Gated Recurrent Unit-Based RNN for Remote Photoplethysmography Signal Segmentation

Rita Meziati Sabour
Univ. Bourgogne Franche-Comté
Dijon, France
rita_meziati-sabour@etu.u-bourgogne.fr

Yannick Benezeth
ImViA Laboratory, Univ. Bourgogne Franche-Comté
Dijon, France
yannick.benezeth@u-bourgogne.fr

Abstract

Remote Photoplethysmography (rPPG) enables quantifying blood volume variations in the skin tissues from an input video recording, using a regular RGB camera. Obtained pulse signals often contain noisy portions due to motion, leading researchers to put aside a great number of rPPG signals in their studies. In this paper, an approach using a Gated Recurrent Unit-based neural network model in order to identify reliable portions in rPPG signals is proposed. This is done by classifying rPPG signal samples into reliable and unreliable samples. For this purpose, rPPG and electrocardiography signals (ECG) were collected from 11 participants, rPPG signal samples were labeled (ECG was used as ground truth), and data were augmented to reach a total number of 11000 *1-minute-long rPPG signals. We developed a model composed of a unidimensional CNN and a Bidirectional GRU (1D-CNN+B-GRU) for this study, and obtained an accuracy rate of* 85.88%.

1. Introduction

Blood volume pulse (BVP) signals describe blood volume changes in vessels over time. They can be measured using an optical technique called *photoplethysmography* (PPG), whose principle is to expose a skin surface to a light source, and to quantify the light reflected or transmitted by the skin tissue. Changes of the light quantity reflects blood volume variations that are provoked by the cardiac activity. PPG was originally integrated in pulse oximeters, and was recently generalized to non-medical applications (sport [46], driving [20], daily activities [39], etc) and was embedded in several supports such as ear sensors, wristbands or mobile phones [31, 35, 42].

Remote photoplethysmography (rPPG) is a completely non-invasive version of reflexive PPG: ambient light constitutes the light source and a camera photosensitive matrix plays the role of the receptor. rPPG allows to follow the quantity of light reflected by a skin surface (usually the face) filmed with a regular camera over time. Indeed, reflected light depends on the continuous blood volume variations within the skin tissues, which lead to subtle changes in the skin color over time [42, 44].

The classic algorithmic chain for rPPG signal extraction follows several image and signal processing steps. First, the *region of interest* (ROI) is detected and located in the input video images [33, 34]. Then, skin pixels are selected, and RGB values are spatially averaged over the ROI [3, 18, 25, 45]. Next, RGB average values are concatenated and form three temporal traces that can be combined to extract the rPPG signal. Usual RGB fusion methods comprise blind source separation [21, 23, 24] and chrominance-based approaches [13]. Emergence and popularization of deep learning methods urged a number of researchers to propose deep learning based models for rPPG signal extraction [6–8, 16, 27, 28, 48]. Several physiological parameters can be extracted from a BVP signal, such as heart rate, respiratory rate, or pulse rate variability (PRV) [17, 22, 41]. However, the PPG and rPPG techniques can be sensitive to noise induced by motion [12, 14, 40, 42].

Recurrent neural networks (RNNs) are a type of artificial neural networks suited for sequence processing. Sequences can be time sequences (such us speech signals or a piece of music), logical sequences, text, etc. RNNs can be used for instance for key-word detection, element prediction in order to complete an input sequence (a musical note to be added to a music score for example), text translation, or sequence classification problems. RNNs are constituted of a set of units, each connected to an element from the input sequence. Every unit is also linked to the previous and the following unit, hence the recurrence qualification for this type of neural networks.

Basic RNNs are confronted, due to their usual substantial size, to the vanishing gradient problem, leading the network to lose information along its units. To palliate this problem, two sorts of units were proposed to replace basic RNN units: *long-short term memory* (LSTM) [15] and

gated recurrent units (GRU) [9], both including a memorization capacity to capture information for longer periods. GRU can be seen as a simplified version of LSTM units, giving the opportunity to build models with less parameters to learn, making them faster to train, for performances often comparable to LSTM models [10].

For cardiac signal processing applications, several researchers integrated RNNs in ECG signal classification problems for instance. In [1], the objective is to determine whether an ECG portion represents the P wave, the QRS complex or the T wave. To this end, authors propose an RNN constituted of two bidirectional LSTM layers (B-LSTM). A. Malali et al. [26] classify ECG signal samples according to their belonging part of the ECG signals (P wave, QRS complex, T wave or neutral) using a unidimensional convolutional neural network (1D-CNN) combined to B-LSTM layers. In [4], three types of networks for QRS complex segmentation are compared: a B-LSTM-based network, a 1D-CNN-based network and a bidimensional CNN-based network taking images representing ECG signals as inputs.

Other studies used RNNs for cardiac pathology detection from cardiac signals. For example, B. Ballinger et al. predict in [2] cardiovascular risk by detecting one or several pathologies among diabetes, high cholesterol, hypertension and sleep apnea. They developed for this purpose a network model based on 1D-CNN and B-LSTM layers, taking as input different sequences, including the heart rate measured by PPG. In [32], a group of common machine and deep learning models are applied for PPG signal classification depending on atrial fibrillation detection. In the same category, the 2017 *PhysioNet/Computing in Cardiology* (CinC) challenge organized by G.D. Clifford et al. [11] focused on atrial fibrillation detection in short ECG signals (with durations lower or equal to $61s$). The winning team [43] conceived a model combing two classification models, including one based on LSTM layers.

