SEQUENCE-RPPG: A FAST BVP SIGNAL EXTRACTION METHOD FROM FRAME SEQUENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

Non-contact heart rate estimation has essential implications for the development of affective computing and telemedicine. However, existing deep learning-based methods often endeavor to achieve real-time measurements, so a simple, fast, pre-processing-free approach is needed. Our work consists of two main parts. Firstly, we proposed SEQ-rPPG, which first transforms the RGB frame sequence into the original BVP signal sequence by learning-based linear mapping and then outputs the final BVP signal using 1DCNN-based spectral transform, and time-domain filtering. Secondly, to address the shortcomings of the existing datasets in training the model, a new large-scale dataset was collected for training and testing. Our approach achieved competitive results on the collected large dataset(the best) and public dataset UBFC-rPPG(0.81 MAE with 30s time window, test only). It requires no complex pre-processing, has the fastest speed, can run in real-time on mobile ARM CPUs, and can achieve real-time beat-to-beat performance on desktop CPUs. Benefiting from the high-quality training set, other deep learning-based models reduced errors by at least 53%. We compared the methods with and without the spectral transformation, and the results show that the processing in the time domain is effective.

1 INTRODUCTION

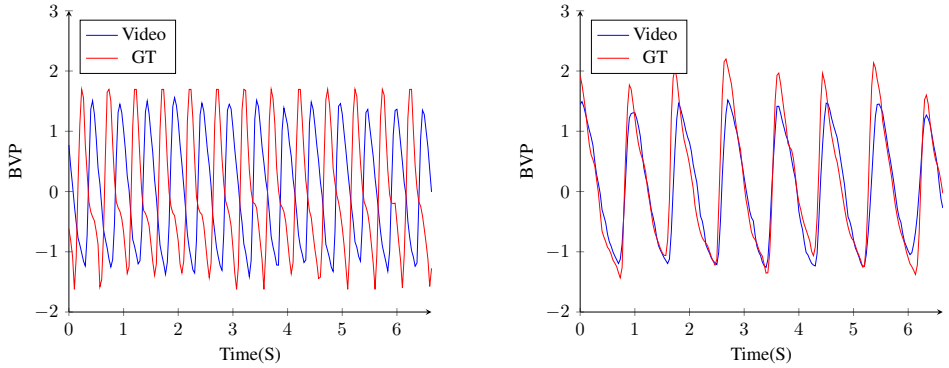
Camera-based blood volume pulse (BVP) is a non-contact physiological measurement used to extract physiological signals from the heart. The most common method of BVP signal measurement is by photoplethysmogram (PPG). However, PPG requires the subject to wear an optical contact sensor, which can increase cost, cause discomfort and limit the scenarios in which it can be used. Some work(Takano & Ohta, 2007; Verkruysse et al., 2008) has successfully used this principle to extract physiological signals from facial videos.Early work focused on temporal modeling, Poh et al. (2010a) blind source separation method for extracting BVP signals from RGB signals by independent component analysis (ICA), In Poh et al. (2010b), they added the detrending operation and increased the robustness. de Haan & Jeanne (2013) propose a new color space and separates the luminance and chromaticity components, and the algorithm extracts the BVP signal in the chromaticity space, so it is robust to luminance. Wang et al. (2017b) define the color space as orthogonal to skin tone to accommodate different skin tones based on skin color. Tulyakov et al. (2016) and Bobbia et al. (2019) separate present classical approach to spatial modeling. They attempt to select the part of the face with the highest signal-to-noise ratio from different regions of the face so that it can accommodate more head movements, illumination and shading. Deep learning-based approach yields better accuracy, but some deep learning-based methods require complex pre-processes. In Niu et al. (2020a), Niu et al. (2020b), Song et al. (2021) and Lu et al. (2021), the pre-processing algorithm first performs facial landmarks annotation, then segments the facial image according to the landmarks, transform the color space, and convert the result to spatio-temporal map. Some deep learning-based small models are of particular interest, which facilitates applications on low-cost devices.Chen & McDuff (2018) use a 2D convolution to process the differential frames and obtain the differential BVP signal, using convolution attention(CAN) to select the region of interest. Liu et al. (2020) introduced time-shift module and implemented multitasking through common intermediate features, and performance tests show that the method can run in real-time on mobile ARM CPUs. Liu et al. (2021) upgraded input size, used self-attention convolution to eliminate appearance branches, and introduced the transformer layers. The transformer layers do not improve

accuracy or speed compared with the convolution-based network, and speed tests on the Raspberry Pi show that convolution-based efficientPhys-C improves performance by one-third compared with TS-CAN. Yu et al. (2019) proposed a 3D convolution-based method that serves as a baseline for simple 3D convolution. Speth et al. (2021) applied dilated convolution to 3D convolution-based model, and the results show that the time receptive field has a large impact on the accuracy, dilated convolution reduces the number of parameters while expanding the field of receptive, showing the potential of convolution-based models for miniaturization and real-time. Botina-Monsalve et al. (2022) implement fast real-time rPPG based on 3D convolution and discuss the impact of different input sizes. Since too large an input resolution reduces the spatial perceptual field, it is better to use small resolutions (e.g. 8x8) for small models. Comas et al. (2022) proposed a small convolution-based network, which uses 2D convolution and differential temporal convolution(DTC) for spatio-temporal modeling, using only 5.26K parameters. Pre-processing is generally considered to degrade algorithm performance and make the application of the algorithm more difficult. However, different pre-processing algorithms cannot be generalized, and all methods require facial detection. The facial detection process may be implicit. Some of the algorithms include a facial image segmentation step, and these pre-processing cannot balance accuracy and speed, facial image segmentation can limit its deployment on mobile devices because it is currently difficult to achieve on mobile devices effectively. The performance of face detection, calculation of differential and average frames, color space conversion, and generation of spatio-temporal maps depend on the advanced level of the pre-processing algorithm. In case of insufficient optimization, these processes can significantly slow down the whole algorithm, however it is possible to optimize it fully. There are many libraries available for fast facial detection like openface, seetaface and mediapipe etc., some of these can run effectively on mobile devices. Other processes, which are almost exclusively numerical computations, can be reduced to negligible time consumption by properly using numerical libraries. Therefore these pre-processes add complexity to the development, but do not limit the algorithm in terms of performance. Ideally, a simple and efficient approach is the most desired.

