PPG-based Heart Rate Estimation with Time-Frequency Spectra: A Deep Learning Approach

Attila Reiss
Robert Bosch GmbH,
Corporate Research, Germany
attila.reiss@de.bosch.com

Ina Indlekofer
University Stuttgart,
Stuttgart, Germany
st151932@stud.uni-stuttgart.de

Philip Schmidt
Robert Bosch GmbH,
Corporate Research, Germany
University Siegen, Germany
philip.schmidt@de.bosch.com

Kristof Van Laerhoven
University Siegen,
Siegen, Germany
kvl@eti.uni-siegen.de

UbiComp/ISWC '18 Adjunct, October 8–12, 2018, Singapore, Singapore

ACM ISBN 978-1-4503-5966-5/18/10...$15.00.
https://doi.org/10.1145/3267305.3274176

Abstract
PPG-based continuous heart rate estimation is challenging due to the effects of physical activity. Recently, methods based on time-frequency spectra emerged to compensate motion artefacts. However, existing approaches are highly parametrised and optimised for specific scenarios. In this paper, we first argue that cross-validation schemes should be adapted to this topic, and show that the generalisation capabilities of current approaches are limited. We then introduce deep learning, specifically CNN-models, to this domain. We investigate different CNN-architectures (e.g. the number of convolutional layers, applying batch normalisation, or ensemble prediction), and report insights based on our systematic evaluation on two publicly available datasets. Finally, we show that our CNN-based approach performs comparably to classical methods.

Author Keywords
Heart rate, PPG, Time-frequency spectrum, Deep learning, CNN, Evaluation methods

Introduction
Continuous heart rate monitoring is essential in a number of domains, e.g. for healthcare or fitness applications. Recently, wrist-worn heart rate monitors (based on photoplethysmography (PPG)), have become widely used. Various commercial products include a PPG-sensor nowadays,

such as the Apple Watch [2] or the Samsung Simband [18]. However, compared to traditional ECG, heart rate estimation is more challenging based on the PPG-signal. Especially motion artefacts, caused by the user's physical activity, can significantly reduce the PPG-signal quality. There exists a large amount of work relying on different (adaptive) filtering methods (e.g. [15]) or separating the heart rate component from the motion artefact and other noise components within the time series (e.g. [11]).

One major issue within this research field is the limited availability of labelled data, hindering purely data-driven approaches. Recently, physical knowledge has been used to alleviate this issue. Since heart rate produces a periodical signal, representations outside the time-domain could prove beneficial to highlight this periodicity. For example, Jaafar and Rozali [10] applied wavelet transform analysis to estimate heart rate and breathing rate from the PPG-signal. Another promising signal representation is the time-frequency spectrum, which emphasises the periodic heart rate as well as periodic physical activities. The core idea of spectrum-based approaches is to distinguish the heart rate and motion-based periodicity, supported by motion-derived time-frequency spectra (e.g. based on simultaneously recorded PPG and acceleration signals) [17, 19]. However, extracting heart rate from time-frequency spectra is cumbersome with current approaches, as they are highly parametrised and were developed to fit only certain scenarios. Our results, presented below, indicate that there is still a need for an approach which can generalise well.

We suggest an end-to-end learning technique, instead of relying on hand-crafted rules and features. Our deep learning approach takes the time-frequency spectra of the PPG- and accelerometer-signals as input, and provides the estimated heart rate as output. Deep learning has already been applied to various time-series signals, such as for human activity recognition [5, 7, 13] or gait parameter extraction [8]. Gjoreski et al. [5] compared deep and classical machine learning methods on a human activity classification task. Hammerla et al. [7] compared CNNs to recurrent neural networks (RNN), concluding that CNNs are more suitable for continuous periodic activities. Hannink et al. [8] used CNNs for regression tasks, estimating different parameters related to human gait. Considering deep learning approaches for PPG-signal analysis, Shashikumar et al. [20] provide an example: They first applied wavelet transform on an 8-channel PPG-signal, and then used a CNN-model to extract relevant information for atrial fibrillation detection.