Studies applying RNNs to rPPG signals are less common. An application that we cite in this paper is rPPG signal filtering, realized by D. Botina-Monsalve et al. [5]. The authors proposed a three-LSTM-layer model able to learn the shape of an rPPG signal. They compared its performance to signals filtered using a band-pass filter and a wavelet-based filter, and worked with PPG signals as ground-truth.

Since BVP signals are subject to noise related to a person's motion when measured by some PPG devices or by rPPG, we developed a GRU-based neural network to locate noisy portions in these signals. Signal segmentation seems to be relevant as simple filtering, as in [5], may be insufficient. In fact, in [36], authors were not able to use their whole dataset because of noisy contact and remote BVP signals. Usually, it is more accurate to eliminate a noisy BVP signal from a study than to unprecisely use it to extract features. Hence, segmentation would allow to make physiological parameter estimation from BVP signals more reliable. The article is organized as follows: in Section 2, the method followed to conduct this study is explained, from data acquisition in Section 2.1, labeling in Section 2.2 and augmentation in Section 2.3 to the model definition in Section 2.4. Obtained results are presented and discussed in Section 3, and a conclusion is given in Section 4.

2. Method

As explained in the introduction, rPPG signals are sensitive to noise induced by motion. To palliate this weakness, we propose to use an RNN model to estimate reliable portions in noisy rPPG signals. To do so, data were collected and labeled. Due to the relatively low number of obtained signals, we augmented our data before our GRU-based model was developed for this study.

2.1. Data acquisition

A dataset was built for this study, by collecting rPPG and ECG signals from 11 participants (2 women and 9 men), data are available on request. ECG signals served as a ground truth for data labeling. ECG are known to be more resilient to motion, contrary to BVP signals that can be affected by noise induced by motion. Participants were PhD students aged between 24 and 30, and did not suffer from cardiac pathologies. They were sitting in front of a screen with a fixed RGB camera that was used to film them (at a distance of nearly $80cm$).

Camera used for this study was the *c920HDpro* from *Logitech*, with a number of frames per second of 30 and a full HD resolution ($1080p$). rPPG signals were extracted using a real-time rPPG measuring algorithm from an input video stream. This algorithm follows the same computing steps as in [37]. ECG signals were collected using the *Polar H10* sensor[1], equipped with electrodes and maintained in contact with the skin via a belt (furnished with the sensor) worn around participants' chest. The combination formed by the sensor and the belt weighted $60g$, and did not created any discomfort among participants during signal measurement. The *Polar Sensor Logger*[2], available in Android App Store, was downloaded on a mobile phone and used to choose the data to be measured (ECG in our study).

ECG and rPPG measures were acquired in parallel. In the meantime, participants were invited to keep seated, and were allowed to work on their laptop or talk at times, as long as their faces remained directed towards the camera and their movements limited (noisy data within rPPG sig-

[1] https://www.polar.com/us-en/products/accessories/h10_heart_rate_sensor

[2] https://play.google.com/store/apps/details?id=com.j_ware.polarsensorlogger&hl=en&gl=US

nals were necessary for this study). Measured signals lasted in average $29min06s$, more details are given in Table 2.

2.2. Data labeling

Since our goal is to build an RNN model in order to classify rPPG signal samples into *reliable* and *unreliable*, acquired data labeling was essential, and was conducted according to following steps:

- **First rPPG peak labeling:** collected ECG signals served as ground truth for rPPG data labeling. First, ECG and rPPG signals were aligned for each participant, then peaks of both signals were detected. These peaks correspond to heart beats in ECG signals, and to blood pulsations in rPPG signals. A first rPPG peak labeling was realized by comparing peak occurring times with those of ECG peaks. Next, a difference d was computed between each rPPG peak occurring time and its nearest ECG peak occurring time. Thereafter, following condition was applied: for each rPPG peak, if d was lower than a fixed threshold, the rPPG peak was considered as *reliable*, and as *unreliable* otherwise (threshold was set to $0.1s$) ;
- **Second rPPG peak labeling after visual inspection:** the first labeling step is based on occurring times and does not take into account rPPG signal shape, yet this can be a key element in rPPG signal analysis, as in [30] where a *Signal Quality Index* (SQI) is defined based on ECG and PPG signal shapes. In fact, authors of [30] evaluate the correlation between PPG waves that constitute a PPG signal. This led us to visually inspect rPPG peak labels defined following the first labeling step, in order to correct mislabeled peaks when necessary ;
- **rPPG sample labeling:** labeled rPPG peaks were used to label rPPG signal samples. To that end, the algorithm presented in Algorithm 1 was developed, and involved following data:

Table 2 gives the length of measured rPPG signals and the percentage of samples labeled as *reliable* in each signal. Signal lengths range between $23min27s$ and $30min34s$, and average *reliable* sample percentage is 74.53%. rPPG signal with the highest percentage of *reliable* samples belongs to participant 2, whereas signal pertaining to participant 7 does not include any *reliable* sample.

2.3. Data augmentation

As mentioned in the previous section, 11 rPPG signals lasting from $23min$ to $31min$ were collected. These durations are considerable, necessitating large RNNs, which would imply heavy and slow computations. For this reason, we chose to work on 1-minute-long signals. This option seemed reasonable as it would not need excessively long training durations while allowing the developed model to process signals with sufficient information about sample reliability criteria.

