Self-supervised Remote Monitoring of Heart Rate from Videos

Divij Gupta, Ali Etemad

Dept. ECE and Ingenuity Labs Research Institute
Queen's University, Kingston, Canada
{gupta.d, ali.etemad}@queensu.ca

Abstract

Remote heart rate estimation by analyzing videos has recently become more feasible as a result of advances in deep learning. To reduce the reliance of video representation learning for remote heart rate monitoring along with improved performance, we introduce a contrastive learning solution which makes use of various augmentations of the original input videos to learn robust spatiotemporal video representations. We propose the use of 3 spatial and 3 temporal augmentations for training an encoder through our contrastive framework, followed by utilizing the late-intermediate embeddings of the encoder for remote photoplethysmogram and heart rate estimation. Our experiments on a publicly available dataset showcase the improvement of our proposed approach over related works as well as several supervised learning baselines, as our results approach the state-of-the-art.

1 Introduction

Photoplethysmography (PPG) is an optical measurement that indicates the changes in blood volume. While conventionally PPG is measured through an oximeter worn by the user on a finger, studies have shown that blood flow, and consequently, PPG, can also be measured from afar (Wieringa, Mastik, and van der Steen 2005), since the blood flow causes subtle color variations on the surface of the skin. This process, termed remote PPG (rPPG), has become more robust in recent years as a result of advances in computer vision and deep learning (Špetlík, Franc, and Matas 2018; Liu and Yuen 2020).

A major limitation of supervised deep learning solutions is the reliance on huge amounts of *annotated* data for proper training. To address this, self-supervised learning has lately begun to gain momentum in the field of deep learning. The central concept behind this paradigm is to generate pseudo-labels instead of human-annotated labels, which would then be used to train the model. These pseudo-labels are often derived by performing various augmentations (transformations) on the available data. The model is then trained to recognize these augmentations, for instance, by detecting that two different transformations applied to the same input sample are indeed renditions of the *same* information. This will allow the network to learn to extract informative representations from the input data itself.

In this work, to provide an effective approach for rPPG estimation, while reducing reliance on output labels we propose a self-supervised learning solution. We believe, given the scarcity of datasets in the field of rPPG, our method provides a valuable avenue for tackling this problem. Our model uses a 3D convolution-based encoder to obtain representations of facial videos through self-supervised contrastive learning. The model is then fine-tuned for rPPG signal estimation, achieving strong results. Our contributions can be summarized as follows. (**1**) We propose an effective solution based on self-supervised contrastive learning for the task of rPPG estimation and ultimately remote heart rate (HR) calculation. (**2**) We perform thorough experiments on a publicly available dataset and validate the effectiveness of our method, showing that our solution approaches the state-of-the-art. Moreover, further experiments demonstrate that the proposed self-supervised approach performs robustly when the amount of *labeled* data for training is reduced.

2 Related Work

rPPG Estimation. There have been several prior works using deep learning for rPPG estimation from facial videos. (Špetlík, Franc, and Matas 2018; Liu and Yuen 2020) used CNN architectures comprising vanilla convolutions to estimate the rPPG signals. (Hu et al. 2021) used spatiotemporal strip pooling in the last layers of their CNN architecture to provide for attention in the feature maps. (Chen and McDuff 2018; Zhao et al. 2021) proposed using two-stream networks where the current frame (appearance) and its normalized difference with the next frame (motion) were processed in two different CNN pathways with intermediate fusions to provide attention to the motion stream based on the appearance. Besides these, a number of non-deep learning methods as proposed by (De Haan and Jeanne 2013; Wang et al. 2016) use color space transformations and signal processing approaches to model the rPPG signals. In general, the deep learning approaches above achieve effective performances as shown in Table 2. Nonetheless, they employ fully supervised methods, heavily relying on the output labels for training.

