation (motion noise): instead of blindly outputting an estimate, it can indicate that the prediction might be inaccurate.

The easiest way to create a probabilistic neural network is to replace the output layer with a probabilistic output layer (ProbOut [27]) which replaces point predictions of deterministic networks by distributions over the output. A neural network is fully probabilistic when the intermediate activations are replaced by distributions as well, but it requires major modification on the architecture and training process (also doubles the number of trainable parameters), thus in this study, ProbOut was applied to estimate pulse-rate.

As mentioned, the output of the network is a probability distribution over the output space $\boldsymbol{y}$ which can be formalized as: $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\Theta}) = p(\cdot|\boldsymbol{x})$. The predictive distribution $p(\boldsymbol{y}|\boldsymbol{x})$ is restricted to be parametric and let the last layer $\boldsymbol{f}^{(l)}$ predict the parameters of this distribution. In case of the normal distribution $p(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{v})$ there are two parameters: its mean parameter $\boldsymbol{\mu}$ decodes the prediction and its variance parameter $\boldsymbol{v}$ the associated uncertainty. These parameters are calculated by the network:

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\Theta}) = (\boldsymbol{\mu}, \boldsymbol{v}) \tag{2.24}$$

We want to associate maximum probability to the ground truth value in the output space:

$$\boldsymbol{\Theta}^* = \max_{\boldsymbol{\Theta}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{N}\left[\boldsymbol{y}_i \mid \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{\Theta})\right] \qquad \boldsymbol{x}_i \in \mathcal{X},\ \boldsymbol{y}_i \in \mathcal{Y} \tag{2.25}$$

If the predicted mean parameter differs from the reference with large confidence (small variance parameter) that will result in small probability (near to zero) which is strongly punished, this will tend to increase the variance parameter, but if the predicted mean is close to the ground truth higher confidence (smaller variance parameter) is preferred as it will result in larger probability.

Chapter 3

Methods

3.1. Data Collection

A *Basler acA2040-55uc* color camera was used for image acquisition with a resolution of 500x500 pixels at 20 frames per second (FPS) and with 12 pixel bit depth. The recorded frames are saved in raw (uncompressed) binary format. For this, a recording software is developed using C++ Qt framework along with OpenCV and Basler's Pylon camera API. The video-recording was done from a number of different camera angles and optics.

Synchronized reference data for the videos – including the pulse-signal and the pulse-rate – were provided by the vital sign monitoring systems of the hospital, from both the *Philips IntelliVue MP20* or *MP50* models. There is an older recording where we used a pulse-oximeter reference named *EVAL-MECG35 SpO2 Module* and manufactured by *Zug Medical Systems SAS* which is also suitable for neonatal usecase.

For illumination, ambient light was used – i.e. no dedicated light source was provided, thus the source of illumination is natural sunlight or the fluorescent lamps of the hospital (or the combination of the two).

A database viewer and tagger system is developed by another member of our team to be able to handle the large amount of video.

Another program was developed to convert selected parts of the database to HDF5 file format (DB-to-HDF5) with many conversion options (e.g. resolution, interpolation method, bit depth etc...) for neural network training and testing. The consideration behind using HDF5 instead of sequence of images is that it is much easier to handle a large file than many little ones. Furthermore, HDF5 is able to load data very efficiently, metadata can be easily added and data can be hierarchically structured internally.

3.2. Loss Functions

3.2.1 NegPeaLoss

This loss function can be utilized if the network outputs a time signal and corresponding reference signal is available. A Pearson correlation based loss function was introduced in [5] which has many benefits over simple pointwise L1 or L2 loss because this considers the temporal nature and correlation of the signals. NegPeaLoss maximizes trend similarity (i.e. waveform) and minimizes peak location errors. It can be formulated as follows:

$$\mathrm{L} = 1 - \frac{T\sum \hat{y}_i y_i - \sum \hat{y}_i \sum y_i}{\sqrt{\left(T\sum \hat{y}_i^2 - (\sum \hat{y}_i)^2\right)\left(T\sum y_i^2 - (\sum y_i)^2\right)}} \tag{3.1}$$

where T is the length of the signals, $\hat{y}$ is the predicted signal, y indicates the ground truth and $\sum$ is summing over all elements ($i = 1, 2, \ldots, T$).

3.2.2 SNRLoss

If the output of the model is a periodic time signal and frequency rate reference is available then SNRLoss can be applied to train the model. This loss can be written as:

$$\mathrm{L} = -\log_{10}\left(\frac{\sum_{f=80}^{250} S(f) \odot u(f)}{\sum_{f=80}^{250} S(f) \odot (1 - u(f))}\right) \tag{3.2}$$

where $S(f)$ is the power spectrum of the output signal, u is the reference binary template function containing 1 at the ground truth frequency and 0 at other f frequencies, finally $\odot$ denotes element-wise multiplication. The summing is performed over the pulse range, from 80 BPM to 250 BPM. The precise definition for the reference binary template function u is the following:

$$u(f) = \begin{cases} 1, & \text{if } f \in [f^* - \delta, \quad f^* + \delta] \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

where f^* is the reference pulse-rate and δ is the acceptance value chosen to be 6 BPM. Note that the power spectrum calculation consists of differentiable steps, therefore it is working well with the back-propagation algorithm, thus can be used for training the neural network.

3.2.3 GaussLoss and LaplaceLoss

These are loss functions which can be applied in case of a probabilistic output layer – which estimates the parameters of the probability distribution over possible outputs (i.e. over the output space). Both the Gauss and Laplace distribution has two parameters, one of them is the mean μ and the other one determines the broadness of the distribution. Let's start with the Gauss (aka normal) distribution:

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{3.4}$$

where σ^2 is the variance. For numerical reasons, we want to minimize the negative logarithm of this function which can be written in the following form:

$$\mathrm{L}_{\mathcal{N}} = e^{-s} \cdot (y-\mu)^2 + s, \quad \text{with} \quad s = \ln \sigma^2 \tag{3.5}$$

where y is the ground truth, μ and σ are the predictions made by the neural network. Note that this is equivalent to the non-probabilistic L2 loss when $s = 0$.

The Laplace Probability Distribution Function (PDF) can be written in the following way:

$$\mathcal{L}(x \mid \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-b|}{b}\right) \tag{3.6}$$

where b is often referred to as the diversity parameter (determining the broadness of the distribution). Similarly, the objective function is constructed by taking the negative natural logarithm of the PDF, now resulting in a L1 like loss:

$$\mathrm{L}_{\mathcal{L}} = e^{-s} \cdot |y-\mu| + s, \quad \text{with} \quad s = \ln b \tag{3.7}$$

Ultimately, these loss functions tend to move the predicted mean μ to the ground truth and minimize uncertainty s, but this latter parameter (if it takes larger value) can also suppress the error term of μ prediction – which is beneficial in case of inaccurate μ estimation – as it is multiplied by e^{-s} .

3.3. Augmentations

Several spatial domain image augmentations were applied and one time domain augmentation was also developed (referred to as frequency augmentation) to improve generalization and avoid overfitting.

3.3.1 Image augmentations

The input of the PhysNet architecture is a sequence of images, thus the same augmentation is applied for each image in the sequence and the parameters of the augmentations vary only between different input sequences. The applied augmentations are listed below:

1. vertical flip
2. horizontal flip
3. rotation
4. brightness tuning
5. contrast tuning
6. saturation tuning

The first three is for pose-invariance and the last three is for illumination-invariance. Note that color (or hue) tuning is not used as it would corrupt the relative pulsatile property of the color channels.

3.3.2 Frequency augmentation

A novel augmentation method is introduced for periodic time-series signals, called frequency augmentation. This is done by resampling in time (using linear interpolation) a shorter or longer video back to the (original) fixed network input length (eg. 128 frames which equal 6.4 sec @ 20 FPS) and changing the reference accordingly. With this, we can imitate pulse-rates in the whole range from 80 BPM to 250 BPM, consequently, we can create a training-set with more diverse pulse-rate values, thus the network will not be biased towards the mean pulse-rate and will handle marginal cases better.

3.4. Neural Networks and Training

For implementation, the PyTorch deep learning framework was used and model training was conducted on a NVIDIA GeForce RTX 2080 Ti 10GB GPU.

3.4.1 DeepPhys

This network [4] has two inputs for its two computation stream: (1) a single frame for the attention stream and (2) the normalized difference of two consecutive frames for the

motion stream, as shown on Figure 3.1. It is trained on the derivative of the pulse signal because it is an appropriate target for the normalized frame difference input.