In order to obtain $1min$-long signals, measured rPPG signals were divided into segments of $1min$ length. This

notation	*signification*
s	input rPPG signal
e	input rPPG signal samples (e_i is the i^{th} sample, $s(e_i)$ is its rPPG value)
L	input rPPG signal length
p	input rPPG signal peaks (p_j is the j^{th} peak)
N	number of input rPPG signal peaks
$E_{\{e \to e'\}}$	set of input rPPG signal samples between sample e and sample e'
$l(e)$	labeling function, common to rPPG samples and peaks ($l(e) = 1$ if sample e is *reliable* and $l(e) = 0$ otherwise)

Table 1. Notation of data involved in the rPPG sample labeling.

Algorithm 1: rPPG sample labeling algorithm

input : labeled rPPG peaks of length L, rPPG signal s
output: labeled samples of signal s

$l(E_{\{e_1 \to p_1\}}) \leftarrow l(p_1)$; `// assign first peak label to preceding samples`
$l(E_{\{p_N \to e_L\}}) \leftarrow l(p_N)$; `// assign last peak label to following samples`
foreach p_i *with* $i \in [2, N]$ **do**
  if $l(p_i) \neq l(p_{i-1})$ **then**
    if $l(p_i) = 0$ **then**
      Find first sample e with $s(e) < 0$ in $E_{\{p_{i-1} \to p_i\}}$;
      $l(E_{\{e \to p_i\}}) \leftarrow 0$;
      $l(E_{\{p_{i-1}) \to e\}} \leftarrow 1$;
    else
      Find last sample e with $s(e) < 0$ in $E_{\{p_{i-1} \to p_i\}}$;
      $l(E_{\{p_{i-1} \to e\}}) \leftarrow 0$;
      $l(E_{\{e \to p_i\}}) \leftarrow 1$;
    end
  else
    $l(E_{\{p_{i-1} \to p_i\}}) \leftarrow l(p_i)$;
  end
end

Participant	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Measured signal durations	$29min$ $25s$	$29min$ $59s$	$23min$ $27s$	$30min$ $00s$	$29min$ $22s$	$29min$ $34s$	$30min$ $14s$	$29min$ $09s$	$28min$ $11s$	$30min$ $34s$	$30min$ $11s$
Percentage of *reliable* samples (%)	80.93	95.68	55.71	93.65	83.59	94.55	0.00	90.61	80.77	85.30	59.06

Table 2. Duration and *reliable* sample percentage of measured rPPG signals for our study.

led us to 316 signals, partitioned over the 11 participants. This constitutes a relatively low number comparing to data sizes that are usually used in deep learning models. To address this problem, we decided to augment our data by generating new signals based on measured rPPG signals. Data augmentation allows to reduce overfitting, preventing the model from learning to reproduce training data while giving weak prediction results with testing data. In fact, by increasing data size, a model is exposed to more observations, and generalizes more easily its learning to unknown data.

To augment our data, rPPG signals were separated into $20s$ and $30s$-long segments. $20s$ segments were next combined by sets of 3 segments, and $30s$ segments by sets of 2 segments, in order to form signals with $1min$ lengths. Generated signals were individual and originated from ($20s$ or $30s$) segments pertaining to each participant separately. Respecting this condition seemed primordial for purposes of preserving the coherence of rPPG measurements associated to each person in terms of shape characteristics (frequency, amplitude, etc). Besides, from a physiological point of view, combining signals that reflect the cardiac activity of distinct people intuitively appears to be inconsistent.

Since measured rPPG signals were mostly constituted of *reliable* samples (as shown in Table 2), we sought at increasing the number of *unreliable* samples. Thus, all possible combinations of $30s$ segments containing *unreliable* samples were considered, generating a total number of 3311 $1min$-long rPPG signals.

For all participants except participant 7, whom rPPG peaks were totally labeled as *unreliable*, $20s$ segments should contain at least 10% of *unreliable* samples. Defining a threshold allowed to avoid generating too many signals with few *unreliable* samples, as our goal was to increase the proportion of *unreliable* samples within our data. Besides, combining all $20s$ segments would have led to an excessively great number of generated signals, contrary to $30s$ segments. The value of 10% was fixed based on the histogram of *unreliable* samples present in $20s$ segments derived from measured rPPG signals except the signal from participant 7 (70% of $20s$ segments contained less than 10% *unreliable* samples).

5334 $1min$-long signals were obtained by combining $20s$ segments from all the participants except participant 7. 2039 signals were kept among possible combinations of $20s$ segments from participant 7's rPPG signal, in order to reach a total number of 11000 signals, distributed for our RNN model training and test as follows: 10000 signals constituted the training set, while the 1000 remaining signals were used to test our model.

30 segments to be merged to form $1min$ signals were selected in the following manner: for each participant, among all possible combinations of segments containing *unreliable* samples, combinations with successive segments in the original rPPG signals were eliminated in order to avoid redundancy in our data. For each pair of segments s_1 and s_2 retained for a fusion (s_1 and s_2 respectively represent the first and the last $30s$ of the generated signal), their derivatives s_1' and s_2' are computed. Samples e_1 and e_2 corresponding to the last sign change of s_1' and to the first sign change of s_2' are located. Samples following e_1 were removed from s_1 and those preceding e_2 were removed from s_2, then the signs of $s_1'(e_1)$ and $s_2'(e_2)$ were compared and s_1 and s_2 were combined only if $s_1'(e_1)$ and $s_2'(e_2)$ were different for the purposes of avoiding signal discontinuity. Triplets of $20s$ segments s_1, s_2 and s_3 to be merged in order to constitute $1min$-long signals were selected according to the same process.