Self-Supervised Learning. Self-supervised learning relies on generating pseudo-labels for pre-training the neural networks prior to fine-tuning for downstream tasks. Contrastive learning is a type of self-supervised learning which has gained momentum in the past few years and has shown

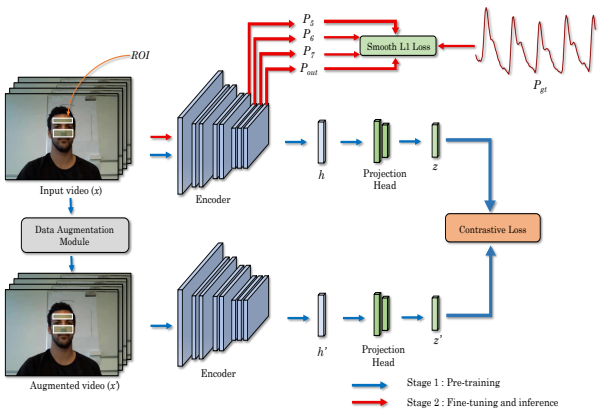

Figure 1: The overall schematic of the proposed two-stage approach.

Table 1: A detailed layout of the encoder architecture used across all experiments.

Layer Name	Kernel Size	Output Size
Input	-	128×64×64×3
Conv1	16, [1,5,5]	128×62×62×16
AvgPool	[1,2,2]	128×31×31×16
ConvBlock1	32, [3,3,3] 32, [3,3,3]	128×31×31×32 128×31×31×32
AvgPool	[1,2,2]	128×15×15×32
ConvBlock2	64, [3,3,3] 64, [3,3,3]	128×15×15×64 128×15×15×64
AvgPool	[1,2,2]	128×7×7×64
ConvBlock3	64, [3,3,3] 64, [3,3,3] 64, [3,3,3]	128×7×7×64 128×7×7×64 128×7×7×64
Global AvgPool	1, [1, 7, 7]	128×1×1×64
Squeeze	-	128× 64
Output	1, [1]	128×1

tremendous improvement across various computer vision tasks such as object detection (Sun et al. 2021), facial expression recognition (Roy and Etemad 2021), etc. SimCLR (Chen et al. 2020) proposed the use of different augmentations to create pseudo-samples of the original data and train the network to learn features to maximize the similarity between the augmented counterparts of the same original sample. A number of contrastive approaches such as SimSiam (Chen and He 2021), BYOL (Grill et al. 2020), have since been proposed which also revolve around the idea of leveraging augmented samples for pre-training to give improved results on the downstream task. Nevertheless, self-supervised approaches, including contrastive learning, have not been widely used for the task of rPPG estimation.

3 Proposed Method

In this section, we present the proposed method in detail. Our proposed solution consists of several components, namely, the encoder, the contrastive learning framework, and the loss functions. We also describe the implementation details along with data pre-processing and metrics.

3.1 Proposed Self-supervised Framework

Encoder. We use a simple 3D CNN architecture as our encoder. The detailed architecture of which is given in Table 1. Each convolution operation is followed by a ReLU activation and batch-normalization. Mathematically, for any input x, $h = Enc(x)$, where $Enc(\cdot)$ is the encoder and h is the intermediate embedding of x.

Projection Head. Following the encoder, we use a projection head to map the obtained embedding onto a lower-dimensional space. To this end, we use a 2-layer dense neural network with 64 and 16 neurons. The final embedding, z is given by $z = Proj(h)$, where $Proj(\cdot)$ is the projection head. The framework is depicted in Figure 1.

Data Augmentation Module. This module applies a set of augmentations to the input video clip x with M frames, $x_1, x_2, ...x_M$, to generate x' which also consists of M frames. In the proposed method, we use two categories of augmentations: (*i*) spatial, and (*ii*) temporal. In terms of spatial augmentations, we employ the following. (1) *Rotation*, where all the frames are rotated by the same angle $\theta \in N$, where θ is chosen randomly from $\{1, 2, ..., 360\}$; (2) *Crop*, where for a frame with height H and width W, we randomly select a cropping scale $\gamma \in R$ from [0.25, 0.75], and choose the cropping window anchor with coordinates $i \in N$ and $j \in N$. The anchor coordinates are chosen randomly from $\{0, 1, ... , W - \gamma W\}$ and $\{0, 1, ... , H - \gamma H\}$ respectively. Accordingly, the crop is performed between (i, j) and $(i + \gamma W, j + \gamma H)$, followed by resizing of the output to $W \times H$; (3) *Flip*, where every frame is flipped along the vertical axis. Mathematically, for pixel value with coordinate (i, j) in frame x'_m from x' and corresponding frame x_m from x, we have $x'_m(i, j) = x_m(W - i, j)$.