Earlier classical methods could usually output the BVP signal frame by frame, so there was hardly any delay. Chen & McDuff (2018) is based on spatial modeling, and the differential BVP signal can be obtained based only on the difference between two frames, so the method also has almost no delay. Although some methods (e.g. Liu et al. (2020; 2021)) can output heart rate in real time, but they, like the vast majority of deep learning-based methods, are based on temporal modeling. These methods require the input of a batch of video frames, and then the neural network will correlate the frames at different times to extract the BVP signal. Therefore, an entire batch of video frames is fed each time the algorithm is executed. According to our tests, Liu et al. (2020) can get high rates of over 100fps on the Raspberry Pi 4B, but its input batch size is 128 frames, which means it has a latency of over 1s. To get a real-time BVP signal, the latency should be as small as possible. It is feasible to reduce the input batch size as well as to increase the speed of model execution, our model achieves real-time BVP signal output on desktop CPUs by significantly increasing the speed, and after appropriate tailoring, the same speed can be achieved on mobile ARM CPUs.

Deep learning-based models are generally trained and validated on several public datasets. For example, the small and medium size datasets such as PURE(Stricker et al., 2014), UBFC-rPPG(Bobbia et al., 2019), UBFC-PHYS(Meziati Sabour et al., 2021),MMSE-HR(Zhang et al., 2016), AFRL(Estepp et al., 2014), contains limited themes, monotonous scenes and simple head movements. the large size datasets such as OBF(Li et al., 2018), MAHNOB-HCI(Wang et al., 2017a), VIPL-HR(Niu et al., 2018), contain a larger number of subjects.OBF includes 106 subjects, all videos are recorded in good lighting conditions without complex head movements, over 176 hours in total. MAHNOB-HCI contains hundreds of videos of subjects performing human-computer interaction, encoded using highly compressed H264 and with only a few head movements and lighting changes. VIPL-HR contains videos of hundreds of subjects and includes a variety of scenes with short videos of head movements, but the videos do not have a stable frame rate. Deep learning-based model training requires high-quality ground truth BVP signals. The common problem is the random shift between the ground truth BVP signal and the video frames. The random shift can severely degrade the performance of the commonly used MSE and MAE loss functions (Yu et al., 2019; Zhan et al., 2020). This problem is discussed in Yu et al. (2019), where they used Negative Pearson(NP) correlation coefficient as the loss to get an improvement. Botina-Monsalve et al. (2022) proposed

Signal-to-Noise Ratio(SNR) loss and used in conjunction with NP loss, called NPSNR loss. Comas et al. (2022) proposed Temporal Adaptive Location Shift (TALOS) loss, which is translation invariant compared with NP loss. To train models with high accuracy and high real-time performance, eliminating the random shift of the ground truth signal would be a key task. McDuff et al. (2022) synthesize a large complex dataset containing accurate physiological signals, which indicates that a synthetic training set may be the way to solve this problem. Fundamentally, we would like to eliminate the delay in dataset recording, so we wrote an application that simultaneously reads the ground truth signal from the blood pulse meter and the RGB signal from the webcam and records all the information with UNIX timestamp. Based on this application, we collected a large dataset and trained models on it. The results show that our proposed model achieves competitive results in cross-validation, while TS-CAN(Liu et al., 2020) and PhysNet(Yu et al., 2019) also significantly improve accuracy by training on our high-quality training sets, especially PhysNet, which obtains a heart rate mean absolute error(MAE) of 0.72 (30-second time window) on UBFC-rPPG.



(a) UBFC-rPPG dataset

(b) Collected dataset

Figure 1: BVP signal extracted from UBFC-rPPG video and our collected video. Random shift do not occur in our videos but are common in UBFC-rPPG.

2 Related Work

2.1 End-to-end Physiological Signal Measurement Network

Many studies focus on simplifying the pre-processing process or even removing it, and are vague in their end-to-end definitions, and they all claim to be end-to-end models. In DeepPhys(Chen & McDuff, 2018) and MTTS-CAN(Liu et al., 2020), the pre-processing algorithms generate differential and average frames, respectively, then enter them into motion branch and appearance branch of the network. In EfficientPhys(Liu et al., 2021), they use a pre-processing approach similar to MTTS-CAN, yet within the model and determine the normalization parameters by learning. It is hard to say if this is a pre-processing or part of the model internals. In PhysNet(Yu et al., 2019) and PhysFormer(Yu et al., 2022), the model accepts original image input directly without additional pre-processing. MTTS-CAN is an improved version of DeepPhys, which has higher accuracy, and although they both require pre-processing, the pre-processing is simple and requires little additional cost. PhysFormer and PhysNet are both end-to-end models. However, the number of parameters and computation of PhysFormer is much larger than the latter, which is not enough to reflect the advantages of lightweight and real-time. So MTTS-CAN and PhysNet are representative end-to-end models, and we will compare the proposed method with both.

2.2 Fast Fourier Convolution

Chi et al. (2020) roposed Fast Fourier Convolution (FFC), which can obtain an extensive range of receptive fields at a shallow layer of the network. It improves accuracy on multiple tasks and datasets by replacing the convolutional layers in general convolutional networks with FFC layers. Suvorov et al. (2022) applies FFC to a wide range of image restoration, showing superior performance, especially for periodic images. Shchekotov et al. (2022) applied FFC to speech and audio signal

denoising, significantly improving the baseline method and reducing the number of parameters. FFC is particularly sensitive to periodic signals and will significantly affect work that deals with periodic signals, such as audio noise reduction and rPPG. The difference between the above work and ours is that we do not model spatially and do not divide the time domain channel and frequency domain channel. We use simple time domain channels and skip them by residual connection. In this view, SEQ-rPPG does not model the video. We filter the noise from the 1D sequence and extract the BVP signal.

3 METHOD

To extract the RGB signal from the frame sequence, SEQ-rPPG performs two main steps: first, it linearly maps the raw 8x8 facial RGB frames to the original signal sequence. Then it uses 1D convolution to filter the original signal sequence and outputs the BVP waveform. The linear mapping is done inside the model, which means that this method does not require pre-processing. Unlike ICA(Poh et al., 2010a;b), CHROM(de Haan & Jeanne, 2013), POS(Wang et al., 2017b), which are dimensionality reduction methods, the parameters of the linear mapping module are obtained by learning, and after training it will have the appropriate weights to combine the RGB signals.