$20s$ and $30s$ segments were combined following Algorithm 2, which merges two input segments into one output signal s. Algorithm 2 was applied twice for $20s$ segments since they necessitated two fusions to form a $1min$ signal. In Algorithm 2, *min* refers to the minimum function: $min(a, b) = a$ if $a < b$ and $min(a, b) = b$ otherwise.

Figure 1 shows an example of Algorithm 2 execution steps on segments s_1 and s_2 (obtained from participant 1), where $|s_1(L_1)|$ and $|s_2(1)|$ first are compared (left figure). In this example, $x = min(|s_1(L_1)|, |s_2(1)|) = |s_1(L_1)|$. Next, time t (a unique integer ranged in $[1, L_2]$) is determined in such a manner that: $|s_2(t+1)| \leq x \leq |s_2(t)|$ (middle figure). Lastly, the portion of s_2 comprised between times $t+1$ and L_2 is kept and combining s_1 and s_2 gives the signal s shown in the right figure of Figure 1.

Labels defined for rPPG samples (as explained in Section 2.2) were used for the generated signals. Phases of $20s$ and $30s$ segment selection and fusion led to truncate the

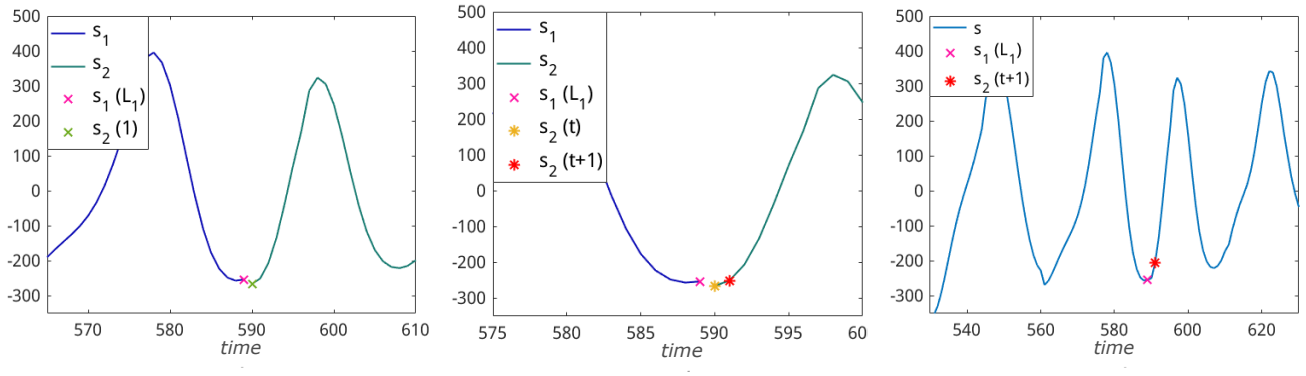

Figure 1. Application of Algorithm 2 steps: left figure shows $s_1(L_1)$ (L_1 is the length of s_1) and $s_2(1)$ values. Minimum value x of their absolute values is computed and time t so that $|s_2(t+1)| \leq x \leq |s_2(t)|$ is determined (middle figure). Lastly, s_1 is concatenated to the portion of s_2 between times $t+1$ and L_2 (L_2 is the length of s_2) to generate signal s.

Algorithm 2: segment fusion algorithm

input : segments s_1 and s_2 of respective lengths L_1 and L_2
output: generated signal s

$x \leftarrow min(|s_1(L_1)|, |s_2(1)|)$;
if $x = |s_1(L_1)|$ **then**
  Find time t so that: $|s_2(t+1)| \leq x \leq |s_2(t)|$;
  $s_2 \leftarrow s_2(t+1 : L_2)$;
else
  Find time t so that: $|s_1(t)| \leq x \leq |s_1(t+1)|$;
  $s_1 \leftarrow s_1(1 : t)$;
end
concatenate s_1 and s_2 to form s;

combined segments, resulting in having signals lasting les than $1min$. The minimum length obtained was $L = 1742$ corresponding to a duration d of $d = \frac{L}{f_s} = \frac{1742}{30} = 58.07$ (f_s being the sampling frequency of rPPG signals). All generated signals were truncated to time $t = 1742$ in order to homogenize our data.

Examples of generated and labeled rPPG signals by original signal divison in $1min$ segments as well as by $20s$ and $30s$ segment combination are given in Figure 2. Sample labeling forms reliable and unreliable portions (i.e. groups of samples labeled as *reliable* and *unreliable* respectively).

2.4. Model definition

After data labeling and augmentation, we defined our RNN model, composed of a unidimensional CNN, followed by a bidirectional GRU, which we denote as 1D-CNN+B-GRU and present in this subsection.

