(Supplementary Material) Meta-rPPG: Remote Heart Rate Estimation Using a Transductive Meta-Learner

Eugene Lee, Evan Chen, and Chen-Yi Lee

Institute of Electronics, National Chiao Tung University, Hsinchu 30010, Taiwan
{eugenelet.ee06g,evanchen.ee06}@nctu.edu.tw, cylee@si2lab.org

1 Performing Transductive Inference During Deployment

Here, we show the algorithm for transductive inference during deployment in Algorithm 1. The inference process is similar to the typical inference of a feed-forward deep neural network (the final line). The difference is in the inclusion of adaptation steps using the first 2 seconds of the video for transductive learning prior to actual inference.

2 Performance Comparison Using Different Adaptation Steps

In this section, we study how the number of adaptation steps, L, used for transductive inference affects performance. We report results under different metrics, namely, mean absolute error (MAE), standard deviation of error (SD), root mean squared error (RMSE) and Pearson correlation coefficient (R). Tabulated performances for MAHNOB-HCI are shown in Table 1, 2, 3 and 4 for MAE, SD, RMSE and R respectively. In the same order, tabulated performances for UBFC-rPPG are shown in Table 5, 6, 7 and 8. Again, using the same order, we show comparison plots in Figure 1, 2, 3 and 4 for both MAHNOB-HCI and UBFC-rPPG.

From the results, we can deduce that the number of adaptation steps used during transductive inference should match the number of steps used during training. This rule only applies to the generation of synthetic gradients for transductive inference but not on the protypical distance minimizer. We hypothesize

Algorithm 1 Transductive Inference During Deployment

1: **Input:** $\boldsymbol{x}$: A single video stream
2: $\hat{\boldsymbol{x}}, \tilde{\boldsymbol{x}} \leftarrow \boldsymbol{x}$ ▷ $\hat{\boldsymbol{x}}$: first 2 seconds of video, $\tilde{\boldsymbol{x}}$: rest of video
3: **for** $i \leftarrow 1, L$ **do** ▷ Adaptation phase (run L steps)
4: $\quad \theta \leftarrow \theta - \alpha(\nabla_\theta \mathcal{L}_{\text{proto}}(\hat{\boldsymbol{z}}, \hat{\boldsymbol{z}}^{\text{proto}}) + f_\psi(\hat{\boldsymbol{z}}))$
5: **end for**
6: $\boldsymbol{y} \leftarrow h_\phi(f_\theta(\tilde{\boldsymbol{x}}))$ ▷ Estimation of rPPG signal using adapted feature extractor

that the prototypical distance minimizer will eventually converge to some value and doesn't hurt performance if run for infinite number of adaptation steps. This is intuitive as the idea of prototypical distance minimizer is to pull out-of-distribution samples towards the center of the distribution that is modeled by the rPPG estimator using the training data.

3 Does Joint Adaptation of both Feature Extractor and rPPG Estimator Give Better Results?

We hypothesize that the estimation of rPPG signal is more efficient if we are able to update the weights of the feature extractor during testing for the adaptation to the new observed distribution. By doing so, we expect that the features generated by the feature extractor fall within the distribution covered by the rPPG estimator, i.e. the weights of the rPPG estimator is obtained through the optimization on the training dataset. One might challenge that the joint adaptation of both feature extractor and rPPG estimator weights might result in better performance, contradicting our hypothesis. To demonstrate that our hypothesis holds, we perform an empirical study on whether the joint update approach or the sole update of the weights of the feature extractor performs better. The implementation is straight forward, the synthetic gradient generator is moved towards the output for the joint adaptation case. We show experiments on MAHNOB-HCI using an adaptation steps of 10 in Table 9. The empirical results support our hypothesis.

Table 1: Results of mean absolute error of HR measurement on MAHNOB-HCI using different adaptation steps, L.

Method	MAE of HR (bpm)				
	$L = 0$	$L = 5$	$L = 10$	$L = 20$	$L = 30$
End-to-end (baseline)	7.47	7.47	7.47	7.47	7.47
Meta-rPPG (proto only)	7.42	6.65	6.05	6.02	**6.02**
Meta-rPPG (synth only)	7.42	5.00	3.88	3.70	6.25
Meta-rPPG (proto+synth)	**7.42**	**3.89**	**3.01**	**3.02**	6.20

4 Visualization of Feature Activation Map Using Different Methods

In this section, we show the visualization for feature activation map using the methods we introduced and is compared with the baseline that uses an end-to-end supervised learning method. Ablation study is also performed by showing feature activation maps obtained using individual methods that we proposed.

Table 2: Results of average standard deviation HR measurement error on MAHNOB-HCI using different adaptation steps, L.