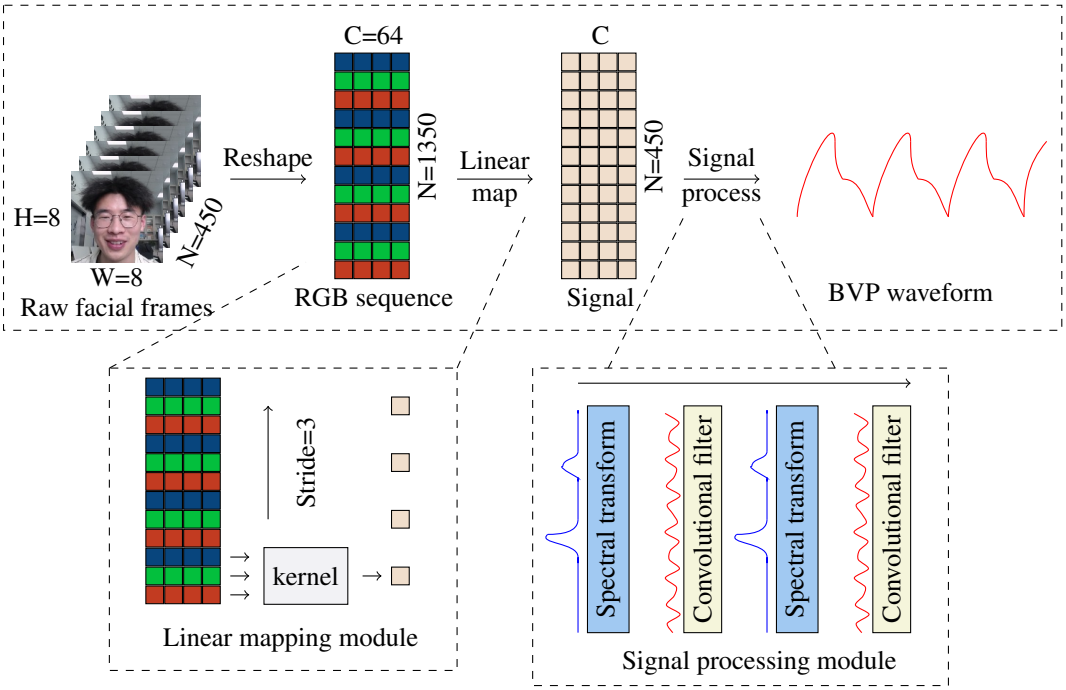


Figure 2: Overview of our framework.

3.1 LINEAR MAPPING MODULE

Given an RGB facial video input $\mathbf{X} \in \mathbb{R}^{N \times W \times H \times C}$, first transpose it to $\mathbf{X} \in \mathbb{R}^{N \times C \times W \times H}$, ensure channel is in the second dimension, then reshape it to $\boldsymbol{X} \in \mathbb{R}^{NC \times WH}$. $\boldsymbol{X}$ consists of N sets of RGB signals, the original signal is converted from RGB as $Y_n = \sum_{i=1}^{WH} \boldsymbol{m}_i \times (R_i, G_i, B_i)^T + b_i$, where $\boldsymbol{m}$ is a row vector, which maps RGB colors to real numbers, b is the bias. The transformation of colors can be achieved by convolution without activation layers, it uses a linear convolution layer with kernel size = 3 and stride = 3, linear mapping outputs multi-channel sequences, the number of channels depends on the number of convolution kernels.

3.2 SIGNAL PROCESSING MODULE

For the given original signal, it is usually filtered using several convolutional layers. This filtering is performed in the time domain, which means that adjacent signal points will be strongly correlated, while distant signal points will be weakly correlated. For periodic signals, considering the correlation between similar periods helps filter noise from a wider range. From the signal processing point of view, classical BVP signal extraction usually requires the use of filters to filter the noise. Time-domain filters such as Gaussian filters, moving average filters, and detrending. Frequency domain filters are commonly bandpass filters to extract signals at specific frequencies. In summary, classical methods contain alternating use of frequency domain filters and time domain filters. We use the spectral transform layer to complete the processing in the frequency domain, while the traditional 1D convolution is used for the time domain processing. For a signal $\boldsymbol{Y} \in \mathbb{R}^{N \times C}$, the spectral filtering layer first performs a Real Fast Fourier Transform(RFFT) on each channel, obtain frequency domain signal $\boldsymbol{Y}_{Freq} \in \mathbb{Z}^{\frac{\lfloor N+1 \rfloor}{2} \times C}$, then it is decomposed into a real part $\boldsymbol{Y}_{Real}$ and a imaginary part $\boldsymbol{Y}_{Img}$, they are combined on the channel as $\boldsymbol{Y}_{Comb} \in \mathbb{R}^{\frac{\lfloor N+1 \rfloor}{2} \times 2C}$. Then a convolution layer is applied to it and re-decomposes the output into real and imaginary parts. It is converted to complex numbers and recovered to the time domain signal by Inverse Real Fast Fourier Transform(IRFFT). The output signal is mixed with the original signal through the residual connection, the number of channels remains constant throughout the process. The final signal is obtained by alternating spectral transform and 1D convolution processing.

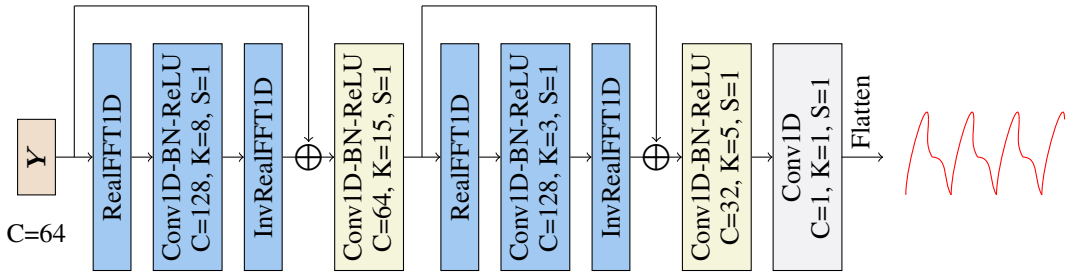


Figure 3: Signal processing module.