In this paper, we investigate different CNN-architectures for heart rate estimation based on PPG-data. In doing so, we rely on physical knowledge of PPG and accelerometer sensor data, as shown beneficial in recent work. Based on the analytic understanding of heart rate and many physical activities (mainly periodic signals), and based on related work with spectrum-based classical models, we use time-frequency spectra as input representation for our CNN-models. To the best of our knowledge, this is the first paper applying deep learning to PPG-spectrum-based heart rate estimation. The main contributions of this work are threefold:

1. We introduce the leave-one-session-out cross-validation to this domain, and show that the generalisation capabilities of current approaches are limited.
2. We introduce deep learning for PPG-spectrum-based heart rate estimation, and compare it to three classical approaches.
3. We investigate different CNN-architectures and report insights based on our evaluation.

Classical Methods

Datasets

We rely on the two publicly available datasets introduced for the IEEE Signal Processing Cup in 2015 [4, 22]: the training dataset (referred to as *IEEE_Training*) and the test dataset of the competition (referred to as *IEEE_Test*). The datasets were recorded with a wrist-worn device, including a two-channel PPG sensor (both green LEDs, wavelength: $515nm$) and a three-axis accelerometer (placed at the same position as the PPG-sensor). Moreover, both datasets include a simultaneously recorded ECG signal, providing heart rate ground truth. The sampling rate of all signals is $125Hz$. The *IEEE_Training* dataset consists of 12 sessions, each recorded from a different subject while running on a treadmill at varying speed. Each session lasted approximately five minutes. The *IEEE_Test* dataset consists of 10 sessions, recorded from 8 different subjects while performing one of two exercise types. Subjects performed various forearm and upper arm exercises during 4 sessions, and performed mainly intensive arm movements (such as boxing) during 6 sessions. Each of these sessions lasted approximately five minutes. More details on both datasets can be found in [4, 22].

Ground truth for both datasets is given every two seconds, calculated as mean heart rate over 8 seconds from the ECG-signal. We adapt this sliding window approach (window length: 8 seconds, window shift: 2 seconds) for heart rate estimation, as was done in previous work [17, 19, 21, 22]. Ground truth and heart rate estimation are given in beats-per-minute (bpm) in this paper. As performance metric, we rely on the mean absolute error (MAE):

$$MAE = \sum_{w=1}^{W} |BPM_{est}(w) - BPM_{ref}(w)| \qquad (1)$$

where W is the total number of windows, and $BPM_{est}(w)$ and $BPM_{ref}(w)$ denote the estimated and reference heart rate value in beats-per-minute on the wth window, respectively.

Algorithms

Based on the above described datasets, various techniques have been proposed to estimate heart rate from PPG-signals corrupted by motion artefacts. We focus on two recent algorithms in this paper, reported to outperform previous approaches (referred to as *SpaMa* [17]) or at least perform similarly with lower computational cost (referred to as *Schaeck2017* [19]). Moreover, we extend the heart rate tracking part of *SpaMa*, which will be referred to as *SpaMaPlus*. A brief description of the three approaches follows below.

SpaMa: This approach first calculates the power spectral density of the PPG and accelerometer signals on each 8-second window. Then, the highest peaks are found in each spectrum. Peaks in the acceleration-spectrum correspond to motion. Therefore, removing these peaks from the PPG-spectrum results in removing the major motion artefacts. The highest peak in the remaining PPG-spectrum corresponds then to the heart rate. Algorithmic details of this approach can be found in [17].

SpaMaPlus: Since abrupt changes in heart rate are physiologically limited, relying on the estimated values from preceding segments is recommendable. However, *SpaMa* only considers the last segment in its heart rate tracking step. In order to increase the robustness of the estimation, we extended this step by considering the last six segments. In case the current segment's heart rate estimation is uncertain, we rely on the mean value of the preceding segments. Moreover, to identify and avoid consecutive false estimations, we examine the plausibility of previous estimations.

Table 1: Evaluation of classical methods on the datasets *IEEE_Training* and *IEEE_Test*. Left column: session optimised results, right column: LOSO cross-validation. Results are reported as MAE $\pm$ STD [bpm], where standard deviation is computed on the different individual sessions.