Input sequences that were given to our model were rPPG temporal traces, their time-frequency representation

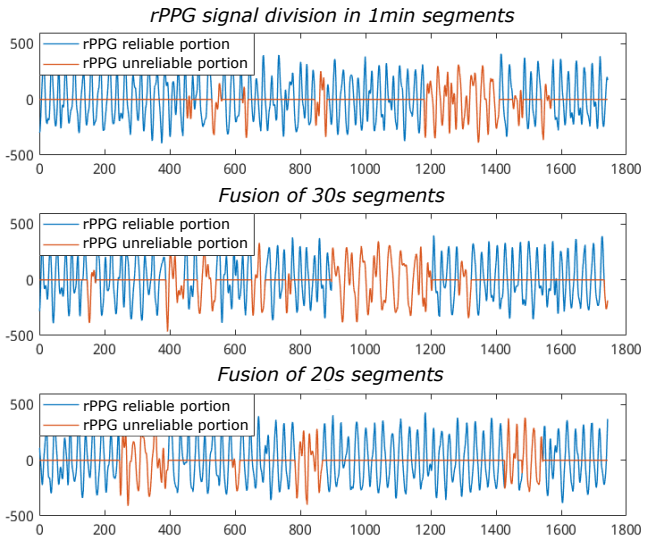

Figure 2. Examples of generated and labeled rPPG signals. Signals were generated by: original signal division into $1min$ segments (top figure), $30s$ segment fusion (middle figure) and $20s$ segment fusion (bottom figure).

as well as the concatenation of these two modalities. Time-frequency representations of rPPG signals were obtained using the *Fourier Synchrosqueezed Transform* (FSST) [29].

FSST computes an amplitude spectrum over windows of the rPPG signal that are centered around each sample, thus allowing to access spectral information while keeping the same signal time resolution. Including frequency components in the analysis of sequential data enables taking into account frequency properties, and is often used in deep learning models for sequence processing. This is the

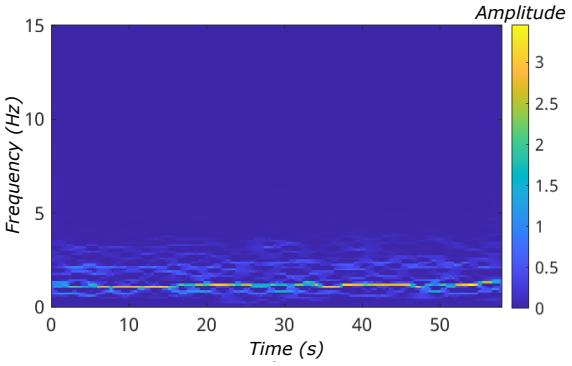

Figure 3. Amplitude spectrum of an rPPG signal by application of the FSST.

case for example in [38], where a CNN is developed for audio signal classifications operating their time-frequency representation, and in [47] where authors combine CNN and RNN networks for a magic word detection in speech signals by analyzing their time-frequency content. In our study, FSST was applied with Kaiser windows (*width* $= 64$ and $\beta = 10$). Figure 3 gives an example of an rPPG signal time-frequency representation from our data.

Figure 4 shows the global architecture of the 1D-CNN+B-GRU model. Input sequences of shape (L, d) (L and d being the length and the width of the sequences) are given to the model through batches. As previously mentioned, $L = 1742$ while d depends on the input modality: $d = 1$ for time sequences, $d = 33$ for time-frequency sequences and $d = 34$ for the concatenation of both modalities. The first layer is a unidimensional CNN (1D-CNN) comprised of 4 filters, each of a kernel size equal to 5 and gives four output batches each of shape (L, d). A *flatten* layer groups these outputs into a batch of shape $(L, 4 \times d)$, which is transmitted to a bidirectional GRU (B-GRU). The B-GRU combines the outputs of two GRU networks (with 128 units each) that process sequences in opposite directions, allowing for each sample to take into account both preceding and following samples. A batch normalization step is then applied, followed by a dropout of 20%. Next, a *dense* layer combines the output of previous layers into a shape of $(L, 1)$. Finally, a probability (between 0 and 1) of a sample being *reliable* or *unreliable* for times t between 1 and L is given with a sigmoid function σ. Retained labels are *reliable* if $\sigma(t) > 0.5$ and *unreliable* otherwise.

3. Results and discussion

Our 1D-CNN+B-GRU model was trained and tested on our data, through three modalities: temporal sequences, time-frequency sequences and combination of time and time-frequency sequences by concatenation. As precised in Section 2.3, the training set was composed of 10000 sequences and the test set of 1000 sequences per modality.

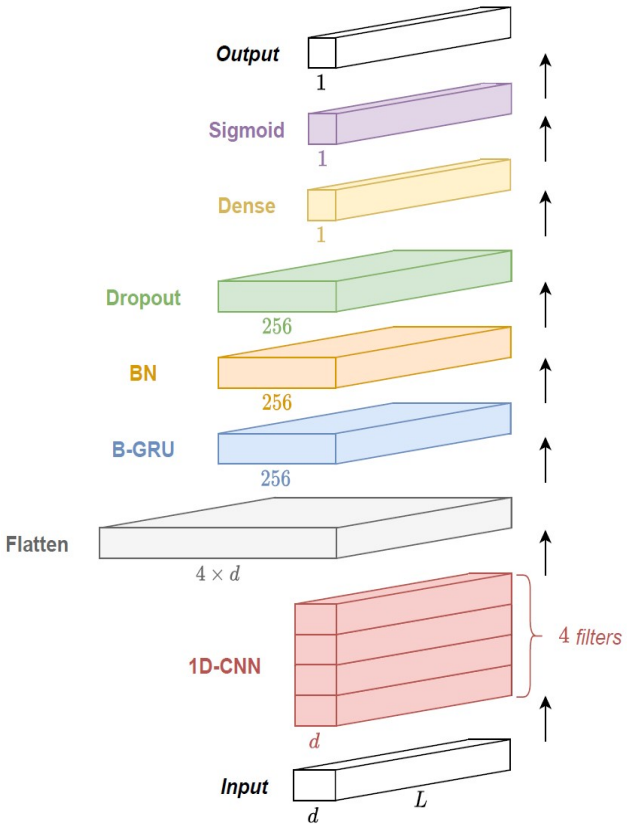

Figure 4. Architecture of the 1D-CNN + B-GRU model.