4 DATASET

To collect the dataset, we wrote a private data capture program to obtain raw data from Logitech C930c webcam and Contech CMS50E Oximeter, respectively. Webcam with two capture modes, 1920*1080 resolution, MJPG encoding, and 640*480 resolution, YUY2 encoding, each mode has the frame rate of 30fps. MJPG is a common fast intra-frame compression format, YUY2 is a format for raw YUV images that uses 4 bytes to store 2 Y-components, 1 U-component and 1 V-component. There are three video encoding formats, RAW RGB, MJPG, and H264. RAW RGB is the original lossless compressed RGB format, and H264 is the common inter-frame compression format on the Internet, multiple formats are recorded simultaneously. Ground truth BVP waveform data read from Oximeter, sampling rate is 20 per second. During data recording, subjects were required to complete a series of tasks or watch a video, and subjects were not required to keep their heads stable, so they may have experienced larger head movements. After completing the assigned task, the subject will take a short break and experimenter will assign him/her the next task. All of the 58 subjects were students of Asian descent, mostly master's students, and some girls may wear makeup.

Table 1: Data collection workflows

Sub-dataset	Task or induction video	Duration(S)	Camera codec	Video codec	Resolution
rPPG	Relaxed	120	YUY2	RGB,MJPG,H264	640×480
	Relaxed & Dark	120			
	Play a game	120			
	Read an article	120			
Emotion	Natural scenery	120	MJPG	MJPG,H264	1920×1080
	Puzzle game	180			
	Comedy	120			
	Illusion picture	20			
	Academic paper	60			
	Video about yawn	60			
Engagement	Video-based learning	240	MJPG	MJPG,H264	1920×1080
	Textbook-based learning	480			
	Watch a public class	420			

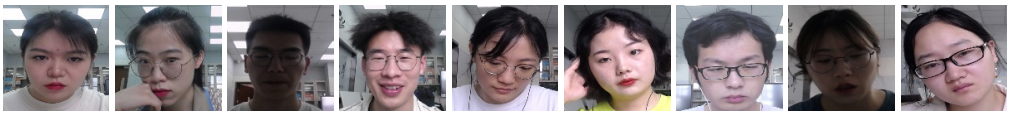


Figure 4: Overview of our dataset.

5 EXPERIMENTS

5.1 TRAINING DATA & TESTING DATA

Since our model cannot be trained effectively on other datasets, we only trained it on the collected dataset, excluding a small amount of data with low frame rate or wrong format due to insufficient performance during recording. Although we acquired the data in multiple video encoding formats, only the MJPG format was used for the entire training and testing. We randomly divided the 58 subjects into 5 folds, the first 4 folds were used as the training set and the last fold was used as the testing set, and all videos of the same subject would only appear in the same fold. The test is divided into two parts, first on the whole testing set and then on the rPPG subset, which contains more complex scenarios with motion. Models were also tested on the UBFC-rPPG public dataset, which includes 42 videos from 42 subjects of approximately 1 minute each, all recorded at resolution of 640x480, frame rate of 30 fps, in RAW RGB format, and all faces were kept stable.

5.2 PERFORMANCE METRICS

We evaluated the accuracy and speed of the model separately, where the accuracy metrics include Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Pearson correlation coefficient (ρ). For the speed test, we calculated the number of parameters, FLOPs and memory usage of the model, and tested the inference speed(ms) on desktop GPU, desktop CPU, and mobile CPU. We recorded the hardware load to indicate the efficiency of the model's utilization of the hardware when the model was running at the highest speed.

Table 2: Theoretical parameters and FLOPs

Model	Parameters(K)	FLOPs(M)
SEQ-tiny	3.46	1.49
SEQ-T	152	136
SEQ-FT	266	157
TS-CAN	533	6682
PhysNet	770	6945

5.3 IMPLEMENTATION & EXPERIMENT DETAILS.

To compare with Liu et al. (2020) and Yu et al. (2019) on the collected data set, We reproduced the TS-CAN directly using the source code provided by Liu et al. (2020). Since Tensorflow performance is usually higher than Pytorch, to be fair in performance tests, we implemented PhysNet using Tensorflow according to the details provided in Yu et al. (2019). We have three models:

SEQ-FT with alternating convolution in the frequency and time domains, SEQ-T with convolution in the time domain only, and the streamlined SEQ-tiny. All models are trained and tested in the same Tensorflow environment, the version is 2.6.0, compiled with cuda 11.6, cudnn 8.3, AVX2. Hardware platforms include Nvidia Geforce RTX 3080 10G (GPU), AMD Ryzen9 5950x 16-core (CPU), ARM Cortex A72 4-core (CPU), when tested on ARM platform, also used Tensorflow 2.6.0, but does not include any acceleration support for GPUs or x86 CPUs. To train efficiently, we cut the video using a moving window with a step size of 3 seconds and used mediapipe to perform face detection in the middle of the frame (sometimes other people appeared in the background but were already processed). Finally, only the facial images were fed to the model. Although Liu et al. (2020) claims to be a pre-processing-free model, face detection had to be performed because of the large movements in the collected dataset. All models used the same data during training and testing, differing only in resolution. In addition, due to the lack of enough high heart rate samples in the collected dataset, we randomly selected a part of the segments with interval skipping half of the frames and did the same for the BVP waveform, which finally made the heart rate twice as high. The additional data accounted for about 25% and were evenly distributed in the training set. TS-CAN uses the Nadam optimizer with default parameters,MAE loss, batch size=128, epochs=3. PhysNet uses the SGD optimizer with learning rate=0.005, NP loss, batch size=32, epochs=8. our model uses the adam optimizer with default parameters, MSE loss, batch size=32, epochs=24. Classical methods and TS-CAN used a 0.75 to 2.5 Hz bandpass filters in the test. Neither our models nor PhysNet applied filters. CHROM, POS and ICA(de Haan & Jeanne, 2013; Wang et al., 2017b; Poh et al., 2010b) use the code provided in Boccignone et al. (2022) and Boccignone et al. (2020). The final heart rate is extracted from the peak of the signal power spectrum, ranging from 0.5 to 4 Hz. In calculating the error and Pearson correlation coefficients, we used a 30-second moving average window with 1-second step.