As mentioned, we also perform temporal augmentations, as follows. (1) *Shuffle*, where frames $x_1, x_2, ...x_M$ are shuffled randomly to obtain x' with a different order of frames $x_{1'}, x_{2'}, x_{3'}, ...x_{M'}$; (2) *Reorder*, where a random index r is selected to cut the video into two clips $x_a = x_1, x_2, ..., x_{r-2}, x_{r-1}$; $x_b = x_r, x_{r+1}, x_{r+2}, ...x_{M-1}, x_M$. x' is then synthesized as $x' = [x_b, x_a]$. (3) *Reverse*, where the order of the frames is reversed to obtain $x' = [x_M, x_{M-1}, ..., x_2, x_1]$.

In our experiments, input x and its augmented counterpart x' make up a *positive* pair between which the similarity is to be maximized with contrastive learning (to be presented in Section 3.2). Alternatively, for two different input clips x_1 and x_2, where $x_1 \neq x_2$, the samples (x_1, x_2), (x'_1, x_2), (x_1, x'_2), and (x'_1, x'_2) constitute the *negative* pairs.

3.2 Loss Functions

Given the two stages in our proposed method (contrastive pre-training and downstream fine-tuning), we use a different loss function for each stage.

Contrastive Loss. As in (Chen et al. 2020), the loss helps in learning representations that maximize the similarity between positive pairs while minimizing the similarity between negative samples. For any pair (m, n) with corre-

sponding projections (z_m, z_n), the cosine similarity is given by:

$$cosine(z_m, z_n) = \frac{z_m^T z_n}{||z_m||.||z_n||}. \quad (1)$$

Subsequently, the loss function is given as:

$$Loss^{con}(z_m, z_n) = -log \frac{exp(cosine(z_m, z_n)/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq m]} exp(cosine(z_m, z_k)/\tau)}, \quad (2)$$

where $1_{[k \neq m]} \in \{0, 1\}$ is the indicator function and outputs 0 iff $k = i$ and 1 otherwise. Also, τ is the temperature hyper-parameter and $2N$ is the total number of samples resulting from augmenting the original N samples.

Smooth L1 Loss. We use the smooth L1 (SL1) loss for the second stage of training. This loss is a combination of both L1 and the L2 losses, by switching between the two depending upon the difference between the amplitude values of the rPPG signal P_{out}, and the ground-truth PPG signal P_{gt}. The L2 loss is quite sensitive to large errors due to its square operation, thus the loss changes to L1 loss for large errors. This allows us to utilize the best of both losses while ensuring smoother training. This loss is given by:

$$\mathcal{L}(P_{out}, P_{gt}) = \begin{cases} \frac{1}{2}(P_{out} - P_{gt})^2, & |P_{out} - P_{gt}| < 1 \\ |P_{out} - P_{gt}| - \frac{1}{2}, & otherwise. \end{cases} \quad (3)$$

We apply this loss to the output embeddings of the final four convolutional layers as opposed to only the final layer, which enables for more effective representations to be learned throughout different parts of our network. Accordingly, our final loss for stage 2 is as:

$$Loss^{SL1} = \mathcal{L}(P_{out}, P_{gt}) + \alpha \sum_{i=\lambda 1}^{\lambda 2} \mathcal{L}(P_i, P_{gt}), \quad (4)$$

where α, $\lambda 1$, and $\lambda 2$ are the hyper-parameters for the weights and the intermediate layers prior to the final layer included in the loss calculation, set to 0.5, 5, and 7 respectively.

3.3 Pre-processing

Since observable changes in blood flow, and thus rPPG, is stronger around the forehead and the cheek regions (Datcu et al. 2013), we detect and crop these regions of interest (ROI) using the Dlib-face Detector (King 2009) (see Figure 1). Next, we concatenate these regions spatially for each frame and resize the outcome to 64×64 pixels. For each video, we use a sliding window with a length of 128 frames and a stride of 8 frames to obtain several smaller clips. Similarly, we segment the ground-truth PPG signals such that the time-synchronicity between each video clip and the corresponding PPG segment is maintained.