	IEEE_Training		IEEE_Test	
	optimised	LOSO	optimised	LOSO
SpaMa	1.33$\pm$1.4	13.1$\pm$20.7	2.53$\pm$2	9.20$\pm$11.4
SpaMaPlus	1.38$\pm$1.4	4.25$\pm$5.9	3.56$\pm$3.9	12.31$\pm$15.5
Schaeck2017	1.33$\pm$1.3	2.91$\pm$4.6	6.48$\pm$8.3	24.65$\pm$24

Schaeck2017: Opposed to *SpaMa* and *SpaMaPlus*, this approach relies on both PPG-signal channels. First, to enhance periodic components, a correlation function is applied on the time series signal. Then, the spectrum of the resulting time series is computed. Similar to *SpaMa*, motion artefacts are reduced by taking the acceleration spectra into account. Finally, a linear least squares fit on the preceding three segments is applied for heart rate tracking. Algorithmic details of this approach can be found in [19].

Evaluation

All three algorithms have in common that they include several adjustable parameters, such as the number of highest peaks to consider in each spectrum, or the minimum required frequency difference for removing motion-induced peaks. Instead of applying a cross-validation scheme, it is common practice in related work to tune these parameters for each dataset-session specifically, in order to report optimal results (low MAE-values). However, the practical relevance of these results is limited in our opinion. When deploying PPG-based heart rate estimation algorithms in real-life settings, there is no access to ground truth information. Therefore, the optimisation of these algorithms to a specific subject or even a specific session is not possible. Instead of reporting session-optimised results, we argue that leave-one-session-out (LOSO) cross-validation should be preferred. During this cross-validation scheme, parameter optimisation is performed on all data except of one session, and the left-out session is used as test data. This procedure is repeated so that each session is used as test data exactly once. Thus, results reported with LOSO cross-validation reflect the generalisation capabilities of algorithms. However, as reported in other domains [16], large performance difference is to be expected between subject/session-dependent and -independent evaluation.

We implemented the three algorithms described above (*SpaMa*, *SpaMaPlus*, and *Schaeck2017*) and evaluated them on the IEEE-datasets. For parameter setting, we performed both session optimisation and LOSO cross-validation, using random search of the parameter space in both cases. We report results in Table 1. Considering session optimisation, the results match the ones reported in the original publications [17, 19]. Small differences can be explained by the random parameter search. We observed that especially the *SpaMa*-based approaches are very sensitive to the parameter setting. Considering LOSO cross-validation, the performance significantly decreased compared to the session-optimised results. Moreover, the three implemented algorithms performed differently on the two datasets. *Schaeck2017* performed best on *IEEE_Training*, but was significantly worse than the other two approaches on *IEEE_Test*. On the other hand, the best performing approach on *IEEE_Test* (*SpaMa*) was the worst on the

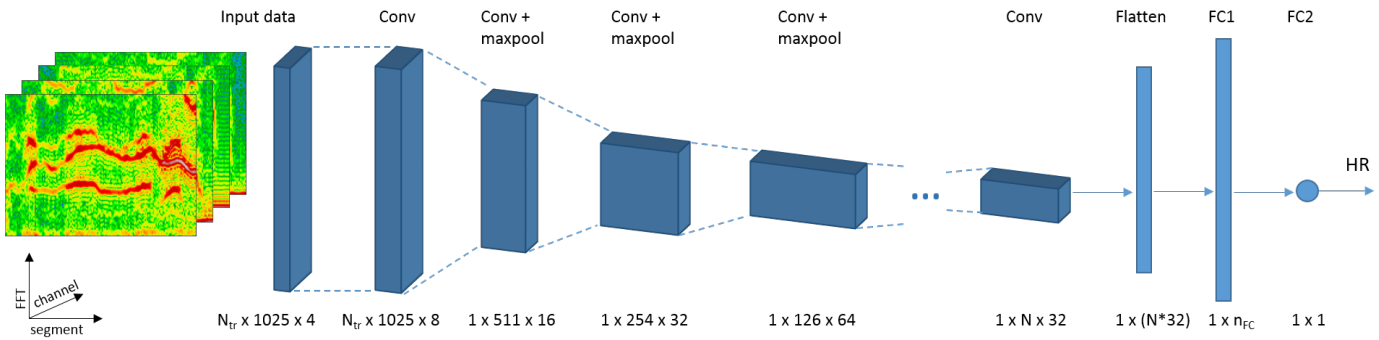

Figure 1: Proposed CNN-architecture with $N_L = 1...8$ convolution-maxpool layers. N depends on N_L. Output: HR [bpm].