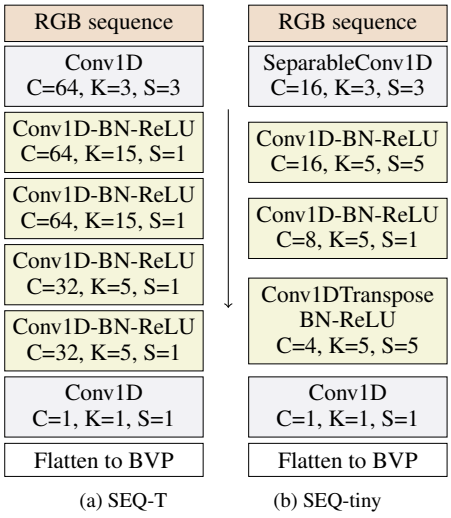


Figure 5: Overview of SEQ-T & SEQ-tiny.

5.4 Experimental Results

The rPPG subset is more difficult because it has more light changes and facial activity. SEQ-FT achieved the best accuracy on the collected dataset, with second place going to TS-CAN. The results show that SEQ-FT is more accurate than SEQ-T, and the spectral transformation is effective. We compared the results of different models tested on UBFC-rPPG, which were trained on different training sets. For our models, they behave similarly and both have high accuracy, although the difference in the number of SEQ-FT and SEQ-tiny parameters is very large, but their accuracy is similar, which may be caused by the stable head and stable lighting environment of UBFC-rPPG. For TS-CAN and PhysNet, they train better on the collected dataset than on other datasets, In particular, the results trained on the collected dataset outperform those trained on UBFC-rPPG, which indicates that a high-quality training set helps improve the model accuracy. Among all the cross-dataset results, the best is TS-CAN trained on the collected dataset, and the second best is PhysNet trained on the collected dataset.

5.5 Computation Cost

We tested the computational overhead of the model on different platforms. Unlike the time spent per frame in Liu et al. (2020), we calculated the time spent to process a whole batch, which is more reflective of the latency caused by the computation. In past work, little attention was paid to the size of the input scale. Usually, the input is resized and a float32 numerical matrix is obtained. If the input size is large, then the memory usage of the input data may be higher than the model itself. The memory usage of the model is a theoretical minimum based on the number of parameters, which will be smaller than the actual value. We tested the average inference time of the model on a large

Table 3: Results on collected dataset

Method	Entire dataset			rPPG subset		
	MAE↓	RMSE↓	ρ↑	MAE↓	RMSE↓	ρ↑
SEQ-tiny	3.57	6.98	0.72	5.70	9.82	0.60
SEQ-T	1.77	4.69	0.85	3.16	6.69	0.72
SEQ-FT	**0.95**	**2.12**	**0.96**	**1.55**	**3.29**	**0.89**
TS-CAN	1.39	2.47	0.95	2.44	3.60	0.86
PhysNet	3.13	6.61	0.78	4.94	9.80	0.56
CHROM	7.73	10.8	0.43	8.50	10.0	0.12
POS	3.91	6.03	0.70	5.10	7.10	0.43
ICA	11.4	13.5	0.29	12.2	14.7	0.17

Table 4: Results on UBFC-rPPG

Method	Train dataset	MAE↓	RMSE↓	ρ↑
SEQ-tiny	Collected	0.82	1.21	0.99
SEQ-T	Collected	0.93	1.47	0.99
SEQ-FT	Collected	0.81	1.21	0.99
PhysNet	Collected	0.72	1.16	0.99
PhysNet	UBFC-rPPG	2.95	3.67	0.98
TS-CAN	Collected	0.69	0.95	0.99
TS-CAN	PURE	1.47	2.31	0.99
TS-CAN	UBFC-rPPG	1.70	2.72	0.99
EfficientPhys-C	PURE	2.04	3.06	0.99
EfficientPhys-C	UBFC-rPPG	1.14	1.81	0.99
PulseGAN	PURE & BSIPL-rPPG	2.09	4.42	0.97
PulseGAN	UBFC-rPPG	1.19	2.10	0.99
DualGAN	PURE	0.74	1.02	0.99
DualGAN	UBFC-rPPG	0.44	0.67	0.99

amount of data as the time-use result, device load is also of concern, and the slow inference speed of the model on some devices may be due to the inability to use computational resources efficiently, resulting in a low load state on the device. Although both low load and high computational volume result in slower speeds, the energy consumed and heat generated are not the same, and smaller computational volume means lower heat generation and more energy efficiency, which is especially important on battery-powered devices. In the test results, our model has a very significant speed advantage, being much faster and consuming much less power than its predecessor. It is worth noting the difference in performance between SEQ-FT and SEQ-T. On the GPU, SEQ-FT is slightly slower than SEQ-T, which is close to the difference in FLOPs; on the desktop CPU, SEQ-FT is significantly slower than SEQ-T, yet the former is also significantly less loaded than the latter, which can be speculated to be the result of the inability of RFFT and IRFFT to parallelize effectively on the desktop CPU; On mobile CPU, the load is close, however SEQ-FT takes about four times as long as SEQ-T, which is not consistent with FLOPs' estimates, RFFT and IRFFT may cost more overhead on ARM CPU, due to the fact that AVX2 instruction set support was turned on during desktop CPU testing, while no accelerated instruction set was available on ARM platforms, which may be the reason for the slower speed of SEQ-FT on the ARM platform. TVM have been used in Liu et al. (2020) to accelerate inference, but we cannot convert models with FFT operations while keeping the model structure unchanged, which we will try to solve by optimizing the model structure.

