Shift Robust Loss Function

Ananya Deepak Deoghare, Kimaya Milind Kulkarni

005627628, 805528337

BE 275 Fall 2022

Contents

1 Introduction **3**

2 Problem Definition **4**

2.1 About the Dataset: UBFC 4

2.2 About the Model: Physnet 5

3 Methods **6**

3.1 Approach 6

3.1.1 Baseline Loss: MSE 6

3.1.2 MSE + Max Correlation 6

3.1.3 Learning Parameter 7

3.2 Experiments 7

3.3 Codebase 8

3.4 Software Packages Used 8

4 Results **10**

4.1 Evaluation Metrics 10

4.2 Study based on max shift values 10

4.3 Analysis 10

4.4 Challenges 10

4.5 Future Steps 11

1 Introduction

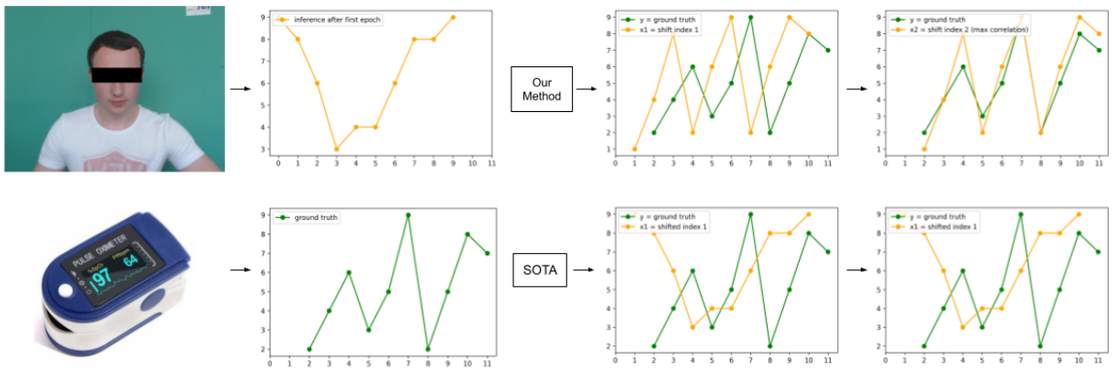
Index terms from class – *K-fold cross validation, Mean Squared Error, Mean Absolute Error, rPPG, Loss Function, PCA, Pearson Correlation*

Heart rate (HR) is an important physiological indicator that reflects the physical and mental status of the human body. In general, HR can be measured by electrocardiography (ECG) or photoplethysmography (PPG), which need to employ specific sensors to make contact with participant's skin. The COVID-19 pandemic accelerated research on remote healthcare. RPPG is a technique that uses slight color changes in a person's face to find remote-PPG signal and estimates heart rate from this signal using signal processing techniques. We use a RGB camera to obtain face videos of volunteers and a finger pulse oximeter to obtain the ground truth PPG signal while collecting data from rPPG. These two devices are not hardware triggered, so there is a slight lag between the frames obtained from the camera and PPG signal.

State-of-the-art methods require the ground truth data to be exactly aligned with the signals. For these methods to be truly practical, large-scale, diverse data is needed to train them. However, such stringent requirements make it almost impossible to scale up data collection for this purpose, since they require manual modification.

To address this problem, we propose a novel shift-robust loss function for regression that enables the network to learn from misaligned data to achieve better results than recent methods. To the best of our knowledge, this is the first work where training the model does not require ideal signals or manual alignment of the signals as the input to the network, which allows us to collect more data for a truly robust model. Our model is trained on artificially misaligned data. We used UBFC-RPPG dataset for all our experimentation.

2 Problem Definition



Comparison of our loss function with state of the art(Sota) loss function

Research in remote, equitable healthcare has boomed since the pandemic. The first step towards this is monitoring different vital signs remotely. This may include ECG, PPG, respiratory signal, SpO2, blood pressure, etc. All these vital signs are measured using different instruments. All of these instruments may lack the ability to use hardware triggers to ensure that data collection begins at the same time. Hence, shift between different signals is inevitable. Manually aligning the data is a long and inefficient way to deal with this, especially for larger datasets.

To estimate these vital signs from videos captured by different types of cameras (thermal, RGB, NIR) and sensors (Radar), it is a necessity to have methods that are robust to the shifts introduced by using different instruments to capture data. In such a scenario, it is necessary to develop a shift-robust loss function to deal with such a shift while learning the signal using deep learning networks. The architecture of our model may change depending on the input and output, but a shift robust loss function can always be used for any such application. Hence, this would be a great contribution and could also be used for applications other than healthcare.

2.1 About the Dataset: UBFC

The dataset consists of 42 RGB videos of subjects' faces with a green screen in background. The ground truth PPG data is collected using a pulse oximeter.

The advantage of UBFC dataset is that it has been recorded under ideal conditions where the subjects head is stationary, background is not busy, lighting is mostly consistent throughout the dataset, and mainly signals are perfectly aligned. They have ensured ideal conditions so that estimation of heart rate from videos is easier using different models.