dataset *IEEE_Training*. It should be noted that our modified version of the *SpaMa*-approach (*SpaMaPlus*) achieved the best combined result on both datasets while applying LOSO cross-validation. Overall, the results achieved with LOSO cross-validation indicate that there is still a need to develop novel algorithms, which have better generalisation capabilities both considering different subjects as well as different situations (physical activities).

Deep Learning Approach

We perform a series of steps before data is fed to the deep learning model. First, we segment the time-series signal (the first PPG-channel and all three accelerometer-channels) with the sliding window as defined before (window length: 8 seconds, window shift: 2 seconds). We then apply FFT on each time-series segment. The results of this step are $N_{ch} = 4$ time-frequency spectra, one per signal channel. In the next step we cut these spectra, keeping only the $0 - 4Hz$ interval ($4Hz$ corresponds to $240bpm$). The resulting number of FFT-points per segment and channel is $N_{FFT} = 1025$. Finally, z-normalisation (zero mean and unit variance) is performed on each channel's spectrum. The final N_{ch} time-frequency spectra serve as input for the deep learning model (see Figure 1). Therefore, each 8-second segment of the original time-series is represented as a $N_{ch} \times N_{FFT}$ matrix. Ground truth of each segment is the heart rate value as given by the datasets.

As argued in the previous section, we will only focus on LOSO cross-validation in the rest of this paper. The applied evaluation scheme on the *IEEE_Training* dataset (consisting of 12 sessions) is as follows. The dataset is randomly split into 4 folds, each containing 3 sessions. 3 folds are used for training, while the remaining fold is split into validation (2 sessions) and test data (1 session). The validation-test split is rotated, then the same procedure is repeated on each of the 4 folds. Thus, training is performed in total 12 times, so that each session serves as test data exactly once. The applied evaluation scheme on the *IEEE_Test* dataset follows a similar procedure. Training is performed 10 times, with each of the 10 sessions serving as test data exactly once. The remaining part of the dataset is randomly split each time, having 7 sessions as training and 2 sessions as validation data.

CNN-architecture
We focus on CNN architectures, as suggested in previous work on continuous periodic time-series analysis [7, 8, 13]. We investigated different network parameters in a pre-study, such as the number of filters (n_f) and filter size ($size_f$) in each convolutional layer, activation function, stride in both the convolutional ($stride_f$) and the pooling layers ($stride_p$), size of the fully connected layer (n_{fc}), dropout-rate, loss function, optimiser, etc. Moreover, we incorporated tracking into the CNN model for two reasons: The estimated heart rate on a segment highly correlates to the preceding segments' values, and heart rate tracking was successfully applied in classical methods (*SpaMaPlus, Schaeck2017*). Therefore, the size of the input matrix for a segment changes to $N_{tr} \times N_{ch} \times N_{FFT}$, where N_{tr} refers to the number of segments used together for heart rate estimation. The final architecture of our CNN model consists of the following layers (see Figure 1):

- convolution, $n_f^1 = 8$, $size_f^1 = (1, 1)$, $stride_f^1 = (1, 1)$
- convolution, $n_f^2 = 16$, $size_f^2 = (N_{tr}, 3)$, $stride_f^2 = (1, 1)$
- max-pooling, $size_p^2 = (1, 2)$, $stride_p^2 = (1, 2)$
- for $i = 1...N_L$:
  - convolution, $n_f^i = 2^{i+4}$, $size_f^i = (1, 3)$, $stride_f^i = (1, 1)$
  - max-pooling, $size_p^i = (1, 2)$, $stride_p^i = (1, 2)$
- convolution, $n_f^{last} = 32$, $size_f^{last} = (1, 1)$, $stride_f^{last} = (1, 1)$
- flattening layer
- fully connected layer, with n_{fc}^1 neurons
- dropout layer, with dropout-rate: 0.5
- fully connected layer, with $n_{fc}^2 = 1$ neuron