The optimization algorithm chosen for model training was Adam, with the recommended values of α, β_1 and β_2 in [19] ($\alpha = 0.001$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$). Selected loss function was binary cross entropy, which is adapted for binary classification problems as it is the case for our study.

We fixed the maximum number of training epochs at 600. From the 500th epoch, an *early stopping* condition was integrated so that the model ended learning if a defined metric stopped evolving. The metric chosen in our study was the test set accuracy, and training was paused if it did not increase for consecutive 40 epochs. Input batches contained 128 sequences for each modality.

Figure 5 gives the curves of the training accuracy and loss evolution over the first 400 epochs for sequences combining time and time-frequency rPPG data. Increasing accuracy indicates that the model improved its capacity to correctly classify rPPG samples over time, while decreasing loss shows that predicted label distribution bonded that of true labels.

Table 3 presents rPPG sample classification results using the 1D-CNN+B-GRU model on temporal sequences (de-

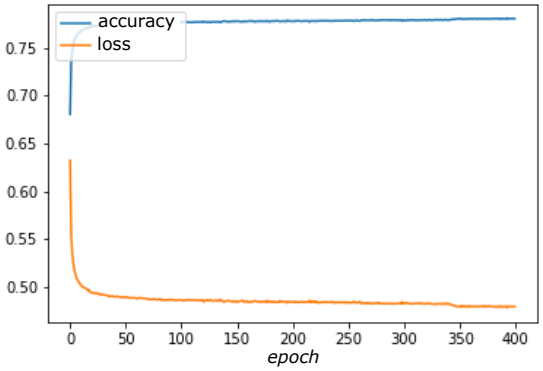

Figure 5. Training accuracy and loss of the 1D-CNN+B-GRU model on rPPG time-frequency sequences.

Input data	*Accuracy (%)*	*epoch*
time	85.88	508
time-frequency	83.33	578
time + time-frequency	84.86	583

Table 3. Results of rPPG sample classification into *reliable* and *unreliable* samples using the 1D-CNN+B-GRU model for the three modalities considered in our study: temporal sequences (*time*), time-frequency representations (*time-frequency*) and combination of both modalities by concatenation (*time + time-frequency*).

noted as *time*), time-frequency representation of rPPG signals (denoted as *time-frequency*), and the combination of these two modalities (*time + time-frequency*).

The best classification accuracy rate of 85.88% was obtained using rPPG time sequences from the test set. Time-frequency representation seems to be at its own insufficient for our classification problem, and combining it to the temporal sequences allowed to reach similar results as with the temporal sequences alone, without exceeding them for our maximum training duration (600 epochs in our study).

Obtained accuracy rate can be justified by the use of the B-GRU layer, as it takes into consideration the neighborhood of each sample before labeling it, knowing that an rPPG sample is luckily to have the same label as its near neighbors. Furthermore, the 1D-CNN layer allows the model to learn local patterns over the input sequences, to which the B-GRU layer adds temporal dependencies.

Figure 6 shows the confusion matrix of rPPG segmentation using the 1D-CNN+B-GRU model, indicating the percentage of correctly labeled and mislabeled samples according to the true labels. The confusion matrix gives the sensitivity (percentage of true positives), and the specificity (percentage of true negatives), which are in our case of 93% and

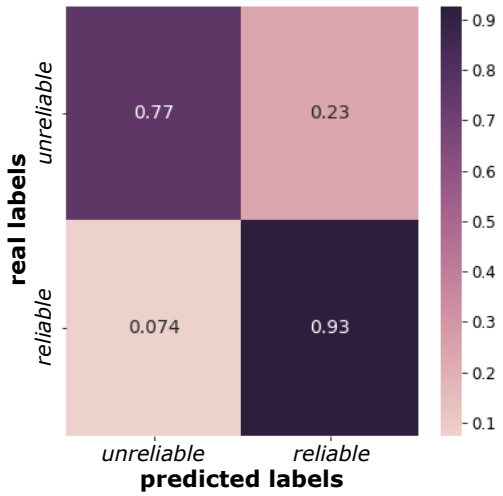

Figure 6. Confusion matrix of rPPG time sequence segmentation using the 1D-CNN+B-GRU model.

77% respectively.

Figure 7 illustrates two examples of rPPG signals from the test set, both with their true and predicted labels. Visually, segmentation results are promising and it can be seen that the model learned to locate *reliable* and *unreliable* samples. Sensitivity and specificity values given by Figure 6 suggest that our model recognize *reliable* sample more frequently that it does with *unreliable* samples. This can be observed in Figure 7, as for both examples, the model seems to locate quite well the beginning and ending of non reliable portions, yet it attributes a *reliable* label to some samples within these portions.

4. Conclusion and perspectives

Being able to define noisy pulse signal portions because of a person's motion led us to explore the feasibility of segmentation method based on a recurrent neural network model. To this end, rPPG and ECG signals were collected from 11 participants. ECG signals were used to label rPPG samples through two phases (peak labeling and sample labeling). A data augmentation step was realized by dividing original rPPG signals into $1min$-long segments, as well as by defining a $20s$ segment and $30s$ segment fusion algorithm, allowing to reach a total number of 11000 signals (10000 and 1000 signals for model training and testing respectively).