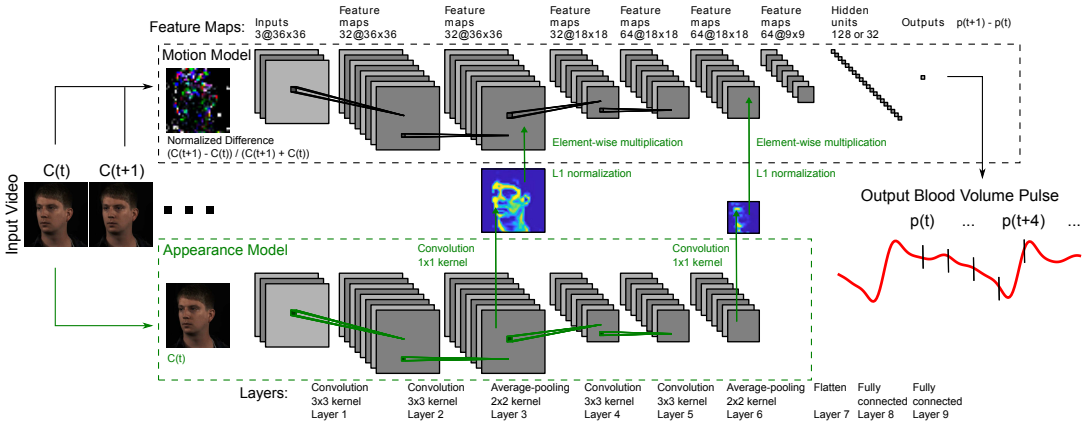

Figure 3.1: The illustration of DeepPhys architecture from the original paper [4].

This is a 2D convolutional network with attention mechanism and a fully connected head. Interestingly, the not so common *tanh* activaton function is applied and *avgpool* instead of *maxpool* as the authors claim it is more suitable for this specific task. The appearance stream is responsible to highlight regions from where pulse information can be extracted with high probability while the motion stream is responsible to detect blood volume change. The final attention weight maps are multiplied with the motion stream activations (each channel element-wise) in the middle and also at the end of the network – which acts as a focus on the important details. The output of the network is a single point (from the derivative of the pulse signal).

Because of its 2D structure, DeepPhys is really fast compared to the 3D PhysNet architecture. Moreover, DeepPhys also slightly outperforms PhysNet (at least on one public dataset) as shown in [5] which is consistent with our results.

For training, the ADAM optimizer and mean absolute error loss (L1 loss) was applied with a learning rate of 1e-4. It was trained for 60 epochs with image augmentations. The batch size was set to 128.

3.4.2 PhysNet

This network operates on video input and extracts pulse signal from it. The best performing neural network variant was picked and implemented from [5], namely PhysNet128-3DCNN-ED which is a fully 3D convolutional (FCN) architecture with encoder-decoder structure in time domain and with 128 frames input video length (as its name implies),

depicted on Figure 3.2. The inputs are first normalized to be in [0, 1] range and then zero centered (its mean is subtracted) carefully such that the relative color channel information is not corrupted (what a channel-wise centralization would do).

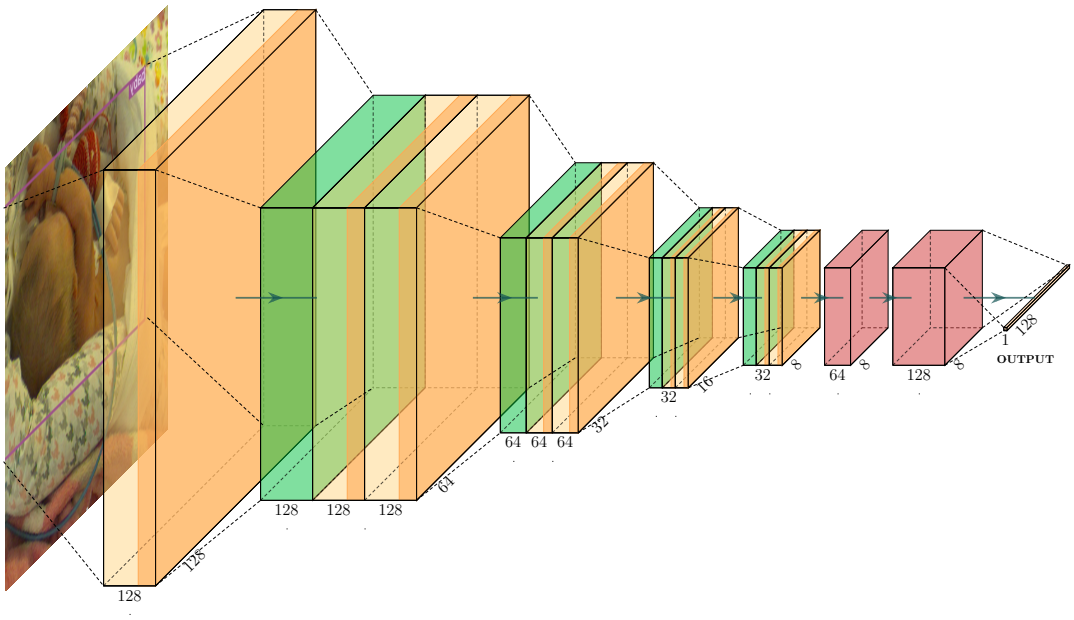

Figure 3.2: Illustration of the PhysNet architecture. The input is 128 stacked frames. Orange color denotes 3D-convolution, red 3D-deconvolution (transposed convolution) and green average pool layers. At the bottom of each layer the time depth and after the layer-blocks the spatial resolution is shown. The output is the corresponding pulse-signal.

ELU activation function was used after each convolutional layer instead of ReLU because it resulted in slightly better performance. Furthermore, we used area based downsampling instead of linear for getting the 128x128 pixels resolution input, again because we observed better performance with this setting.

After every convolutional layers, batchnorm and ELU activation function is applied in this order. The spatio-temporal kernels (Depth, Height, Width) of the convolutions in the encoder part have a size of $[3,\ 3,\ 3]$ with stride 1 and padding 1, but the first convolution is different with a kernel size of $[1,\ 5,\ 5]$. In the temporal-decoder part the kernel size of deconvolution is $[4,\ 1,\ 1]$ with stride $[2,\ 1,\ 1]$ and padding $[1,\ 0,\ 0]$ which results in doubling the depth dimension. The input channel number is 3 (128 stacked RGB frames) which is increased to 32 and then to 64. The channel number remains 64 until the last convolution which outputs 1 channel. In the encoder part, the time dimension is squeezed to its quarter and in the decoder part it is stretched back to its

original size. The output is a vector containing the 1-dimensional pulse signal.

For training, the NegPeaLoss objective function was applied with ADAM optimizer and learning rate of 1e-4. The model was trained for 15 epochs as the validation loss did not improved after that. The batch size was set to 8. On another dataset, it was further trained with SNRLoss, then fused together with RateProbEst and trained together even further with image and frequency augmentations.

3.4.3 RateProbEst

This network tries to estimate the pulse-rate – with associated uncertainty – from pulse-signal. It is a 1-dimensional fully convolutional network, see Figure 3.3.

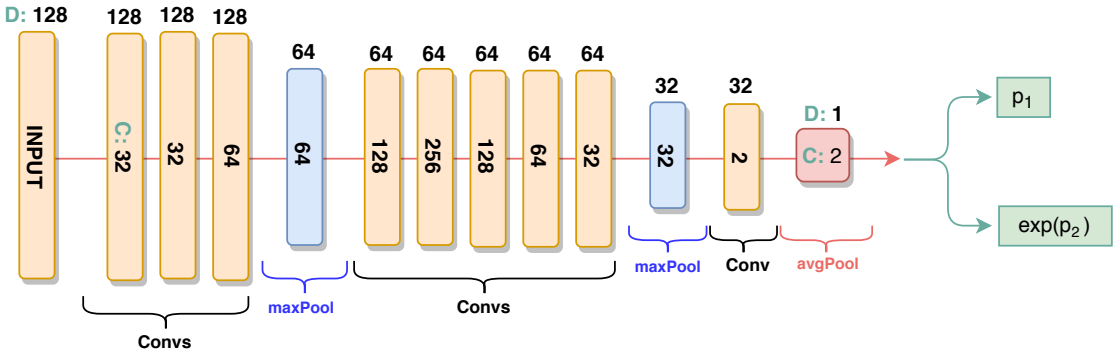

Figure 3.3: Illustration of the RateProbEst network. The time-depth (D) marked at the top and the number of channels (C) in the middle. The outputs are the parameters of the probability distribution.

It consists of 8 convolutional layers with kernel size of 17. Dropout is applied before each convolution. After convolutions, a batchnorm and ELU activation function follows. The channel number of the input is 1 (as it is a 1-dimensional pulse-signal). With the first convolution it is increased to 32, then doubled in every consecutive layer until the channel number reaches 256. Then it is decreased in the same way to 32. After the 3rd and 8th convolutional layer, a max pool layer is inserted with a kernel size of 5 and a stride of 2. Before the final average pool – that outputs the estimated probability distribution parameters – the output channels of the last (8th) convolutional layer are combined using another convolution with a kernel size and stride of 1 and output channel of 2. Finally, the second estimated parameter p_2 is inserted into an exponential function thus ensuring it is non-negative because in the loss function its logarithm will be calculated (i.e. this is the variance parameter of the distribution which must be positive).

For training, GaussLoss and LaplaceLoss was applied with ADAM optimizer and

learning rate of 1e-3. The target pulse-rate was converted to Hertz. All of the discussed image and frequency augmentations were applied. Typically, it is trained together with (an already pretrained) PhysNet. The learning took place over 60 epochs with batch size of 8.