The first convolutional layer performs fusion of the input channels (PPG and accelerometer spectra), and the second convolutional layer performs fusion of the segments involved in heart rate tracking. The subsequent convolution-pooling layers serve to increase the learning capability of the model, as demonstrated by the results below. The last convolutional layer is included to reduce the input dimension of the fully connected layer, leading to less model parameters and a shorter training time. Exponential linear unit (ELU) [3] is used as activation function in the entire model. The parameters N_L, n_{fc}^1, and N_{tr} are further investigated in the evaluation-section below. The last fully connected layer outputs one value, the estimated heart rate. The loss function is defined as the absolute difference between this output and the respective segment's ground truth value. Finally, Adam [12] is used as optimiser.

All deep learning models described in this paper were implemented in TensorFlow, version 1.3 [1]. Training and evaluation were done on Nvidia GTX 1080 TI GPUs with $12GB$ of RAM. For each test run, 15000 training iterations with a batch size of 128 were performed. This proved to be sufficient for reaching convergence of the validation error. For testing, we used the model with the lowest validation error. We repeated each experiment 7 times, randomly generating the training and validation sets each time (applying the evaluation scheme as described above).

Evaluation
We performed a thorough evaluation to investigate the effect of the following key parameters of our CNN-architecture:

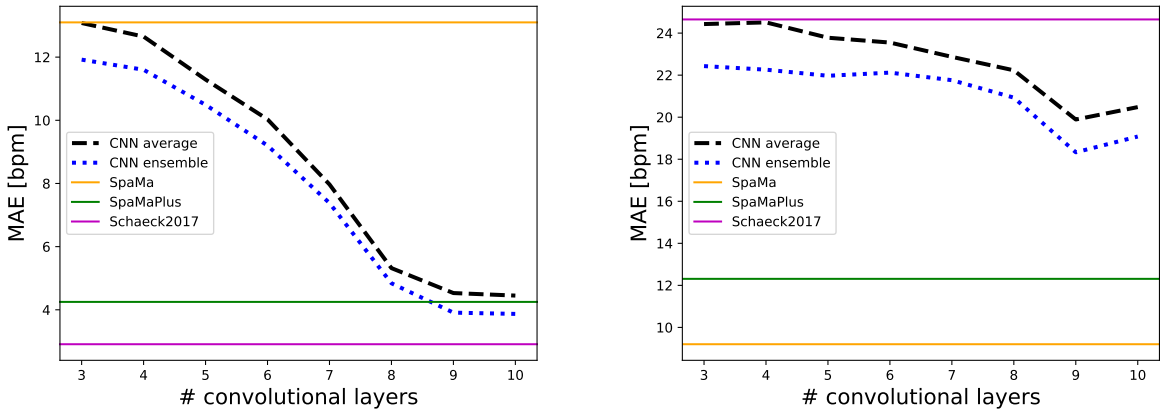

Figure 2: Evaluation results on *IEEE_Training* (left) and *IEEE_Test* (right) datasets, investigating the number of convolutional layers and ensemble prediction. LOSO results of the classical methods displayed for reference.

N_L, n^1_{fc}, and N_{tr}. Moreover, we investigated the effect of ensemble prediction and batch normalisation. Our findings are reported below.

Number of convolutional layers: We considered the range $N_L = 1...8$ (corresponding to $3...10$ convolutional layers in total when including the first two layers of the architecture). For this experiment, $n^1_{fc} = 512$ and $N_{tr} = 7$ were fixed (best setting according to our pre-study). The evaluation results on both datasets are displayed in Figure 2. Overall, each additional layer increased the learning capabilities of the model, peaking at $N_L = 7$. Therefore, we used $N_L = 7$ for the experiments presented below.

Ensemble prediction: For each dataset-session as test data, results from 7 repetitions (each providing a heart rate value per segment) are available. These were achieved with differently trained models, due to random initialisation and training-validation split. Therefore, since capturing more data variability during training, combining these models could lead to increased modelling capabilities [6]. We defined ensemble prediction as the mean value of the 7 repetitions. Results in Figure 2 show that combining the models decreases MAE by $6 - 14\%$. For example, with $N_L = 7$: from $4.53bpm$ to $3.91bpm$ on *IEEE_Training* and from $19.89bpm$ to $18.33bpm$ on *IEEE_Test*, respectively.