3.4 Metrics

After obtaining the estimated rPPG signals, similar to (Špetlík, Franc, and Matas 2018; Liu and Yuen 2020) we calculate the HR by measuring the largest peak obtained from the Welch power spectrum. Next, mean absolute error (MAE) and root mean square error (RMSE), both in beats per minute (bpm), along with correlation (*R*) of the estimated HR, are used for evaluation.

Table 2: Comparison of various methods on the COHFACE dataset.

Method	*MAE*↓	*RMSE*↓	*R*↑
CHROM (De Haan and Jeanne 2013)	7.8	12.45	0.26
POS (Wang et al. 2016)	13.43	17.05	0.07
HR-CNN (Špetlík, Franc, and Matas 2018)	8.10	10.78	0.29
DeepPhys (Chen and McDuff 2018)	6.89	13.89	0.34
DeeprPPG (Liu and Yuen 2020)	3.07	7.06	0.86
CNN+Att. (Hu et al. 2021)	5.19	7.52	0.68
CDConv+Att. (Zhao et al. 2021)	1.71	3.57	0.96
Sup. (3D)	2.62	4.59	0.90
Sup. (3D w/ res.)	4.04	7.68	0.74
Sup. (R(2+1)D)	2.71	4.53	0.91
Ours (Self-Sup.)	2.16	3.61	0.94

3.5 Implementation

The batch size for stage 1 and stage 2 of the training was set to 16 and 8, where we trained the model for 50 and 10 epochs, respectively. The learning rates were set to 1e-4 and 2e-4 for stages 1 and 2, respectively, with Adam as the optimizer for both. After pre-training, the projection head was discarded, and the model was fine-tuned for rPPG estimation. All the codes were written in PyTorch and run on an NVIDIA GTX 2080 Ti GPU.

4 Experiments and Results

Dataset. We used COHFACE (Heusch, Anjos, and Marcel 2017) for our experiments. It comprises of 160 facial videos and their corresponding PPG obtained from 40 subjects. We used data of 24 subjects for training and 16 for testing, as designated by the original authors of the dataset.

Results. Table 2 presents the results of our experiments. We implement three supervised models using 3D convolutions, R(2+1)D (Tran et al. 2018), and residual skip connections (He et al. 2016), as baselines. These baselines are similar to our main proposed approach, with two exceptions: (1) the architectures of the encoders, and (2) the supervised nature of the baselines vs. the self-supervised nature of our framework. We observe that our proposed self-supervised method outperforms all the baselines demonstrating the clear benefits of the self-supervised contrastive aspect of our approach. Moreover, the comparison of our method with prior works on rPPG shows that our method approaches the state-of-the-art. In Table 3, we present the impact of using different augmentations in the contrastive pre-training. As discussed, we experiment with spatial and temporal augmentations. In the case of spatial augmentations, *flip* provides the best results across all the metrics, while for temporal augmentations, *shuffle* shows the best performance.

Lastly, to further illustrate the advantage of using self-supervised learning vs. fully supervised, we compare the two approaches on reduced amounts of *labeled* data. Figure 2 presents the performance for the best spatial and temporal augmentations when used to train the model with different amounts of labeled data. As we observe, both self-supervised augmentation approaches lead to more robustness (suffer smaller drops in performance) when dealing

Table 3: Comparison of the results obtained for different augmentations used in the proposed approach.

Category	Augmentations	*MAE*↓	*RMSE*↓	*R*↑
Spatial	Crop	2.96	4.44	0.90
	Flip	2.16	3.61	0.94
	Rotation	2.70	4.63	0.89
Temporal	Reverse	2.67	4.48	0.90
	Reorder	2.59	4.32	0.91
	Shuffle	2.22	3.67	0.94

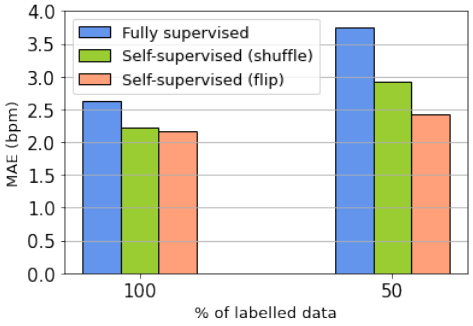

Figure 2: Performance of fully supervised and proposed self-supervised method on reduced amounts of labeled data.

with reduced labels.