3.5. Evaluation Process

Different metrics were computed in order to quantify the pulse-rate estimation performance of different methods, namely the followings:

- **MEA**: Mean Absolute Error between reference and pulse-rate estimate.
- **RMSE**: Root Mean Squared Error between reference and pulse-rate estimate.
- **R**: Pearson correlation coefficient of the reference and estimated pulse-rate in time.
- **MSNR**: Mean Signal-to-Noise Ratio of the estimated signal, where the reference frequency component is chosen to be "the signal" and the other components to be "the noise".

In the case of DeepPhys and PhysNet, the output is a signal but the metrics are pulse-rate based (except the last), therefore it is estimated from the signal via frequency analysis. First, the pulse-signal was filtered using a 6th order bandpass Butterworth filter with 80 BPM and 250 BPM cutoff frequencies. Then the Fourier transform of a sliding window of length 512 samples was computed with a stride of 1 second. A Hamming window was applied on the sliding window before the Fourier transform to avoid spectral leakage. The maximum power spectrum component is chosen to be the estimated pulse-rate.

The mathematical formulation of the MSNR metric is as follows:

$$\mathrm{MSNR} = \frac{1}{N}\sum_{k=1}^{N}\left\{10\log_{10}\left(\frac{S_k(f=f^*)}{\sum_{f\notin F^*}S_k(f)}\right)\right\} \tag{3.8}$$

where S_k is the power spectrum of the kth pulse-signal segment, N is the number of segments, f^* is the ground truth pulse-rate and $F^* = [f^* - \delta,\ f^* + \delta]$ where $\delta = 4\ BPM$. The sum in the denominator is over the pulse range (from 80 BPM to 240 BPM) except the reference frequency and its proximity – i.e., possible heart rate variability and small differences are not punished.

Chapter 4

Results

4.1. Neonate Dataset

Since at the time of writing this thesis we are not aware of the existence of a publicly available infant database for rPPG purposes, we started to construct our own.

Data were collected from a Neonatal Intensive Care Unit (NICU) in Hungary, namely the *Ist Dept. of Neonatology of Pediatrics, Dept. of Obstetrics and Gynecology, Semmelweis University, Budapest.* A total of 400 hours of data were collected (at the time of writing this study) from 5 infants, which occupies approximately 20 TB of storage space. From this large amount of recording, only a small fraction is usable for training and testing. The handling of the videos is done using tags with timestamps which were added manually to our database based on the consent decision of a group of three taggers. From these tags, we can filter categories which are appropriate for neural network training. There is approximately 15 hours of data where the following necessary conditions holds: illumination level is acceptable; there is no medical or parental intervention; there is a clean view of the baby (that is relatively calm); reference data is available. Besides that, there are 190 hours when a baby is visible, there is no intervention but also no reference; 120 hours with an empty incubator or dislocated view; 75 hours when the light conditions do not allow camera-based monitoring. Furthermore, there are other situations which occur on a regular basis, for them the following tags stands: excessive movement of baby, caring, blurred image, UV lamp is on, saturated image, feeding.

The adequate data was selected through tags and converted into a HDF5 data structure for model training. The conversion includes area based downscaling to decrease the size of the file and to average the camera noise – i.e. it averages subareas. The drawback of this is that it smooths the edges. We inspected the effect of the different interpolation

techniques on the training/inference, and the area based turned out to be the best option (except for the DeepPhys model, which uses bicubic interpolation). The pixel intensities of the frames are stored in 8 bits (UINT8). The information from the HDF5 files used for training and benchmarking are included in Table 4.1.

Type	**Length**	**Date code**	**Available refs.**	**Subject id.**	**Resolution**	**Bit depth**
benchmark	15 hours	200101	pulse-rate	B	128x128	8
train, test	5 hours	191111	pulse-rate	B	128x128	8
train, test	3 min.	190111	pulse-rate, pulse-sig	A	128x128	8

Table 4.1: Information about training and benchmark datasets. The Phillips reference monitors does not provide continuous pulse-signal data – which is indispensable in case of pulse-signal based training. The dataset in the last row (with date code 190111) was created using an other reference device which provided appropriate pulse-signal.

As we can see in the Table 4.1, there are two individual subjects (denoted with "A" and "B") among the datasets, but the longer training-set and the benchmark-set contains the same infant. In the future, it would be preferable to validate on an independent baby – which is not seen by the networks during training – for a more authentic evaluation.

The issue with the used Phillips patient monitors is that it sends data in irregular packets to the receiver and so much data is missing that it is almost unusable. Therefore, we were forced to work only with pulse-rate reference in case of these datasets.

4.2. Neural Network Performance

In this section, I will show the results regarding the performance of the neural networks evaluated on our challenging, 15-hour long benchmark dataset.

In general, the evaluation of methods is performed on selected, ideal data where the illumination conditions are appropriate and subject motion is minimized. In contrast, here the methods are evaluated on data that contains real situations – as our ultimate goal is to develop a long-term continuous monitoring system – for example: heavy motion, low illumination conditions, the head of neonate is covered with a blanket. The results presented here are the overall performance of the networks in all situations, therefore the metrics can be misleading, in-depth analysis and explanation is required for understanding. The overall results are shown in Table 4.2 which will be interpreted in the following subsections. I will shed more light on the details separately for each model and mode

of training. More details about the evaluation process can be found in Section 3.5. All corresponding source code is available on GitHub[1].

#	Arch.	Pretrained	Trained	Augm.	MAE [BPM]	RMSE [BPM]	Corr.	MSNR [dB]
1	DeepPhys	-	190111-signal	img	**27,00**	42,00	-0,0285	**-7,9**
2	PhysNet	-	190111-signal	-	17,00	27,00	0,0028	-6,6
3	PhysNet	-	190111-signal	img	37,00	49,00	0,055	-8,1
4	PhysNet		190111-signal	img, feq	28,00	**40,00**	-0,173	-8,58
5	PhysNet	190111	191111-hr	img, freq	8,62	14,37	0,487	**-3,2**
6	PhysNet	190111	191111-hr-crop	img	6,61	9,67	0,51	-1,25
7	PhysNet	190111	191111-hr-crop	img, freq	10,42	16,89	0,268	-3,55
8	RateProbEst	190111, 191111	191111-hr-crop	img, freq	7,26	10,15	**0,59**	-
9	RateProbEst	190111, 191111	191111-hr	img, freq	**7,22**	**10,023**	0,56	-

*Table 4.2: The overall performance of different methods. In the **Arch** column, the name of the model is written. The **Pretrained** column denotes whether the model was pretrained and if it was then on which dataset (contains the i.d. of dataset). The **Trained** column contains the dataset i.d. and reference type used for training. The **Augm.** column contains the applied augmentations. **MAE** is the Mean Absolute Error between reference and estimated pulse-rates, **RMSE** is the Root Mean Square Error between reference and estimated pulse-rates. In the **Corr.** column, the Pearson correlation coefficients can be found which describe how well the estimated and reference pulse-rate values correlate in time. It is not a reliable measure in our benchmark most of the time, because its value is mainly driven by the models behavior in situations where it couldn't predict the pulse-rate (e.g. it returns zeros or random noise or the average). In the last column, **MSNR** stands for Mean Signal-to-Noise Ratio, it is computed for the estimated signal, where the reference frequency component was chosen as the signal component and the other components as noise. Excluded results are denoted with red color, the explanation for exclusion is in the text. The results corresponding different training datasets are separated with vertical line. The best results are highlighted with bold font type in each section.*

4.2.1 DeepPhys

We can see from Table 4.2 that the performance of DeepPhys is comparable to PhysNet – which was trained on the same data (#2–4). It even outperforms *PhysNet#3* and *PhysNet#4* which are the reliable versions (more details about what this "reliability"

[1]https://github.com/terbed/Deep-rPPG/tree/v1-thesis

means can be found in the subsequent section). This model was trained on a really small dataset (3 min.). The disadvantage of this network is that it demands preprocessing steps like the construction of normalized frame differences (as one of the network input is this). Because of the specific input it is also harder to train, but the main drawback comes with the fact that it has got single point output which renders it impossible to train it with custom loss functions specifically developed for signals like NegPeaLoss or NegSNRLoss (see in Section 3.2). Another big difference compared to the PhysNet architecture is that it is a 2 dimensional convolutional network, thus cannot exploit temporal features. However, we have to note that compared to the small training-set it performs still fairly well on totally unknown data (see Figure 4.1 and slightly Figure 4.2) and outperforms PhysNet.