Size of the first fully connected layer: We considered the range $n^1_{fc} = 128...1024$, evaluation results are shown in Table 2. This parameter seems to have a marginal effect on the overall model performance. Thus, we kept $n^1_{fc} = 512$ for the further experiments.

Batch normalisation: We added batch normalisation [9] after each convolutional and the first fully connected layer, results are shown in Table 2. While the effect on the dataset

Table 2: Evaluation of size of the first fully connected layer (n_{fc}^1) and batch normalisation (BN). Average and ensemble prediction results are given, based on the 7 repetitions.

	IEEE_Training		IEEE_Test	
	CNN average	CNN ensemble	CNN average	CNN ensemble
FC size: 128	5.26	4.58	20.67	18.62
FC size: 256	5.04	4.41	20.79	19.64
FC size: 512	4.53	3.91	19.89	18.33
FC size: 1024	4.70	4.09	20.17	18.34
BN + FC size: 512	4.58	4.00	18.27	16.51

IEEE_Training is only marginal, a significant improvement can be observed on the *IEEE_Test* dataset. Thus, we included batch normalisation for the further experiments.

Number of tracking segments: We considered the range $N_{tr} = 1...11$. Results are not displayed due to brevity, and due to the only marginal effect on the model's performance. However, one benefit of using $N_{tr} > 1$ is that heart rate estimation becomes smoother over time, which might be beneficial in practical applications. Thus, we consider $N_{tr} = 7$ a good choice.

Overall, the best performing deep learning model is using the CNN-architecture as described above (see Figure 1), incorporating batch normalisation and ensemble prediction, and applying the following parameter settings: $N_L = 7$, $n_{fc}^1 = 512$, and $N_{tr} = 7$. Results of this CNN-model are the following: MAE$= 4bpm$ on *IEEE_Training* and MAE$= 16.51bpm$ on *IEEE_Test*, respectively. These results are comparable to the ones achieved with *SpaMaPlus*, which provided the best combined results on the two datasets (see Table 1).

Conclusion

We introduced deep learning for PPG-based heart rate estimation with time-frequency spectra. We investigated different CNN-architectures, and showed that they achieve comparable results to classical methods. Considering the evaluation results of most approaches, the *IEEE_Test* dataset seems to be more challenging. This can be explained by the fact that it includes recordings from different activities (such as boxing or arm rehabilitation exercises), while only consisting of 10 sessions. Especially for deep learning approaches, the available data per activity seems to be insufficient. Overall, the topic of PPG-based heart rate estimation still needs further investigation, both on the algorithmic (e.g. considering various signal representations as input or deep learning architectures other than CNNs) and the evaluation levels. The generalisation capabilities of different approaches should be further explored, especially by introducing larger datasets recorded under real-life conditions. Further challenges also include exploiting large amounts of unlabelled data (available since PPG-sensors can be easily worn during everyday life, while acquiring ground truth is more difficult) and personalisation approaches, as defined in similar domains [14].