Limitations:

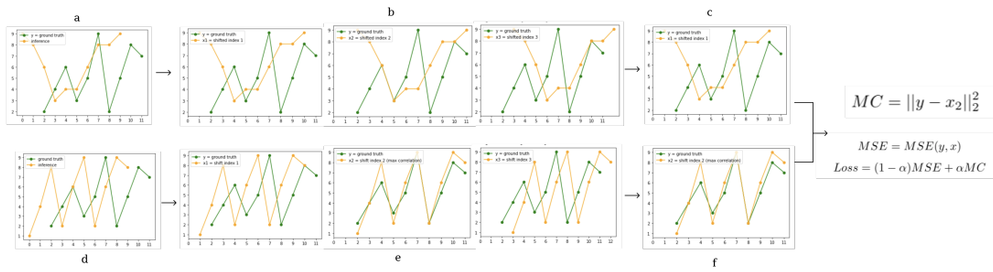
- It is a small dataset for deep learning models, hence harder for models to generalize
- It does not consist of data from diverse demographic so models trained might be biased against certain skin tones.

In our project the goal was not to make a better rPPG system, the goal was to develop a new loss function hence we needed a dataset which worked well on rPPG models and had ideal conditions such as lighting condition, alignment etc. We wanted to artificially misalign as we wanted to analyse results for different values of misalignment and that is why we chose UBFC dataset for our project.

2.2 About the Model: Physnet

Physnet is an end-to-end framework with spatio-temporal networks which is able to recover rPPG signals from raw facial videos fast and efficiently. Physnet takes 128 frames of videos as input and gives same number of rPPG signal points as output. For our loss function to work we needed a network which gave a signal as an output. That is why, we decided to use Physnet architecture instead of deepphys in our analysis.

3 Methods



Here, Fig (a), (b), (c) represent earlier epochs and (d), (e), (f) represent later epochs.
(a) and (d) Shifted ground-truth and inference signals.
(b) and (e) Shifting signals and calculating cross-correlation at each shift. In (b), randomness in early epochs causes low/ inaccurate correlation values.
(c) and (f) Choose the shifted signal with maximum correlation. In (c), the chosen signal has very low correlation.

3.1 Approach

We have proposed a loss function that can learn from misaligned signal data since current state-of-the-art methods require perfect alignment. The ground-truth signal from the pulse oximeter has a considerable lag behind the estimated signal from the relevant areas of the face. To account for this, we artificially introduced a shift in our input and target data. We then trained our model on various loss functions.

3.1.1 Baseline Loss: MSE

We chose MSE as a baseline loss function. The MSE is a gauge of an estimator's performance; numbers nearer to zero are preferable, and the MSE is always non-negative. MSE is sensitive to outliers. MSE loss is sensitive to the shift present in input and output signals as it takes mean of squared error between two signals.

3.1.2 MSE + Max Correlation

The loss function calculates the cross correlation between the two signals, the input x and the output y. The signal y is then shifted by n indices, and the cross correlation is computed for each index individually. The index where x, y had the maximum correlation is chosen as the most closely-aligned pair of signals, and the first part of the loss is calculated as

$L_1 = ||x - y||_2$.

However, the predicted output y is close to random in the first few epochs of training, causing the index of the maximum correlation between x and y to be random. To compensate for this, $L_2 = MSE(x, y)$

is calculated, which has been shown on PhysNet to accurately predict heart rate via rPPG. Thus, in the first few epochs, the loss L_2 offsets the randomness in L_1, and in the latter epochs, when y is not random, the loss L_1 compensates for the error due to misaligned data. Thus, the

$Loss = L_1 + L_2$.

Due to addition of maximum correlation loss, this loss function will be less sensitive to shift in signals. MSE will act as a stabilizing loss where max correlation loss will be random. In those epochs, MSE will make sure the model learns the signals, and as the signals become more correlated, the Max correlation part of the loss function will help the model learn better. Addition of max correlation loss will push the model to perform better and generalize well in case of new data.

3.1.3 Learning Parameter

Since L_2 is most effective at the beginning of training, and L_1 becomes more and more effective as training progresses, their contribution can be controlled by a parameter α, so that:

$$Loss = \alpha L_1 + (1 - \alpha) L_2$$

where $\alpha = \frac{E}{N}$
N = total number of epochs
E = number of completed epochs.
Since we trained for 30 epochs, we used $\alpha = 0.033*$(number of completed epochs).

In the previous loss we had just added two loss functions. We know the purpose of MSE loss is just to stabilize in first few epochs and eventually its weight should be minimum as MSE is sensitive to the shift in data and will not help the network learn. Hence, we have introduced a learning parameter which will decrease the weight of MSE as number of epochs increases, this means the sensitivity to shift in data will also decrease. The weight for max correlation will increase as number of epochs increases, this means that the function will be more and more shift robust as the number of epochs increases and it will be less sensitive to outliers and shifts.

In this case we take advantage of parts of loss function that are most beneficial in that instant.