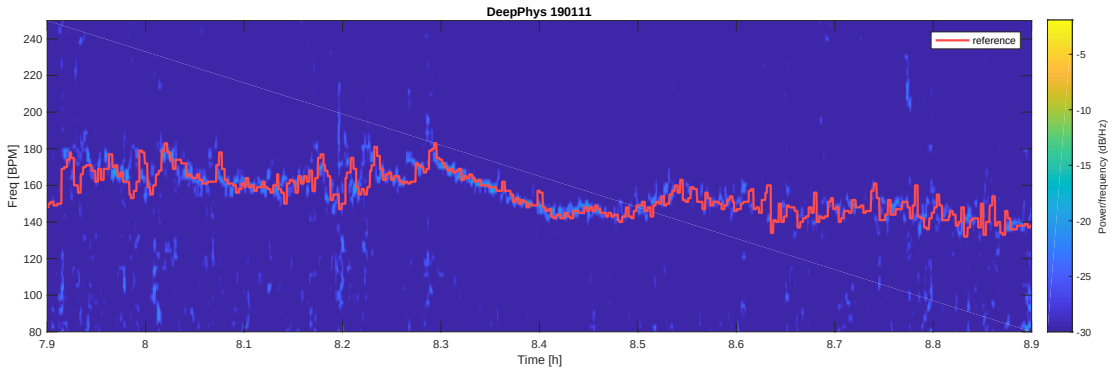

Figure 4.1: The spectrogram of DeepPhys model output signal and reference pulse-rate values on a selected 1 hour interval. The right frequency component of the signal is weak but follows the reference in majority.

Compared to PhysNet, DeepPhys has a lightweight architecture and runs one order of magnitude faster (see Section 4.3).

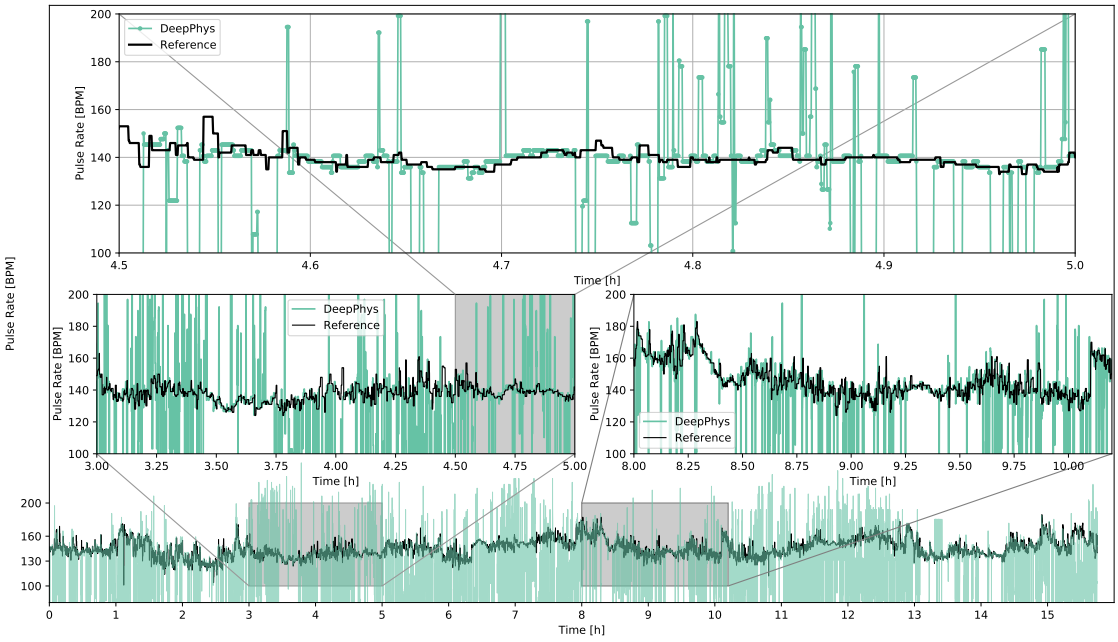

Figure 4.2: The calculated rate estimates from the output signal of DeepPhys network and corresponding reference values along the whole benchmark dataset (with enlarged selected segments).

4.2.2 PhysNet trained on pulse-signal

The corresponding results in Table 4.2 are the rows 2-4. Here the PhysNet model was trained on pulse-signal reference on the small dataset with NegPeaLoss. Based on the table mentioned before, seemingly the model without augmentations outperforms the others. What is misleading here is that this model learned to give the average pulse-rate in cases when it couldn't predict the true value (see in Figure 4.3) – which is an undesired behaviour.

On the contrary, the model trained with augmentations does not return the average pulse-rate but shoots (more or less) randomly above or below (see in Figure 4.4). This is the reason why *PhysNet#2* seems to outperform *PhysNet#4* (see in Table 4.2), but in fact the latter is more preferable.

It should be emphasized that only a very limited amount of training data was available so far, but it was enough to train the models to be able to extract pulse-signal from an independent video which is still impressive.

The network's ability to generalize greatly benefits from augmentations with which we can simulate different illumination conditions and posture. These are the static image augmentations, but another important aspect is the temporal feature of the signal. Most of the time, we can observe the neonate pulse-rate in the range of 120 BPM to 170 BPM,

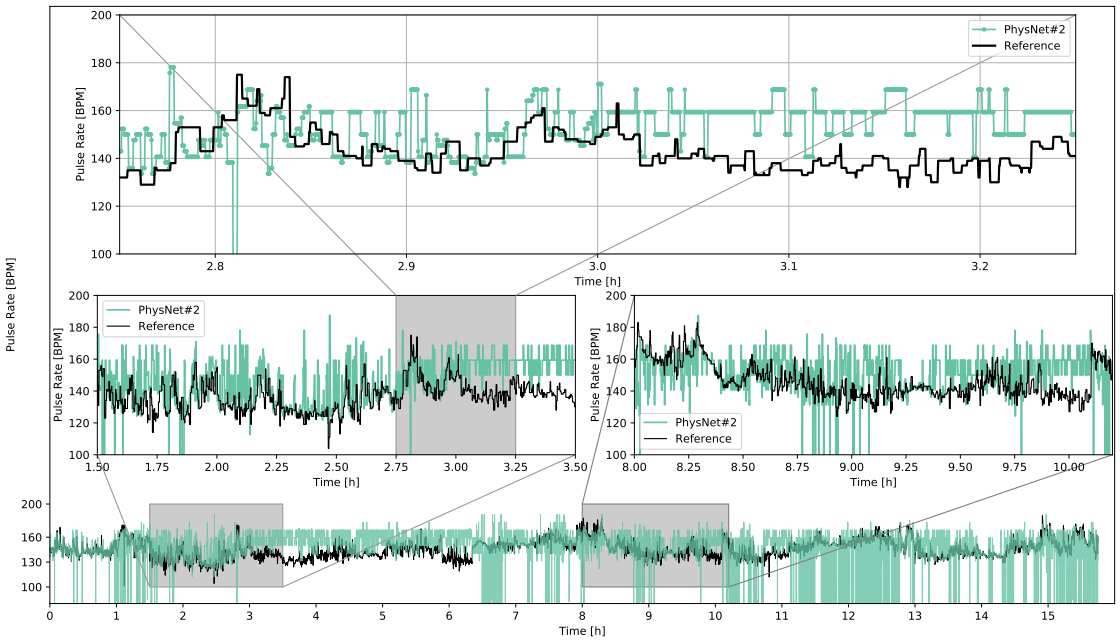

Figure 4.3: The estimated pulse-rates from PhysNet model's output signal and the corresponding reference pulse-rate values on the whole benchmark. This model is trained without augmentations and learned to give the average pulse-rate value.

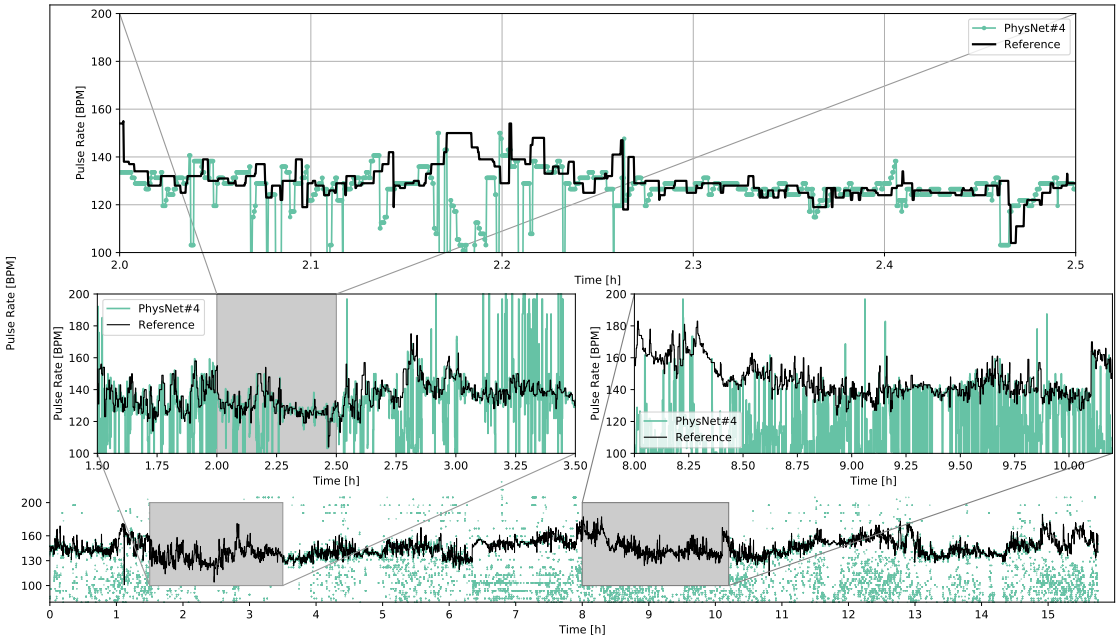

Figure 4.4: The estimated pulse-rates from PhysNet model's output signal and the corresponding reference pulse-rate values on the whole benchmark. This model is trained with augmentations, which prevents the model to return the average pulse-rate.

but sometimes it falls out of this range. This means that we have naturally biased data which could have negative effect on the model prediction in marginal cases – which in

turn would be the most important situations to indicate (too low or high pulse-rate). Therefore, we applied frequency augmentation to have a uniform distribution of pulse-rate values in the training data. The application of such augmentation improves the overall performance of the network (see Figure 4.5).