REFERENCES

1. M. Y. Abadi and others. 2016. TensorFlow: A System for Large-scale Machine Learning. In *12th USENIX conference on Operating Systems Design and Implementation (OSDI)*. 265–283.
2. Apple. 2018. Apple Watch Series official website. (2018). Retrieved July 20, 2018 from `https://www.apple.com/lae/watch/`
3. D.-A. Clevert, T. Unterthiner, and S. Hochreiter. 2015. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *ArXiv e-prints* (2015).
4. IEEE Signal Processing Cup. 2015. Heart Rate Monitoring During Physical Exercise using Wrist-Type Photoplethysmographic (PPG) Signals. (2015). Retrieved July 20, 2018 from `https://sites.google.com/site/researchbyzhang/ieeespcup2015`
5. H. Gjoreski, J. Bizjak, M. Gjoreski, and M. Gams. 2016. Comparing Deep and Classical Machine Learning Methods for Human Activity Recognition using Wrist Accelerometer. In *IJCAI-16 Workshop on Deep Learning for Artificial Intelligence (DLAI)*.
6. Y. Guan and T. Ploetz. 2017. Ensembles of Deep LSTM Learners for Activity Recognition using Wearables. In *ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*.
7. N. Y. Hammerla, S. Halloran, and T. Ploetz. 2016. Deep, Convolutional, and Recurrent Models for Human Activity Recognition using Wearables. In *International Joint Conference on Artificial Intelligence (IJCAI)*. 1533–1540.
8. J. Hannink, T. Kautz, C. F. Pasluosta, K.-G. Gasmann, J. Klucken, and B. M. Eskofier. 2017. Sensor-Based Gait Parameter Extraction With Deep Convolutional Neural Networks. *IEEE Journal of Biomedical and Health Informatics* 21, 1 (2017).
9. S. Ioffe and C. Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *ArXiv e-prints* (2015).
10. R. Jaafar and M. A. A. Rozali. 2017. Estimation of Breathing Rate and Heart Rate from Photoplethysmogram. In *6th International Conference on Electrical Engineering and Informatics (ICEEI)*.
11. B. S. Kim and S. K. Yoo. 2006. Motion Artifact Reduction in Photoplethysmography using Independent Component Analysis. *IEEE Transactions on Biomedical Engineering* 53, 3 (2006), 566–568.
12. D. P. Kingma and J. Ba. 2014. Adam: A Method for Stochastic Optimization. *ArXiv e-prints* (2014).
13. S. Muenzner, P. Schmidt, A. Reiss, M. Hanselmann, R. Stiefelhagen, and R. Duerichen. 2017. CNN-based Sensor Fusion Techniques for Multimodal Human Activity Recognition. In *International Symposium on Wearable Computers (ISWC)*. 158–165.
14. T. Ploetz and Y. Guan. 2018. Deep Learning for Human Activity Recognition in Mobile Computing. *Computer* 51, 5 (2018).
15. M. R. Ram, V. M. Madhav, E. H. Krishna, N. R. Komalla, and K. A. Reddy. 2012. A Novel Approach for Motion Artifact Reduction in PPG Signals based on AS-LMS Adaptive Filter. *IEEE Transactions on Instrumentation and Measurement* 61, 5 (2012), 1445–1457.

16. A. Reiss and D. Stricker. 2012. Creating and benchmarking a new dataset for physical activity monitoring. In *Proceedings of 5th Workshop on Affect and Behaviour Related Assistance (ABRA)*.

17. S. Salehizadeh, D. Dao, J. Bolkhovsky, C. Cho, Y. Mendelson, and K. Chon. 2015. A Novel Time-varying Spectral Filtering Algorithm for Reconstruction of Motion Artifact Corrupted Heart Rate Signals During Intense Physical Activities using a Wearable Photoplethysmogram Sensor. *Sensors* 16, 1 (2015).

18. Samsung. 2018. Samsung Simband official website. (2018). Retrieved July 20, 2018 from `https://www.simband.io/`

19. T. Schaeck, M. Muma, and A. M. Zoubir. 2017. Computationally Efficient Heart Rate Estimation During Physical Exercise using Photoplethysmographic Signals. In *25th European Signal Processing Conference (EUSIPCO)*.

20. S. P. Shashikumar, A. J. Shah, Q. Li, G. D. Clifford, and S. Nemati. 2017. A Deep Learning Approach to Monitoring and Detecting Atrial Fibrillation using Wearable Technology. In *IEEE EMBS International Conference on Biomedical Health Informatics (BHI)*. 141–144.

21. Z. Zhang. 2015. Photoplethysmography-based Heart Rate Monitoring in Physical Activities via Joint Sparse Spectrum Reconstruction. *IEEE Trans. Biomed. Eng.* 62, 8 (2015), 1902–1910.

22. Z. Zhang, Z. Pi, and B. Liu. 2015. TROIKA: A General Framework for Heart Rate Monitoring using Wrist-type Photoplethysmographic Signals During Intensive Physical Exercise. *IEEE Trans. Biomed. Eng.* 62, 2 (2015), 522–531.