5 Conclusion and Future Work

In this work, we proposed the use of contrastive learning for remote PPG estimation and remote HR prediction from facial videos. We showcased that introducing contrastive learning as a pre-training measure helps in more robust learning of the network for the downstream tasks. It also reduces the reliance on having large amounts of *labeled* data, which is often required for proper training of fully supervised approaches. Our comprehensive experiments show that our self-supervised approach outperforms many fully supervised techniques to obtain results close to the state-of-the-art while remaining more consistent when the amount of labeled data are reduced. In future, we may expand the set and also explore the combination of different augmentations. It would also be interesting to explore the generalizability of the proposed approach on other remote PPG datasets. Also, we may introduce concepts such as attention mechanisms to our encoder and explore other loss functions for the different components of the pipeline.

References

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. *International Conference on Machine Learning*, 1597–1607.

Chen, W.; and McDuff, D. 2018. Deepphys: Video-based physiological measurement using convolutional attention networks. *European Conference on Computer Vision*, 349–365.

Chen, X.; and He, K. 2021. Exploring simple siamese representation learning. *IEEE Conference on Computer Vision and Pattern Recognition*, 15750–15758.

Datcu, D.; Cidota, M.; Lukosch, S.; and Rothkrantz, L. 2013. Non-contact automatic heart rate analysis in visible spectrum by specific face regions. *International Conference on Computer Systems and Technologies*, 120–127.

De Haan, G.; and Jeanne, V. 2013. Robust pulse rate from chrominance-based rPPG. *IEEE Transactions on Biomedical Engineering*, 60(10): 2878–2886.

Grill, J.-B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P. H.; Buchatskaya, E.; Doersch, C.; Pires, B. A.; Guo, Z. D.; Azar, M. G.; et al. 2020. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Heusch, G.; Anjos, A.; and Marcel, S. 2017. A reproducible study on remote heart rate measurement. *arXiv preprint arXiv:1709.00962*.

Hu, M.; Qian, F.; Wang, X.; He, L.; Guo, D.; and Ren, F. 2021. Robust Heart Rate Estimation with Spatial-Temporal Attention Network from Facial Videos. *IEEE Transactions on Cognitive and Developmental Systems*.

King, D. E. 2009. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*, 10: 1755–1758.

Liu, S.-Q.; and Yuen, P. C. 2020. A general remote photoplethysmography estimator with spatiotemporal convolutional network. *IEEE International Conference on Automatic Face and Gesture Recognition*, 481–488.

Roy, S.; and Etemad, A. 2021. Spatiotemporal Contrastive Learning of Facial Expressions in Videos. *arXiv preprint arXiv:2108.03064*.

Špetlík, R.; Franc, V.; and Matas, J. 2018. Visual heart rate estimation with convolutional neural network. *British Machine Vision Conference*, 3–6.

Sun, B.; Li, B.; Cai, S.; Yuan, Y.; and Zhang, C. 2021. FSCE: Few-shot object detection via contrastive proposal encoding. *IEEE Conference on Computer Vision and Pattern Recognition*, 7352–7362.

Tran, D.; Wang, H.; Torresani, L.; Ray, J.; LeCun, Y.; and Paluri, M. 2018. A closer look at spatiotemporal convolutions for action recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 6450–6459.

Wang, W.; den Brinker, A. C.; Stuijk, S.; and De Haan, G. 2016. Algorithmic principles of remote PPG. *IEEE Transactions on Biomedical Engineering*, 64(7): 1479–1491.

Wieringa, F. P.; Mastik, F.; and van der Steen, A. F. 2005. Contactless multiple wavelength photoplethysmographic imaging: A first step toward "SpO 2 camera" technology. *Annals of Biomedical Engineering*, 33(8): 1034–1041.

Zhao, Y.; Zou, B.; Yang, F.; Lu, L.; Belkacem, A. N.; and Chen, C. 2021. Video-Based Physiological Measurement Using 3D Central Difference Convolution Attention Network. *IEEE International Joint Conference on Biometrics*, 1–6.