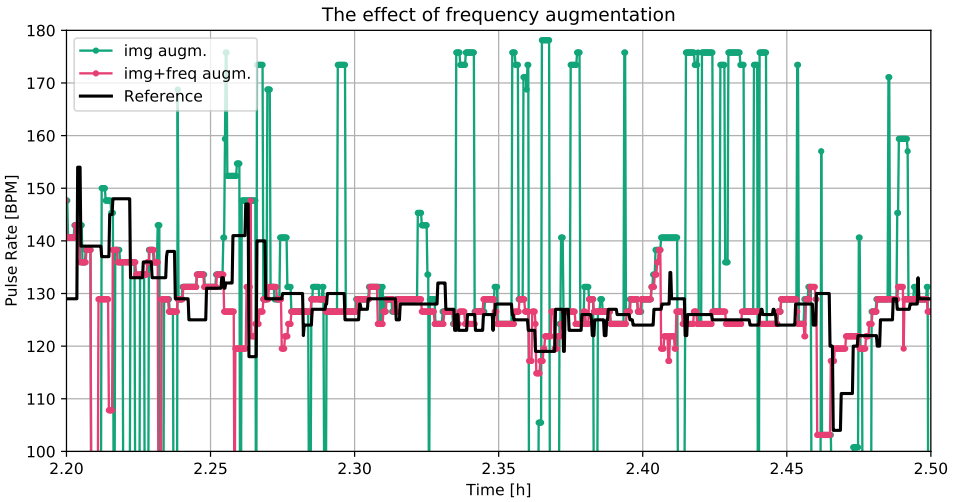

Figure 4.5: The estimated pulse-rates from PhysNet model's output signal – in case of image augmentation and additional frequency augmentation – and the corresponding reference pulse-rate values on a selected time segment. The training of the model benefits a lot from frequency augmentation.

4.2.3 PhysNet trained on pulse-rate

From now on, the PhysNet network is further trained on a large dataset (1-3 hours) using the numeric pulse-rate reference with NegSNRLoss. The corresponding results in Table 4.2 are the rows from 5 to 7. A large leap can bee seen in the performance compared to the previous models, this can be attributed partly to the orders of magnitude larger training dataset and to the fact that this training-set contains the subject also used in the benchmark. On the other hand, NegSNRLoss tends to maximize the spectral energy of the signal into the right frequency band – and cleans other frequencies not corresponding to pulse – (see Figure 4.6) that is in line with the post-processing (where we chose the maximum frequency component as the estimated pulse-rate). This is also evident from the *MSRN* metric found in the result table: using this loss function, the *MSNR* increased from -8.58 dB to -3.55 dB.

The difference between *PhysNet#5* and the following two networks is that the input of the former is not cropped. To determine the RoI to be cropped a YOLO [31] object

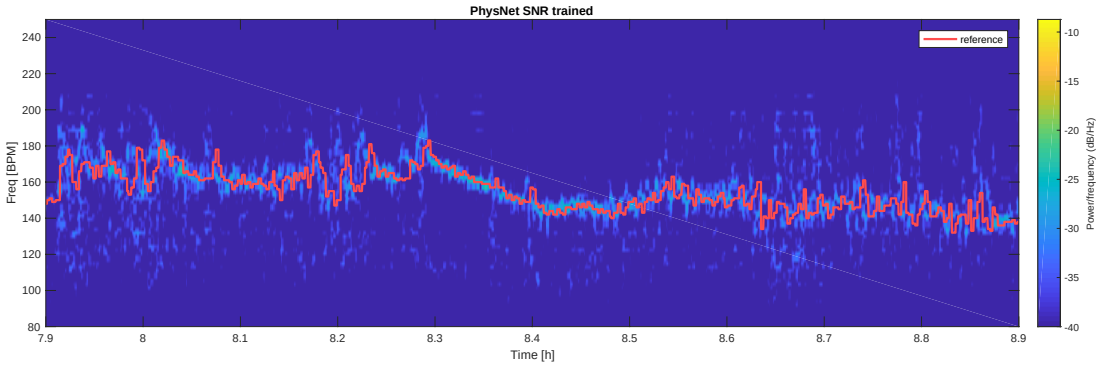

Figure 4.6: The spectrogram of PhysNet model output signal and the corresponding reference pulse-rate values on a selected 1 hour interval. Training with NegSNRLoss tends to maximize the signal energy in the right frequency band.

detector network was used. Our assumption was that cropping the torso and head of the infant – and so discarding the non-skin parts – would improve the performance – as we focusing the input of the network on the region of interest. Surprisingly, we can read the exact opposite from the table: the mean absolute error for the non-cropped version is 8.62 BPM and 10.42 BPM for the other version. The reason behind it might be the fact that the constructed datasets were already downscaled to 128x128 resolution, consequently the downscaled image was cropped and then resized back to the input resolution of 128x128. We can anticipate information gain only when we crop on the original image (500x500) and then downscale that. Nonetheless, we expected an increase in performance even so because what we did functions like an attention mechanism (without additional information gain only pruning the irrelevant features). The explanation for the worse performance might be that the aspect ratio is changing between crops and resizing back to square resolution lengthen the image. These unstable input features might decrease the performance of the network. The solution for this issue might be: (1) to crop on the original (500x500) image; (2) and then pad to square before downscaling it to 128x128.

Regarding augmentations, the situation is the same as it was in the previous section. Based on Table 4.2 *PhysNet#6* – which was trained only with image augmentations – with a mean absolute error of 6.61 BPM seems to outperform the others – where additional frequency augmentation was also used – with a mean absolute error of 8.62 BPM and 10.42 BPM. This network performs well close around the average pulse-rate value but fails further from it (see Figure 4.7). Yielding the average pulse-rate if the estimation fails is a good strategy to perform well in distance metrics (but undesirable behaviour for medical use-cases). In contrast, if we apply frequency augmentation, the

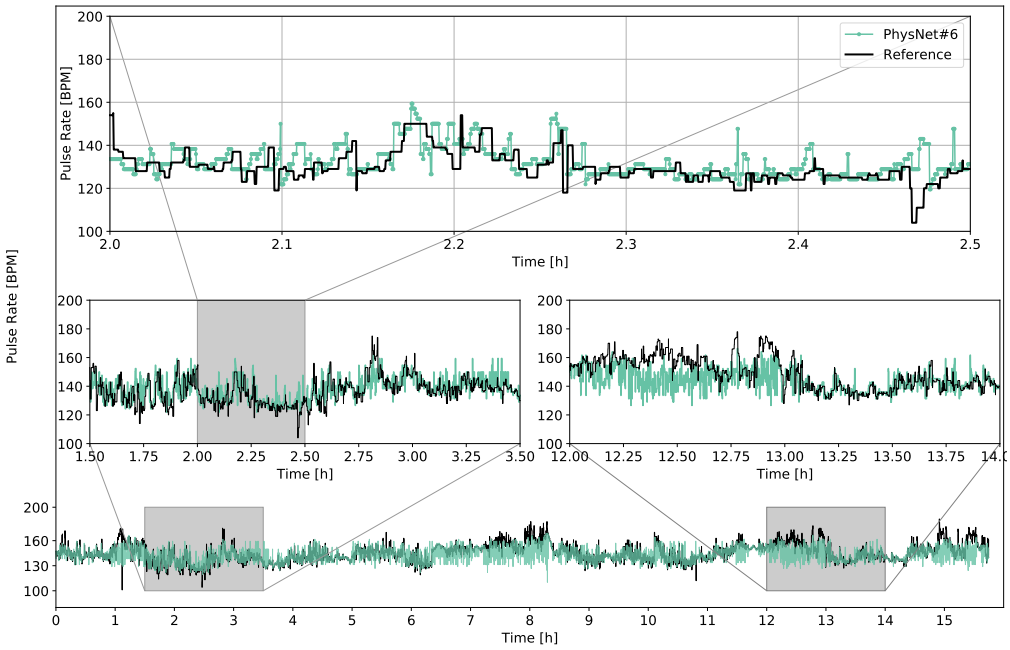

Figure 4.7: The calculated rate estimates from the output signal of PhysNet network and the corresponding reference. This network was trained only with image augmentations and performs well around the average pulse-rate, but unable to capture marginal cases (see 12-13 hours) – which is undesirable.

network is not biased towards the average value but shoots above or below "randomly" on the whole range if estimation fails (see Figure 4.8) – resulting in larger distance errors compared to the previous case. All things considered, frequency augmentation happens to be beneficial again.