3.2 Experiments

We perform all our experiments on misaligned UBFC dataset with Physnet128 model. The misalignment in UBFC is performed using *np.random.randint* where we specify max shift = 0, 4, 7, 10. This function gives an array of random integers which are a part of discrete normal distribution. In case of max shift = 0 we basically get aligned signals. In other cases, we get different values of misalignment for each video. We keep this constant throughout the experimentation for that specific max-shift k-fold run. Physnet128 model takes 128 frames from the videos as input and gives 128 points of rPPG signal as output.

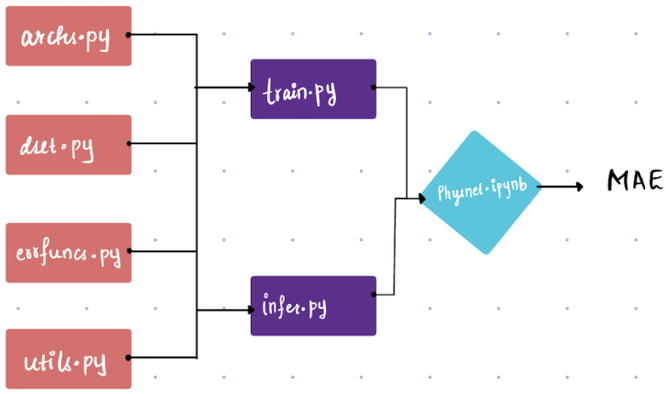
To avoid bias caused by way the data is split into train-test sets, we implemented **k-fold cross validation** $k = 5$, and averaged the **MAE** over all five runs. The proposed loss function calculates the cross correlation of input and output signals x and y at 10 different indices and selects the index with the highest correlation.

We run each experiment for 30 epochs. Then we choose a model instance and test the test videos on this model instance. To find MAE value we first find the heart rate from the rPPG

signal by finding FFT of the same and then finding the peak in the range of frequencies in which human heart rate may exist. Once we have all HRs we find MAE between these.

$HR = 60 * (frequency)$

3.3 Codebase



Flowchart of the Codebase

The codebase consists of a folder called "src" which contains files called "errfuncs.py" which has all the loss functions, "archs.py" which contains all the model architectures (Namely Physnet and Deepphys) and "dset.py" contains dataloaders for dataset. We use DALI dataloader to load our video dataset.

In the main folder there is "train.py" which basically loads the data, trains the chosen model using selected loss function on the chosen dataset. "infer.py" file tests the selected model instance on the test dataset. The main notebook 'Physnet.ipynb' is used to run everything.

The following cells exist in Physnet.ipynb:

- Preamble and Imports: Imports necessary packages
- Train Physnet: Runs train.py file, needs inputs of dataset name, path, loss function name, number of epochs
- Estimate signals of new videos: Test on videos in test set, need to specify test samples, paths, checkpoint to be used
- Obtain Results and Metrics: Finds heart rates from windowed RPPG signals, finds MAE values, and averages them.

3.4 Software Packages Used

We used various packages and tools to help us build our code. Some of the important ones are highlighted here :

- Pytorch

- numpy
- yolo
- torch vision
- math
- re
- scipy
- nvidia.dali

4 Results

4.1 Evaluation Metrics

To evaluate the performance of the network we use Mean Absolute Error (MAE). MAE is the sum of absolute differences between actual and predicted heart rate values.

$$MAE = (\frac{1}{n}) \sum_{i=1}^{n} |y_i - x_i| \tag{1}$$

4.2 Study based on max shift values

Table 1: Dependence of MAE on the Misalignment

Loss Function	**Max Shift**	**Average MAE**
MSE	10	5.946
	7	6.688
	4	5.852
	0	3.250
Max Corr + MSE	10	4.118
	7	3.324
	4	**3.116**
	0	**2.633**
(1-α)*MSE+α*(max corr), α=0.033*epoch	10	**3.136**
	7	**3.288**
	4	3.146
	0	2.685

4.3 Analysis

As can be seen from table 1, as expected the loss function with learning parameter works better than all the loss functions tried for max shift = 7, 10. However in case of no shift and max shift chosen as 4, just the addition of MSE and Max correlation give better results than the remaining loss functions. The MAE values for learning parameter loss are not far behind. For greater shift the logic of giving more importance to specific term in a loss function works better. In case of lesser shifts it does not show a large improvement.

We can confidently say that both loss functions introduced in this project work better than the baseline loss function MSE even in case of no shift. The results are in acceptable range of error. We can conclusively say that both loss functions are shift robust and can be used in **regression** applications where input and output both are signals. The learning parameter loss works better in case of greater misalignment.

4.4 Challenges

- We realized our loss function would not work on DeepPhys model as deepphys gives one point as an output, so we had to switch to Physnet.

- We tried **PCA**, but in PCA back-propagation is not possible because it's not differentiable. So we could not implement the same here.
- We tried implementing **Negative Pearson Coefficient loss function**, but the values were very bad, due to which the same have not been mentioned in the paper. (We were not able to improve those values)

4.5 Future Steps

- More experimentation is needed on different datasets and different models in rPPG to truly say that the loss function is shift robust for rPPG.
- Find more applications that may have a misalignment in their data due to use of different instruments and try out those datasets along with respective models with our loss function.