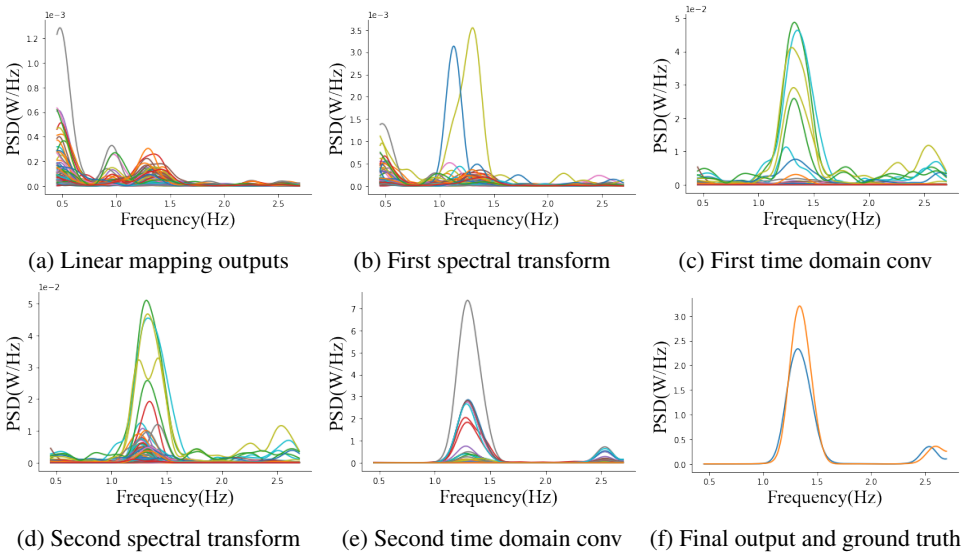
5.6 Visualization & Discussion

SEQ-FT was chosen for visualization, and we randomly selected a video to feed into the model, and the intermediate feature maps were captured and used for visualization whenever possible. Each feature map can be viewed as a multi-channel signal sequence, and its Power Spectral Density (PSD)

Table 5: Computation cost on different platforms

Method	Batch frames	Memory usage		Desktop GPU		Desktop CPU		Mobile CPU	
		Model (KB)↓	Input (KB)↓	Time (ms)↓	Load↑	Time (ms)↓	Load↑	Time (ms)↓	Load↑
SEQ-tiny	450	13.8	346	1.1	97%	0.7	12%	6.3	40%
SEQ-T	450	608	346	1.7	99%	2.4	50%	34	72%
SEQ-FT	450	1064	346	2.2	99%	12	17%	146	61%
TS-CAN	128	2132	1991	9.1	99%	79	77%	1260	91%
PhysNet	128	3080	1573	9.7	99%	48	79%	1290	89%

is calculated for each channel. The 0.5-2.5 Hz portion is intercepted for visualization. A significant heart rate peak is observed in the output from the linear mapping. However, a considerable amount of noise power is also present in the PSD. As the features go deeper, the noise is contained, and some details start to appear, such as subpeaks in the high-frequency region. The final output PSD is very close to the ground truth. Through feature visualization, our model is well interpreted, and our visualization of temporal modeling demonstrates the role of convolution networks in another dimension than Liu et al. (2020) and Niu et al. (2020a)'s focus on spatial modeling.



(a) Linear mapping outputs (b) First spectral transform (c) First time domain conv

(d) Second spectral transform (e) Second time domain conv (f) Final output and ground truth

Figure 6: Power Spectral Density(PSD) of the feature maps.

6 Conclusions

In terms of accuracy, our model performs similarly to or better than existing models; in terms of speed, our model reaches several times more than current models and can run effectively on mobile CPUs. On Raspberry Pi 4B, our highest accuracy model inference takes 146ms, and the fastest model takes 6.3ms, compared with TS-CAN's 1260ms. However, SEQ-FT does not perform as expected on CPUs, and it is worth investigating how to make the spectral transform run more efficiently on CPUs. The quality of our dataset is better than UBFC-rPPG and PURE. The accuracy of existing models can be improved significantly by training on the collected dataset, e.g., the MAE of TS-CAN decreases from 1.47 (trained on PURE) to 0.69; the MAE of PhysNet decreases from 2.95 (trained on UBFC-rPPG) to 0.72, and we expect the collected dataset to be a baseline for training sets. However, the proposed method is sensitive to the random shift present in the dataset, and we will try to deal with this issue in the future.

7 REPRODUCIBILITY STATEMENT

Although the steps required to reproduce the results have been described in this paper, there are factors that significantly affect the results that need to be highlighted in this section. This model has not been finely tuned to the parameters, and its accuracy is not very sensitive to the parameters but to the training set. When working with datasets, there were some key operations: First we implemented face detection and tracking. However, face detection frames are usually difficult to keep stable. We added a displacement threshold of size 25 pixels when implementing face tracking, any movement at a distance less than the threshold will be filtered, and greater than the threshold will move the detection frame in a sudden change, thus avoided introducing periodic noises. Secondly, when making the training set, we made additional high heart rate samples with heart rates in the range of 120-180, representing about 25% of the total number of samples. The samples were obtained by skipping half of the frames at intervals, so the heart rate would be twice as high, which is exceptionally important for training the model to correctly identify the high heart rate samples; if enough high heart rate samples are not correctly generated in the training set, then great errors may occur by chance when estimating the high heart rate test set samples. Finally, we ensured that the BVP signal in the training set was strictly aligned with the frames by the recorded UNIX timestamps. The above steps were applied to all models we implemented.

8 ETHICS STATEMENT

For the collected dataset, data were collected from a total of 58 subjects, who clearly understood that their data were being collected and consented to these data being shared with other research institutions and being used for scientific research only. The entire dataset contained the subjects' videos, BVP signals, affective experience scales, learning engagement scales, gender, age, grade, and major. There was no other identifying information.

REFERENCES

Serge Bobbia, Richard Macwan, Yannick Benezeth, Alamin Mansouri, and Julien Dubois. Unsupervised skin tissue segmentation for remote photoplethysmography. *Pattern Recognition Letters*, 124:82–90, 2019.

Giuseppe Boccignone, Donatello Conte, Vittorio Cuculo, Alessandro D'Amelio, Giuliano Grossi, and Raffaella Lanzarotti. An open framework for remote-PPG methods and their assessment. *IEEE Access*, pp. 1–1, 2020. doi: 10.1109/access.2020.3040936. URL https://doi.org/10.1109/access.2020.3040936.

Giuseppe Boccignone, Donatello Conte, Vittorio Cuculo, Alessandro D'Amelio, Giuliano Grossi, Raffaella Lanzarotti, and Edoardo Mortara. pyvhr: a python framework for remote photoplethysmography. *PeerJ Computer Science*, 8:e929, 2022.

Deivid Botina-Monsalve, Yannick Benezeth, and Johel Miteran. Rtrppg: An ultra light 3dcnn for real-time remote photoplethysmography. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 2146–2154, June 2022.

Weixuan Chen and Daniel McDuff. Deepphys: Video-based physiological measurement using convolutional attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

Lu Chi, Borui Jiang, and Yadong Mu. Fast fourier convolution. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 4479–4488. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/2fd5d41ec6cfab47e32164d5624269b1-Paper.pdf.