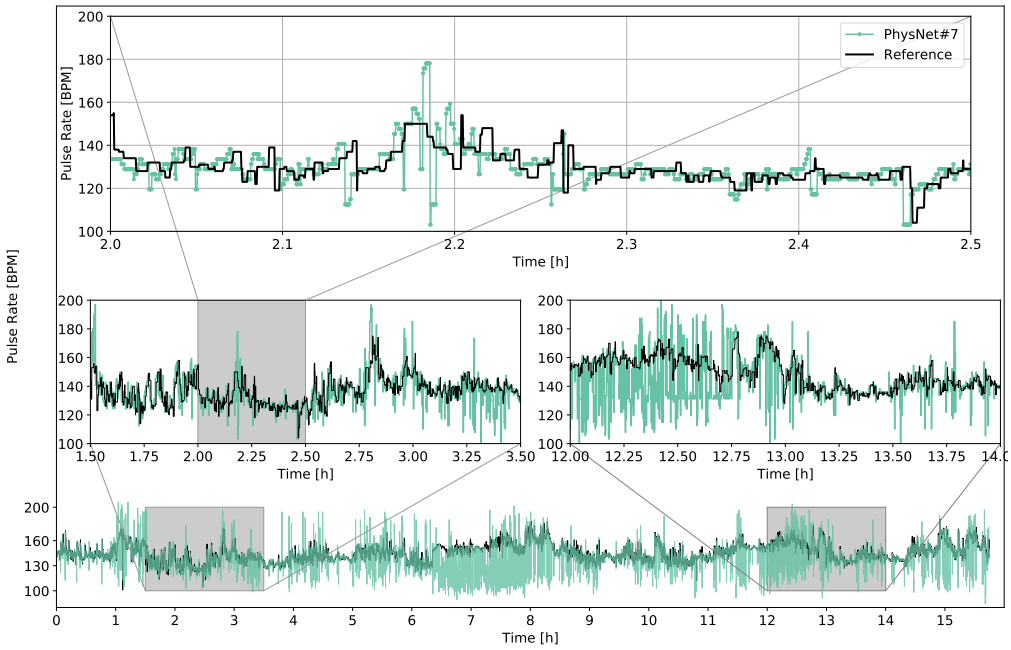

Figure 4.8: The calculated rate estimates from the output signal of PhysNet network and the corresponding reference. This network was trained both with image and frequency augmentations and it is more stable for marginal cases.

4.2.4 RateProbEst

This section inspects the results corresponding the rate probability estimator network which input is the output pulse-signal of the PhysNet network. More precisely, these results are corresponding to the ensemble of: (1) PhysNet which is responsible for pulse-signal extraction from video; (2) and RateProbEst which is responsible to estimate a pulse-rate – with a corresponding confidence (or quality) index – from the output of PhysNet. These networks were fused and trained together.

The associated results are the 8th and 9th rows in Table 4.2. The difference between the two alternatives is that the first used a cropped input while the other used the original video. The considerations behind using a cropped input and the interpretation of the performance difference between them is the same as what was detailed in the previous section. From now on, the not cropped versions will be used for fair comparison and illustration.

Based on the results, using a rate estimator network after the signal extractor network is beneficial as the mean absolute error decreased from 8.62 BPM to 7.22 BPM and the correlation increased from 0.487 to 0.56. In addition, using a probabilistic output layer we can also predict the uncertainty of the estimate shown on Figure 4.9.

In medical applications, it is important to know how reliable the estimated value is,

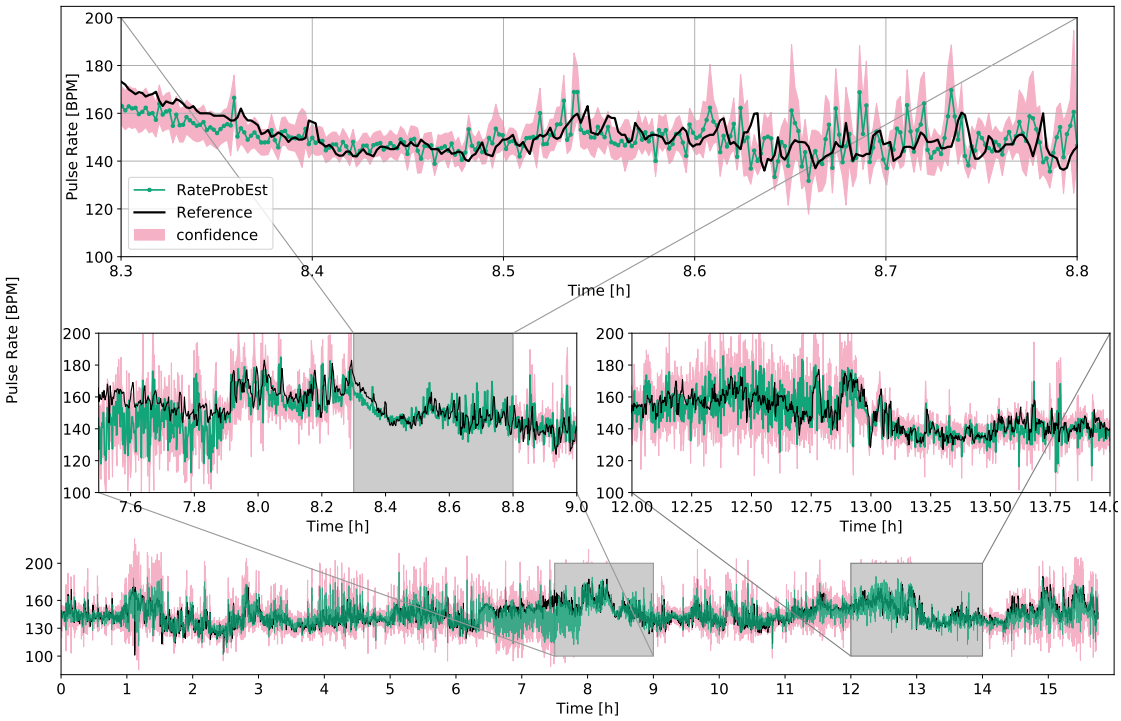

Figure 4.9: The output of the RateProbEst network and the corresponding reference. This network was fused and trained together with PhysNet. With accurate estimation, the confidence interval decreases and with inaccurate estimation just on the other way around. At some points, the method fails and mishits with large confidence – the true value is outside the confidence interval (red) in this case.

which is possible with Bayesian Neural Networks. The introduced RateProbEst network is the most simple variant: it is an ordinary fully convolutional neural network which outputs the parameters of the probability distribution function. We can see on Figure 4.9 that it is more or less functional – accurate estimation has small confidence interval and inaccurate estimation has large confidence interval, thus the true value is within the confidence interval most of the time. However, there are some points where the method fails and the true value is outside of the confidence interval. The solution for this problem might be to apply more advanced Bayesian Neural Network techniques.

4.2.5 Comparison with classical algorithm

In order to compare the performance of neural networks with classical algorithmic approaches, a pilot study was performed. One of the most suitable state-of-the-art methods for continuous long-term monitoring was implemented, namely the *Full Video Pulse extraction* algorithm [6] (FVP) with the *Plane Orthogonal to Skin-tone* [6] (POS) core rPPG algorithm. It has been already applied successfully on neonates in NICU [6]. This

method eliminates the steps of RoI initialization, detection and tracking. It performs well in case of static background (because it uses overall color statistics that would be corrupted by any change in the background) which holds in case of an incubator. We applied this algorithm on a 5 hour long dataset without success: it could not estimate the pulse along the whole video. Our dataset was not appropriate for this algorithm. We observed a systematic background noise which could distract the method[2]. Therefore, we developed our own simple method that finds the skin regions based on colors, averages subareas and then applies the POS algorithm to extract the pulse-signal from the color channels. We applied this method on our data and it could estimate the pulse-rate – i.e. the estimate was within 6 BPM distance from the reference – for a total 95 minutes from 5 hours. We also employed the PhysNet network – which was trained with NegSNR-Loss on different data – and calculated the spectrogram of the output pulse-signal. The dominant frequency component of the spectrogram followed the reference throughout the 5 hour long data. One half-hour chunk of the spectrogram together with the reference signal and the estimate of the classical algorithm[3] is depicted on Figure 4.10.

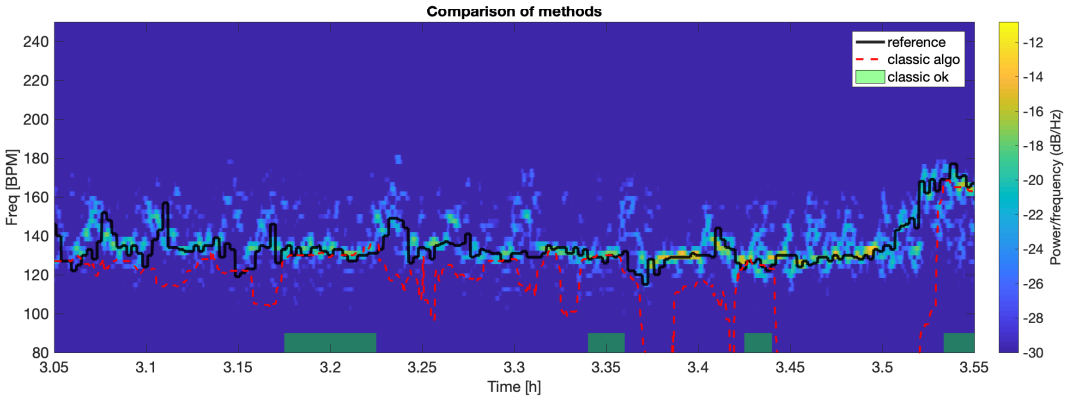

Figure 4.10: A half-hour chunk of the spectrogram of the output pulse-signal of PhysNet along with the reference (black line) and the estimate of the classical algorithm (red dashed line). The segments where the classical algorithm could predict the true value (continuously for at least 1 minute) are denoted with green color above the time axe. The dominant frequency of the spectrogram follows the reference.