Model developed in our study is constituted of two major layers: a unidimensional CNN (1D-CNN), and a bidirectional GRU (B-GRU). Our model was trained on rPPG time sequences, time-frequency representation of the rPPG signals (obtained by applying the FSST), and the concate-

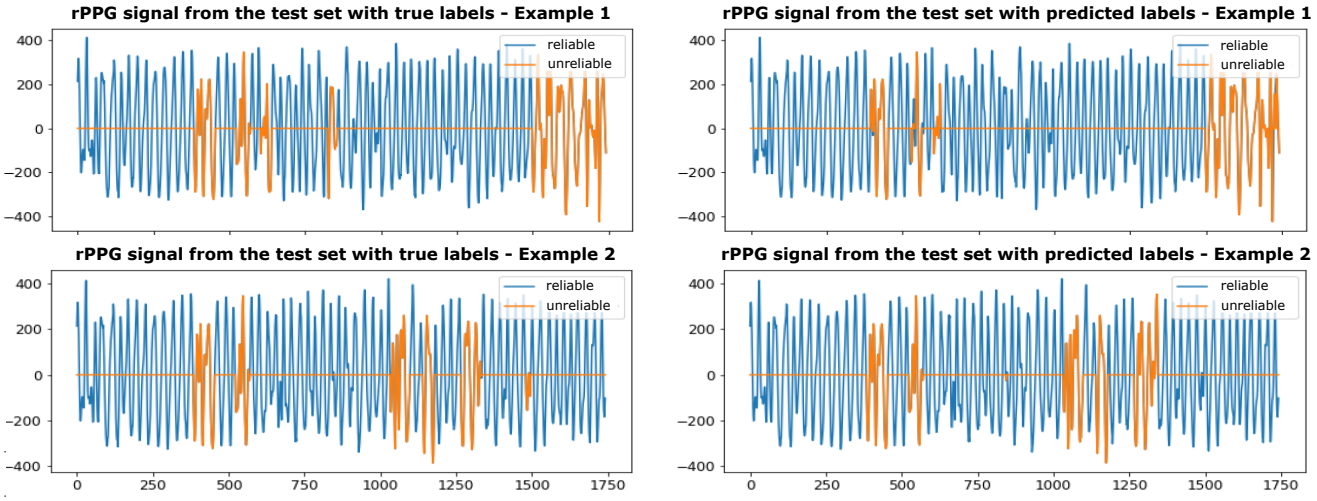

Figure 7. Two examples of rPPG signal segmentation using the 1D-CNN+B-GRU (input sequences are temporal signals from the test set).

nation of these two modalities. The best accuracy rate of 85.88% was reached by the time sequences, and predicted labels are visually close to true labels.

Locating noisy portions of rPPG signals allows not only to directly extract reliable portions instead of eliminating noisy signals, but also to correctly extract related data (such as features of PRV signals). However, our model can be improved by making its architecture more complex (an attention mechanism can be included for example), or precisely fine tuning the hyperparameters, so as to avoid mislabeling samples within noisy portions.

Further works have to be led in order to either group all the reliable portions found in an rPPG signals, taking into account discontinuity-related problems, or define reliable features that can hold rPPG information over ultra short durations.

References

[1] H. Abrishami and M. Campbell et al. Supervised ECG interval segmentation using lstm neural network. *Int'l Conf. Vioinformatics and Computational Biology*, 2018. 2

[2] B. Ballinger and J. Hsieh et al. Deepheart: Semi-supervised sequence learning for cardiovascular risk prediction. *The Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 2

[3] S. Bobbia and Y. Benezeth et al. Remote photoplethysmography based on implicit living skin tissue segmentation. *IEEE International Conference on Pattern Recognition (ICPR)*, 2016. 1

[4] A. Borde and V. Skuratov. Development of neural network-based approach for qrs segmentation. *25th Conference of Open Innovations Association (FRUCT)*, 2019. 2

[5] D. Botina-Monsalve and Y. Benezeth et al. Long short-term memory deep-filter in remote photoplethysmography. *Conference on Computer Vision and Pattern Recognition*, 2020. 2

[6] F. Bousefsaf and A. Pruski et al. 3d convolutional neural networks for remote pulse rate measurement and mapping from facial video. *Applied Sciences*, 2019. 1

[7] W. Chen and D. McDuff. DeepPhys: Video-based physiological measurement using convolutional attention networks. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 1

[8] C.H. Cheng and K.L. Wong et al. Deep learning methods for remote heart rate measurement: A review and future research agenda. *Sensors*, 2021. 1

[9] K. Cho and B. van Merriënboer et al. On the properties of neural machine translation: Encoder-decoder approaches. *NIPS 2014 Deep Learning and Representation Learning Workshop*, 2014. 2

[10] J. Chung and C. Gulcehre et al. Empirical evaluation of gated recurrent neural networks on sequence modeling. *NIPS 2014 Deep Learning and Representation Learning Workshop*, 2014. 2

[11] G.D. Clifford and C. Liu et al. Af classification from a short single lead ECG recording: the physionet/computing in cardiology challenge 2017. *Computing in Cardiology*, 2017. 2

[12] L. F. and L.-M. Po et al. Motion-resistant remote imaging photoplethysmography based on the optical properties of skin. *IEEE Transactions on Circuits and Systems for Video Technology*, 2015. 1

[13] G. De Haan and V. Jeanne. Robust pulse-rate from chrominance-based rPPG. *IEEE Trans. on Biomedical Engineering*, 2013. 1