J. Comas, A. Ruiz, and F. Sukno. Efficient remote photoplethysmography with temporal derivative modules and time-shift invariant loss. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2181–2190, Los Alamitos, CA, USA,

jun 2022. IEEE Computer Society. doi: 10.1109/CVPRW56347.2022.00237. URL https://doi.ieeecomputersociety.org/10.1109/CVPRW56347.2022.00237.

Gerard de Haan and Vincent Jeanne. Robust pulse rate from chrominance-based rppg. *IEEE Transactions on Biomedical Engineering*, 60(10):2878–2886, 2013. doi: 10.1109/TBME.2013.2266196.

Justin R. Estepp, Ethan B. Blackford, and Christopher M. Meier. Recovering pulse rate during motion artifact with a multi-imager array for non-contact imaging photoplethysmography. In *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1462–1469, 2014. doi: 10.1109/SMC.2014.6974121.

Xiaobai Li, Iman Alikhani, Jingang Shi, Tapio Seppanen, Juhani Junttila, Kirsi Majamaa-Voltti, Mikko Tulppo, and Guoying Zhao. The obf database: A large face video database for remote physiological signal measurement and atrial fibrillation detection. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pp. 242–249, 2018. doi: 10.1109/FG.2018.00043.

Xin Liu, Josh Fromm, Shwetak Patel, and Daniel McDuff. Multi-task temporal shift attention networks for on-device contactless vitals measurement. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 19400–19411. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/e1228be46de6a0234ac22ded31417bc7-Paper.pdf.

Xin Liu, Brian L. Hill, Ziheng Jiang, Shwetak Patel, and Daniel McDuff. Efficientphys: Enabling simple, fast and accurate camera-based vitals measurement, 2021. URL https://arxiv.org/abs/2110.04447.

Hao Lu, Hu Han, and S. Kevin Zhou. Dual-gan: Joint bvp and noise modeling for remote physiological measurement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12404–12413, June 2021.

Daniel McDuff, Miah Wander, Xin Liu, Brian L Hill, Javier Hernandez, Jonathan Lester, and Tadas Baltrusaitis. Scamps: Synthetics for camera measurement of physiological signals. *arXiv preprint arXiv:2206.04197*, 2022.

Rita Meziati Sabour, Yannick Benezeth, Pierre De Oliveira, Julien Chappe, and Fan Yang. Ubfc-phys: A multimodal database for psychophysiological studies of social stress. *IEEE Transactions on Affective Computing*, pp. 1–1, 2021. doi: 10.1109/TAFFC.2021.3056960.

Xuesong Niu, Hu Han, Shiguang Shan, and Xilin Chen. Vipl-hr: A multi-modal database for pulse estimation from less-constrained face video. In *Asian conference on computer vision*, pp. 562–576. Springer, 2018.

Xuesong Niu, Shiguang Shan, Hu Han, and Xilin Chen. Rhythmnet: End-to-end heart rate estimation from face via spatial-temporal representation. *IEEE Transactions on Image Processing*, 29: 2409–2423, 2020a. doi: 10.1109/TIP.2019.2947204.

Xuesong Niu, Zitong Yu, Hu Han, Xiaobai Li, Shiguang Shan, and Guoying Zhao. Video-based remote physiological measurement via cross-verified feature disentangling. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (eds.), *Computer Vision – ECCV 2020*, pp. 295–310, Cham, 2020b. Springer International Publishing. ISBN 978-3-030-58536-5.

Ming-Zher Poh, Daniel J McDuff, and Rosalind W Picard. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. *Optics express*, 18(10):10762–10774, 2010a.

Ming-Zher Poh, Daniel J McDuff, and Rosalind W Picard. Advancements in noncontact, multiparameter physiological measurements using a webcam. *IEEE transactions on biomedical engineering*, 58(1):7–11, 2010b.

Ivan Shchekotov, Pavel Andreev, Oleg Ivanov, Aibek Alanov, and Dmitry Vetrov. Ffc-se: Fast fourier convolution for speech enhancement, 2022. URL https://arxiv.org/abs/2204.03042.

Rencheng Song, Huan Chen, Juan Cheng, Chang Li, Yu Liu, and Xun Chen. Pulsegan: Learning to generate realistic pulse waveforms in remote photoplethysmography. *IEEE Journal of Biomedical and Health Informatics*, 25(5):1373–1384, 2021. doi: 10.1109/JBHI.2021.3051176.

Jeremy Speth, Nathan Vance, Patrick Flynn, Kevin Bowyer, and Adam Czajka. Unifying frame rate and temporal dilations for improved remote pulse detection. *Computer Vision and Image Understanding*, 210:103246, 2021. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2021.103246. URL https://www.sciencedirect.com/science/article/pii/S1077314221000904.

Ronny Stricker, Steffen Müller, and Horst-Michael Gross. Non-contact video-based pulse rate measurement on a mobile service robot. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, pp. 1056–1062, 2014. doi: 10.1109/ROMAN.2014.6926392.

Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2149–2159, January 2022.

Chihiro Takano and Yuji Ohta. Heart rate measurement based on a time-lapse image. *Medical Engineering & Physics*, 29(8):853–857, 2007. ISSN 1350-4533. doi: https://doi.org/10.1016/j.medengphy.2006.09.006. URL https://www.sciencedirect.com/science/article/pii/S1350453306001901.

Sergey Tulyakov, Xavier Alameda-Pineda, Elisa Ricci, Lijun Yin, Jeffrey F. Cohn, and Nicu Sebe. Self-adaptive matrix completion for heart rate estimation from face videos under realistic conditions. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2396–2404, 2016. doi: 10.1109/CVPR.2016.263.

Wim Verkruysse, Lars O Svaasand, and J Stuart Nelson. Remote plethysmographic imaging using ambient light. *Opt. Express*, 16(26):21434–21445, Dec 2008. doi: 10.1364/OE.16.021434. URL http://opg.optica.org/oe/abstract.cfm?URI=oe-16-26-21434.

Wenjin Wang, Albertus C. den Brinker, Sander Stuijk, and Gerard de Haan. Algorithmic principles of remote ppg. *IEEE Transactions on Biomedical Engineering*, 64(7):1479–1491, 2017a. doi: 10.1109/TBME.2016.2609282.

Wenjin Wang, Albertus C. den Brinker, Sander Stuijk, and Gerard de Haan. Algorithmic principles of remote ppg. *IEEE Transactions on Biomedical Engineering*, 64(7):1479–1491, 2017b. doi: 10.1109/TBME.2016.2609282.