From this, we conclude that the deep learning approach is more stable on our data. In contrast to the study of Wang et al. [7], the FVP algorithm was not able to extract pulse from our videos at all. The other classic method which also utilizes the POS

[2]Note that at this time the YOLO network was not yet applicable, which could solve this issue by selecting the RoI.

[3]A one minute moving average was also applied on the estimate as a post-processing step.

algorithm was able to estimate the pulse-rate in some sections, but the PhysNet network outperformed it.

The explanation for its observed superior performance compared to classical methods may be that: (1) it can utilize also BCG information; (2) it can select more sophisticated RoI; (3) it was optimized on the given subject. The first two assumptions require further confirmations.

4.3. Resource Limited Environment

If our goal is to develop a system that is able to do predictions in real-time and also cost efficient, it is really important to inspect its running time on an average hardware. Our choice was the *NVIDIA Jetson Nano* which is an embedded system with a 128-core Maxwell GPU, Quad-core ARM A57 @ 1.43 GHz CPU and 4 GB 64-bit LPDDR4 25.6 GB/s memory (that is shared between CPU and GPU). This device is developed specifically for deep learning applications and it is in the low price range (among the similar other options): at the time of writing this thesis, it costs around 100 dollars. For comparison, the *Raspberry Pi model 4* with a Quad core ARM A72 @ 1.5GHz CPU and 4GB LPDDR4-3200 SDRAM memory – but without GPU – costs around 70 dollars. Additionally, the inference time was also measured on a 2.6 GHz Quad-Core Intel Core i7 (2012) CPU.

All the previously applied networks were evaluated regarding the inference speed. Multiple inputs can be computed in parallel with stacking them into the (first) batch dimension. Utilizing this, we can input 128 frames to the DeepPhys network in order the have a fair comparison with PhysNet.

The method of evaluation was to run the networks – and measure its running time – in inference mode repeatedly 50 times, then calculate the mean and standard deviation statistics discarding the first 10 instances which counts as warm-up rounds. The results are included in Table 4.3. The networks and their inference was implemented in Python using the PyTorch framework. The script is available on GitHub[4].

In average, the inference time was five times faster on the GPU of *NVIDIA Jetson Nano* than on the CPU of the personal computer. The larger the network was, the faster the computation becomes on GPU compared to CPU. In the followings, we will analyse the results regarding the *NVIDIA Jetson Nano* (GPU).

[4]https://github.com/terbed/Deep-rPPG/tree/v1-thesis/nano

Network	Input shape	Device	Duration [ms]	Std. [ms]
YOLO	[1, 3, 128, 128]	CPU	228,90	7,85
YOLO	[1, 3, 128, 128]	GPU	71,27	1,43
YOLO	[1, 3, 416, 416]	GPU	428,99	1,34
DeepPhys	[128, 3, 36, 36] x 2	CPU	754,56	12,49
DeepPhys	[128, 3, 36, 36] x 2	GPU	186,81	1,35
PhysNet	[1, 3, 128, 128, 128]	CPU	10930,14	1298,24
PhysNet	[1, 3, 128, 128, 128]	GPU	1877,25	6,87
RateProbEst	[1, 1, 128]	CPU	9,98	0,85
RateProbEst	[1, 1, 128]	GPU	9,09	0,24
RateProbEst	[10, 1, 128]	GPU	14,81	0,19
PhysNet+RateEst	[1, 3, 128, 128, 128]	CPU	10539,16	1036,84
PhysNet+RateEst	[1, 3, 128, 128, 128]	GPU	1884,94	8,40
PhysNet+RateEst	[3, 3, 128, 128, 128]	GPU	5674,59	37,79

Table 4.3: The results regarding the execution time of the networks. In the first column, the name of the networks are written. In the second column, the shape of the network input can be seen. The first dimension is always the batch dimension and the second is always the number of channels, the third is the time depth (if there is such) and the last two is the height and width (except RateProbEst). The following column (named as "Device") denotes the device on which the network was ran. The last two columns are the average inference time and the standard deviation of it.

The YOLO [31] network is really fast with input image resolution of 128x128 the required time was 71.27 ms in average (see in Table 4.3). If we increase the input resolution to 416x416, one prediction takes approximately 0.5 seconds. This network is used to localize the neonate on every 128th (6.4 seconds at 20 FPS) frame and the resulting bounding box is fixed between the updates. It is typically sufficient because newborns cannot change their posture by themselves – so they stay within the bounding box in that short period of time. Consequently for our application, the speed of this network is more than sufficient.

In case of the DeepPhys network, we stack as many images into the batch dimension as the temporal depth of the PhysNet network, in this way both networks calculate the estimate for a 6.4 second long video segment – which enables us to compare them. As we

can see from Table 4.3, DeepPhys is approximately 10 times faster than PhysNet which is not surprising. First of all, DeepPhys requires smaller input image resolution even if we take into account the two input streams. Furthermore, it is a 2D CNN network which has got a lower computational complexity compared to a 3D CNN. And finally but not least, the frames stacked into the batch are computed in parallel, thus it is really efficient.

The DeepPhys network is really fast as it requires 186.81 ms to compute a 6.4 second long video segment. The PhysNet network needs approximately 1.9 seconds for it, which is still sufficient for a real-time application (i.e. the pulse-rate can be updated in every 2 seconds).

The RateProbEst network is a relatively small 1D CNN which requires a negligible 9.09 ms of time in average to calculate the pulse-rate estimate from the pulse-signal. The sequential application of the extractor and estimator network was also tested: as expected, the running times add up. Finally, it is also inspected how many times this ensemble of networks fit into the GPU memory. For this question, the answer is three times. (see last row of Table 4.3).

From all of these results, we can conclude that the neural networks used throughout this study are capable to function in real-time in a resource limited environment (embedded system) similar to the *NVIDIA Jetson Nano*.

Chapter 5

Summary

5.1. Conclusions

In this section, the main conclusions and take-home messages of this study are briefly summarized.

On our limited amount of data, DeepPhys slightly outperformed[1] PhysNet (in accordance with results in [5]). DL and classical algorithmic approaches are compared, the results suggest that DL methods have superior performance but further detailed verification is needed. With the proposed frequency augmentation (and with general image augmentations), the generalization capability of PhysNet and RateProbEst could be largely improved. Without augmentations, the DNNs tend to predict the average pulse-rate. If there is only the pulse-rate reference available, it is still possible to train PhysNet with SNRLoss, but it is more beneficial to fuse it with a rate calculator network (in this study with RateProbEst) and train them together as the results suggest. It is possible to associate the prediction with a quality index (which describes the accuracy of the estimation) with the application of the probabilistic output layer (without significantly modifying standard DNN architectures). The corresponding results are promising but the reliability could be improved with more advanced Bayesian Neural Network techniques. Finally, it is shown that the inference speed of the presented DNNs enables real-time functioning in an NVIDIA Jetson Nano system, and the DeepPhys network is around one order of magnitude faster than PhysNet.

[1] Based on the MAE and MSNR metrics.

5.2. Objectives and Achievements

The objectives of this thesis and the achievements can be summarized in seven major points:

1. The revision of existing rPPG methods and corresponding literature.

   During this study, extensive research has been done on both classical algorithmic and deep learning approaches which are summarized in Chapter 2.
   /personal work/

2. The selection and implementation of one or more neural networks used for rPPG-signal extraction.

   Two different state-of-the-art neural networks have been selected and implemented, namely the DeepPhys [4] and PhysNet [5] architectures. In addition, a rate extractor network (RateProbEst) has been developed based on other studies found in literature [9]. Details about these are included in Chapter 3.
   /personal work/

3. The creation of an rPPG training and benchmark dataset for the special case of neonates.

   Using the database tools which were developed by multiple members from our research group (including myself), I was able to construct appropriate training and benchmark dataset for my study (see "Neonate Dataset" section in Chapter 4).
   /group work/

4. The evaluation of the implemented networks on the created datasets.

   The networks have been evaluated on the constructed benchmark dataset and different video augmentation techniques were analyzed (see Chapter 4). A new kind of augmentation technique has been proposed, namely frequency augmentation.
   /personal work/

5. The comparison of the results (of neural networks) with other (algorithmic) methods.

   A pilot study has been designed to compare the performance of deep learning approach with algorithmic methods which can be found in the "Comparison with classical algorithms" section in Chapter 4.
   /personal work/

6. The proposal of a technique that is able to provide a quality index that measures the reliability of the prediction.

   For this problem, the field of probabilistic neural networks has been proposed (see in Chapter 2) which allows us to get information also about the uncertainty of the estimation. This idea has been applied in practice – for the first time in the field of rPPG – on our data (see RateProbEst in Chapter 4).
   /personal work/

7. The examination of the applicability of neural networks on a resource-limited environment (embedded system).

   All of the neural networks used in this study has been tested on a *NVIDIA Jetson Nano* embedded system (see in Chapter 4).
   /personal work/

5.3. Future directions

In the future, a higher quality database could be created with an increased subject number – and where also pulse-signal reference is available – which is essential for DeepPhys and PhysNet training. In this study, we were largely constrained by the only available pulse-rate reference and the pulse-signal label was missing most of the time. Furthermore, a differentiated, scenario-specific benchmark dataset could be built with which we could test not only the average performance of the methods but their performance for each task (e.g.: the performance of methods in mild, medium/heavy subject motion, in case of adequate or inadequate illumination, etc.).