[14] M.J. Hayes and P.R. Smith. Artifact reduction in photoplethysmography. *Applied Optics*, 1998. 1

[15] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 1997. 1

[16] G.S. Hsu and A. Ambikapathi et al. Deep learning with time-frequency representation for pulse estimation from fa-

cial videos. *IEEE International Joint Conference on Biometrics*, 2017. 1

[17] K. Humphreys and T. Ward et al. Non contact simultaneous dual wavelength photoplethysmography : A further step toward non contact pulse oximetry. *Review of Scientific Instruments*, 2007. 1

[18] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 1

[19] D.P. Kingma and J.L. Ba. Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations (ICLR)*, 2015. 6

[20] K.J. Lee and C. Park et al. Tracking driver's heart rate by continuous-wave Doppler radar. *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2016. 1

[21] M. Lewandowska and J. Ruminski et al. Measuring pulse rate with a webcam - a non-contact method for evaluating cardiac activity. *Proceedings of the Federal Conference on Computer Science and Infor-mation Systems*, 2011. 1

[22] D. Luguern and S. Perche et al. An assessment of algorithms to estimate respiratory rate from the remote photoplethysmogram. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1

[23] R. Macwan and Y. Benezeth et al. Parameter-free adaptive step-size multi-objective optimization applied to remote photoplethysmography. *IEEE EMBS International Conference on Biomedical and HealthInformatics*, 2018. 1

[24] R. Macwan and Y. Benezeth et al. Remote photoplethysmography with constrained ica using periodicity and chrominance constraints. *BioMedical Engineering OnLine, Springer*, 2018. 1

[25] R. Macwan and Y. Benezeth et al. Heart rate estimation using remote photoplethysmography with multi-objective optimization. *Biomedical Signal Processing and Control*, 2019. 1

[26] A. Malali and S. Hiriyannaiah et al. Supervised ECG wave segmentation using convolutional lstm. *ScienceDirect*, 2020. 2

[27] X. Niu and S. Shan et al. Rhythmnet: End-to-end heart rate estimation from face via spatial-temporal representation. *IEEE Transactions on Image Processing*, 2019. 1

[28] X. Niu and Z. Yu et al. Video-based remote physiological measurement via cross-verified feature disentangling. *European Conference on Computer Vision*, 2020. 1

[29] T. Oberlin and S. Meignen et al. The fourier-based synchrosqueezing transform. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014. 5

[30] C. Orphanidou and T. Bonnici et al. Signal-quality indices for the electrocardiogram and photoplethysmogram: Derivation and applications to wireless monitoring. *IEEE Journal of Biomedical and Health Informatics*, 2015. 3

[31] A. Pedrana and D. Comotti et al. Development of a wearable in-ear ppg system for continuous monitoring. *IEEE Sensors Journal*, 2020. 1

[32] T. Pereira and C. Ding et al. Deep learning approaches for plethysmography signal quality assessment in the presence of atrial fibrillation. *Physiological Measurement*, 2019. 2

[33] M.Z. Poh and D.J. McDuff et al. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. *Optics Express*, 2010. 1

[34] M.Z. Poh and D.J. McDuff et al. Advancements in noncontact, multiparameter physiological measurements using a webcam. *IEEE Transactions on Biomedical Engineering*, 2011. 1

[35] M.-Z. Poh and K. Kim et al. Cardiovascular monitoring using earphones and a mobile device. *IEEE Pervasive Computing*, 2012. 1

[36] R. Meziati Sabour and Y. Benezeth et al. UBFC-Phys: A multimodal database for psychophysiological studies of social stress. *IEEE Transactions on Affective Computing*. 2

[37] R. Meziati Sabour and Y. Benezeth et al. UBFC-Phys: A Multimodal Database for Psychophysiological Studies OF Social Stress. *IEEE Transactions on Affective Computing*. 2

[38] J. Salamon and J.P. Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 2016. 6

[39] K. Shin and Y. Kim et al. A novel headset with a transmissive ppg sensor for heart rate measurement. *13th International Conference on Biomedical Engineering*, 2009. 1

[40] R. Spetlık and V. Franc et al. Visual heart rate estimation with convolutional neural network. *Proceedings of the British Machine Vision Conference*, 2018. 1

[41] Y. Sun and S. Hu et al. Non contact imaging photoplethysmography to effectively access pulse rate variability. *Journal of Biomedical Optics*, 2012. 1

[42] Y. Sun and N. Thakor. Photoplethysmography revisited: from contact to non contact, from point to imaging. *IEEE Trans. on Biomedical Engineering*, 2016. 1

[43] T. Teijeiro and C.A. García et al. Arrythmia classification from the abductive interpretation of short single-lead ECG records. *Computing in Cardiology*, 2017. 2

[44] W. Wang and B.D. Brinker et al. Algorithmic principles of remote-PPG. *IEEE Transactions on Biomedical Engineering*, 2017. 1

[45] W. Wang and S. Stuijk et al. A novel algorithm for remote photoplethysmography : Spatial subspace rotation. *IEEE Transactions on Biomedical Engineering*, 2015. 1

[46] B.-F. Wu and C.-H. Lin et al. A contactless sport training monitor based on facial expression and remote-ppg. *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017. 1

[47] T. Yamamoto and R. Nishimura et al. Small-footprint magic word detection method using convolutional lstm neural network. *Proc. Interspeech 2019*, 2019. 6

[48] Z. yu and W. Peng et al. Remote heart rate measurement from highly compressed facial videos: An end-to-end deep learning solution with video enhancement. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 1