Zitong Yu, Xiaobai Li, and Guoying Zhao. Remote photoplethysmograph signal measurement from facial videos using spatio-temporal networks. *arXiv preprint arXiv:1905.02419*, 2019.

Zitong Yu, Yuming Shen, Jingang Shi, Hengshuang Zhao, Philip Torr, and Guoying Zhao. Physformer: Facial video-based physiological measurement with temporal difference transformer. In *CVPR*, 2022.

Qi Zhan, Wenjin Wang, and Gerard de Haan. Analysis of cnn-based remote-ppg to understand limitations and sensitivities. *Biomed. Opt. Express*, 11(3):1268–1283, Mar 2020. doi: 10.1364/BOE.382637. URL http://opg.optica.org/boe/abstract.cfm?URI=boe-11-3-1268.

Zheng Zhang, Jeff M. Girard, Yue Wu, Xing Zhang, Peng Liu, Umur Ciftci, Shaun Canavan, Michael Reale, Andy Horowitz, Huiyuan Yang, Jeffrey F. Cohn, Qiang Ji, and Lijun Yin. Multimodal spontaneous emotion corpus for human behavior analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

A APPENDIX

A.1 PRIVACY

If the rPPG method is applied to a commercial application, the server expects to capture the user's facial image in order to optimize the user experience and improve the algorithm. However, the user's facial image involves privacy issues, and the higher the resolution image, the more privacy information it contains, so using low resolution images helps to protect privacy. Besides, low-resolution images can also reduce the network usage during uploading and improve the user experience. Our model uses 8x8 inputs, which implies a strong privacy.

(a) 128x128 (b) 64x64 (c) 32x32 (d) 8x8

Figure 7: Facial images in different resolutions.

A.2 LOW LATENCY

Since our model uses a long sequence of 450 frames of input, it usually requires a wait of 15 seconds to obtain the BVP signal. Compared with models using 128 frames of input, our model seems to have to wait longer. However, in engineering this can be optimized. For example, for a 60 frames input, we simply repeat it 8 times to get 480 frames of input and intercept 450 of them. In the output signal, the BVP waveform can be obtained by intercepting 60 frames from the middle at the corresponding position, and this operation greatly reduces the delay.

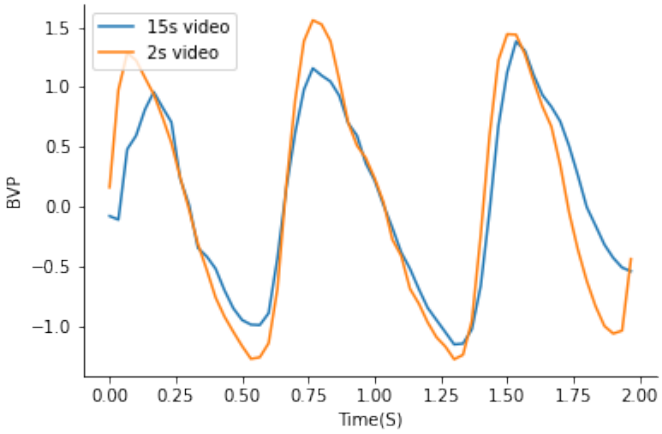


Figure 8: Extract the BVP signal from 2 seconds or 15 seconds of video in the same duration

A.3 SIGNAL QUALITY

TS-CAN must use a bandpass filter to improve signal quality, for PhysNet, additional filters are not required. the use of additional filters has almost no effect for SEQ-FT.

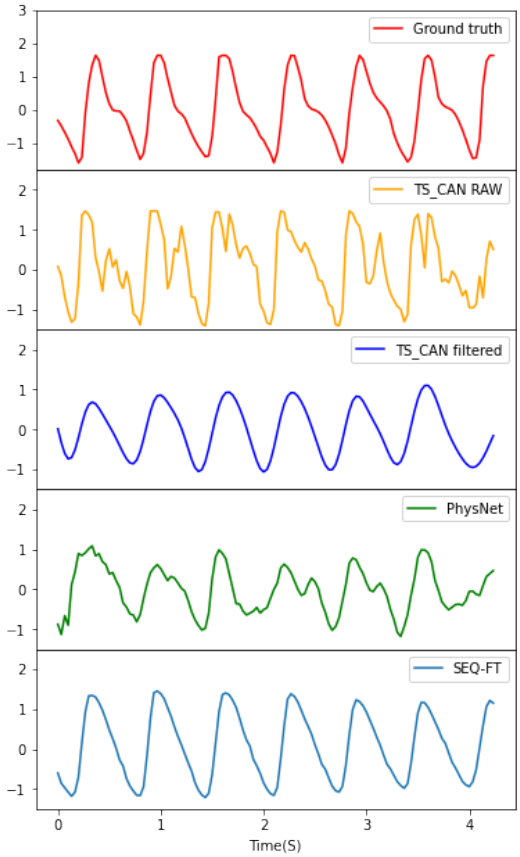


Figure 9: BVP Signal from different model

A.4 More Efficient Pre-processing

Simple pre-processing can be eliminated by reasonable engineering optimization. In the original article, the pre-processing of TS-CAN was done by numpy, and we tried to do the same using Tensorflow and move this operation inside the model. This means that we only need to input the original video after deploying the model and do not need to pay attention to the specific implementation of pre-processing, which helps it to be implemented in non-python environments (e.g. mobile or web), this operation does not affect the accuracy and makes full use of Tensorflow's powerful matrix operations.

```
# Approach of the original literature
motion, appearance = pre_process(vid) # Pre-processing with numpy
BVP = TS_CAN([motion, appearance])
```

```
.......
   def call(self, x):
      x_ = (x[1:] - x[:-1])/(x[1:]+x[:-1])
      x_ = (x_ - tf.reduce_mean(x_, axis=(0, )))/tf.math.reduce_std(x_,
          axis=(0, ))
      motion, appearance = tf.concat([x_, tf.zeros([1, *size, 3])],
          axis=0), tf.expand_dims(tf.reduce_mean(x, axis=(0, )), axis=0)
      d1, r1 = self.diff_input(motion), self.rawf_input(appearance)
      .........
# A better approach, closer to end-to-end
BVP = TS_CAN(vid)
```