In this preliminary study, we showed that it is possible to apply probabilistic neural networks to make predictions with a corresponding certainty value – without significantly changing the standard architecture, only by applying a probabilistic output layer that predicts the parameters of a chosen probability distribution. For this, there are more advanced methods where all of the single model weights represent a distribution and at each forward pass they are sampled from the learned distribution, i.e. at each forward pass, the model is differently parameterized. Thus, if we run inference on the same input multiple times the output will not be exactly the same, calculating the statistics (e.g., mean and standard deviation) of the outputs yields the overall prediction and the accuracy of the prediction (which is very similar to ensemble networks). After all, these methods can only model uncertainties coming from external noise (e.g. motion, low

illumination, camera noise, etc.) but cannot handle model errors on the other hand which comes from the imperfection of the training dataset. When the input is unfamiliar to the trained neural network, its output will be still unreliable, fortunately, there exist also such techniques with which we can model this kind of uncertainty. For example, letting the (specifically installed) dropout layers turned on in test time will act as a Bayesian approximation [30], [32]. Using this technique, the so-called Bayesian Neural Network (BNN) will indicate if the kind of input sample is not seen before. This property is really important also for online learning (or even generally for neural network training) because the data samples which carry lots of (so far unknown) information can be selected.

These methods could be effectively applied not exclusively on this task (rPPG) but on every medical application where a prediction uncertainty measure would be beneficial.

Acknowledgements

First of all, I would like to thank my supervisor for the guidance in my research and for the supportive environment he provided. Also thanks for those members in our research group in SZTAKI who somehow contributed to my work.

Finally, thanks for the support of the Hungarian grant for supporting the development of new Reanimation Table and intelligent monitoring techniques for Newborn Infants (VEKOP-2.2.1-16-2017-00002).

Bibliography

[1] W. Verkruysse, L. O. Svaasand, and J. S. Nelson, "Remote plethysmographic imaging using ambient light.," *Optics express*, vol. 16, no. 26, pp. 21 434–21 445, 2008.

[2] K. Anand and F. M. Scalzo, "Can adverse neonatal experiences alter brain development and subsequent behavior?" *Neonatology*, vol. 77, no. 2, pp. 69–82, 2000.

[3] S. Henry, M.-A. Richard-Yris, S. Tordjman, and M. Hausberger, "Neonatal handling affects durably bonding and social development," *PloS one*, vol. 4, no. 4, 2009.

[4] W. Chen and D. McDuff, "Deepphys: Video-based physiological measurement using convolutional attention networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 349–365.

[5] Z. Yu, X. Li, and G. Zhao, "Remote photoplethysmograph signal measurement from facial videos using spatio-temporal networks," in *Proc. BMVC*, 2019, pp. 1–12.

[6] W. Wang, A. C. den Brinker, S. Stuijk, and G. de Haan, "Algorithmic principles of remote ppg," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 7, pp. 1479–1491, 2016.

[7] W. Wang, A. C. den Brinker, and G. De Haan, "Full video pulse extraction," *Biomedical Optics Express*, vol. 9, no. 8, pp. 3898–3914, 2018.

[8] T. Vogels, M. van Gastel, W. Wang, and G. de Haan, "Fully-automatic camera-based pulse-oximetry during sleep," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1349–1357.

[9] R. Špetlık, V. Franc, and J. Matas, "Visual heart rate estimation with convolutional neural network," in *Proceedings of the British Machine Vision Conference, Newcastle, UK*, 2018, pp. 3–6.

[10] G. De Haan and V. Jeanne, "Robust pulse rate from chrominance-based rppg," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 10, pp. 2878–2886, 2013.

[11] L. F. C. Martinez, G. Paez, and M. Strojnik, "Optimal wavelength selection for non-contact reflection photoplethysmography," in *22nd Congress of the International Commission for Optics: Light for the Development of the World*, International Society for Optics and Photonics, vol. 8011, 2011, p. 801 191.

[12] A. A. Kamshilin, E. Nippolainen, I. S. Sidorov, P. V. Vasilev, N. P. Erofeev, N. P. Podolian, and R. V. Romashko, "A new look at the essence of the imaging photoplethysmography," *Scientific reports*, vol. 5, p. 10 494, 2015.

[13] A. A. Kamshilin and N. B. Margaryants, "Origin of photoplethysmographic waveform at green light," *Physics Procedia*, vol. 86, pp. 72–80, 2017.

[14] A. V. Moço, S. Stuijk, and G. de Haan, "New insights into the origin of remote ppg signals in visible light and infrared," *Scientific reports*, vol. 8, no. 1, pp. 1–15, 2018.

[15] M. Lewandowska, J. Rumiński, T. Kocejko, and J. Nowak, "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)*, IEEE, 2011, pp. 405–410.

[16] M.-Z. Poh, D. J. McDuff, and R. W. Picard, "Advancements in noncontact, multiparameter physiological measurements using a webcam," *IEEE transactions on biomedical engineering*, vol. 58, no. 1, pp. 7–11, 2010.

[17] G. De Haan and A. Van Leest, "Improved motion robustness of remote-ppg by using the blood volume pulse signature," *Physiological measurement*, vol. 35, no. 9, p. 1913, 2014.

[18] W. Wang, S. Stuijk, and G. De Haan, "Unsupervised subject detection via remote ppg," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 11, pp. 2629–2637, 2015.

[19] G.-S. Hsu, A. Ambikapathi, and M.-S. Chen, "Deep learning with time-frequency representation for pulse estimation from facial videos," in *2017 IEEE International Joint Conference on Biometrics (IJCB)*, IEEE, 2017, pp. 383–389.

[20] X. Niu, H. Han, S. Shan, and X. Chen, "Synrhythm: Learning a deep heart rate estimator from general to specific," in *2018 24th International Conference on Pattern Recognition (ICPR)*, IEEE, 2018, pp. 3580–3585.

[21] Q. Zhan, W. Wang, and G. de Haan, "Analysis of cnn-based remote-ppg to understand limitations and sensitivities," *Biomedical Optics Express*, vol. 11, no. 3, pp. 1268–1283, 2020.

[22] L. Scalise, N. Bernacchia, I. Ercoli, and P. Marchionni, "Heart rate measurement in neonatal patients using a webcamera," in *2012 IEEE International Symposium on Medical Measurements and Applications Proceedings*, IEEE, 2012, pp. 1–4.

[23] L. A. Aarts, V. Jeanne, J. P. Cleary, C. Lieber, J. S. Nelson, S. B. Oetomo, and W. Verkruysse, "Non-contact heart rate monitoring utilizing camera photoplethysmography in the neonatal intensive care unit—a pilot study," *Early human development*, vol. 89, no. 12, pp. 943–948, 2013.

[24] M. Villarroel, A. Guazzi, J. Jorge, S. Davis, P. Watkinson, G. Green, A. Shenvi, K. McCormick, and L. Tarassenko, "Continuous non-contact vital sign monitoring in neonatal intensive care unit," *Healthcare technology letters*, vol. 1, no. 3, pp. 87–91, 2014.

[25] M. van Gastel, B. Balmaekers, S. B. Oetomo, and W. Verkruysse, "Near-continuous non-contact cardiac pulse monitoring in a neonatal intensive care unit in near darkness," in *Optical Diagnostics and Sensing XVIII: Toward Point-of-Care Diagnostics*, International Society for Optics and Photonics, vol. 10501, 2018, p. 1 050 114.

[26] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" In *Advances in neural information processing systems*, 2017, pp. 5574–5584.

[27] J. Gast and S. Roth, "Lightweight probabilistic deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3369–3378.

[28] A. Loquercio, M. Segu, and D. Scaramuzza, "A general framework for uncertainty estimation in deep learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3153–3160, 2020.

[29] L. V. Jospin, W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun, "Hands-on bayesian neural networks–a tutorial for deep learning users," *arXiv preprint arXiv:2007.06823*, 2020.

[30] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, 2016, pp. 1050–1059.

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[32] I. Osband, "Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout," in *NIPS Workshop on Bayesian Deep Learning*, vol. 192, 2016.