Method	SD of HR (bpm)				
	$L=0$	$L=5$	$L=10$	$L=20$	$L=30$
End-to-end (baseline)	**7.39**	7.39	7.39	7.39	7.39
Meta-rPPG (proto only)	7.91	6.08	6.95	6.89	**6.86**
Meta-rPPG (synth only)	7.91	5.89	5.09	4.96	7.72
Meta-rPPG (proto+synth)	7.91	**4.90**	**3.68**	**4.95**	7.81

Table 3: Results of root mean squared error of HR measurement on MAHNOB-HCI using different adaptation steps, L.

Method	RMSE of HR (bpm)				
	$L=0$	$L=5$	$L=10$	$L=20$	$L=30$
End-to-end (baseline)	**8.63**	8.63	8.63	8.63	8.63
Meta-rPPG (proto only)	8.65	6.97	6.79	6.71	**6.67**
Meta-rPPG (synth only)	8.65	5.15	3.96	4.02	7.11
Meta-rPPG (proto+synth)	8.65	**4.65**	**3.66**	**3.68**	6.94

Table 4: Results of Pearson correlation coefficient of HR measurement on MAHNOB-HCI using different adaptation steps, L.

Method	R of HR (bpm)				
	$L=0$	$L=5$	$L=10$	$L=20$	$L=30$
End-to-end (baseline)	0.70	0.70	0.70	0.70	0.70
Meta-rPPG (proto only)	0.74	0.77	0.77	0.77	**0.79**
Meta-rPPG (synth only)	0.74	0.79	0.81	0.81	0.77
Meta-rPPG (proto+synth)	**0.74**	**0.83**	**0.85**	**0.85**	0.75

Table 5: Results of mean absolute error of HR measurement on UBFC-rPPG using different adaptation steps, L.

Method	MAE of HR (bpm)				
	$L=0$	$L=5$	$L=10$	$L=20$	$L=30$
End-to-end (baseline)	**12.78**	12.78	12.78	12.78	12.78
Meta-rPPG (proto only)	13.23	9.24	8.07	7.82	**7.53**
Meta-rPPG (synth only)	13.23	11.03	7.04	9.11	11.97
Meta-rPPG (proto+synth)	13.23	**7.68**	**6.07**	**5.97**	9.82

Table 6: Results of average standard deviation HR measurement error on UBFC-rPPG using different adaptation steps, L.

Method	SD of HR (bpm)				
	$L=0$	$L=5$	$L=10$	$L=20$	$L=30$
End-to-end (baseline)	**13.70**	13.70	13.70	13.70	13.70
Meta-rPPG (proto only)	14.17	10.39	9.33	9.17	**8.23**
Meta-rPPG (synth only)	14.17	11.00	8.37	11.92	13.94
Meta-rPPG (proto+synth)	14.17	**9.01**	**7.89**	**7.12**	11.93

Table 7: Results of root mean squared error of HR measurement on UBFC-rPPG using different adaptation steps, L.

Method	RMSE of HR (bpm)				
	$L=0$	$L=5$	$L=10$	$L=20$	$L=30$
End-to-end (baseline)	**13.30**	13.30	13.30	13.30	13.30
Meta-rPPG (proto only)	14.63	10.62	9.43	9.37	**8.90**
Meta-rPPG (synth only)	14.63	13.41	8.55	11.55	14.62
Meta-rPPG (proto+synth)	14.63	**8.92**	**7.86**	**7.42**	11.21

Table 8: Results of Pearson correlation coefficient of HR measurement on UBFC-rPPG using different adaptation steps, L.

Method	R of HR (bpm)				
	$L=0$	$L=5$	$L=10$	$L=20$	$L=30$
End-to-end (baseline)	0.27	0.27	0.27	0.27	0.27
Meta-rPPG (proto only)	0.35	0.45	0.47	0.48	**0.50**
Meta-rPPG (synth only)	0.35	0.44	0.47	0.42	0.39
Meta-rPPG (proto+synth)	**0.35**	**0.49**	**0.52**	**0.53**	0.42

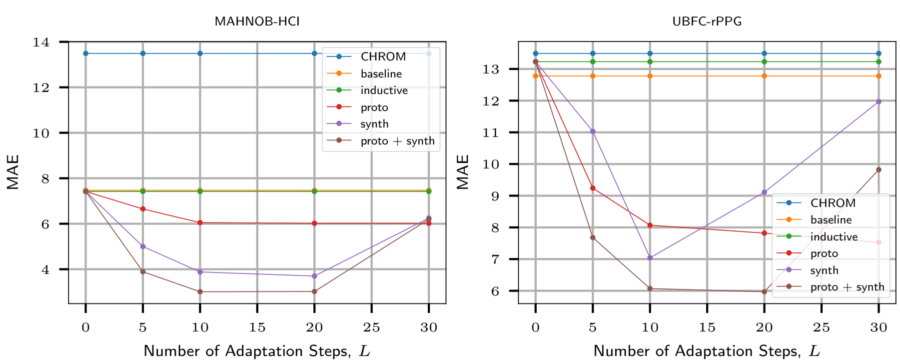

Fig. 1: Mean absolute error, MAE, obtained using different rPPG estimation methods. Demonstrates how the number of adaptation steps, L, affects performance.

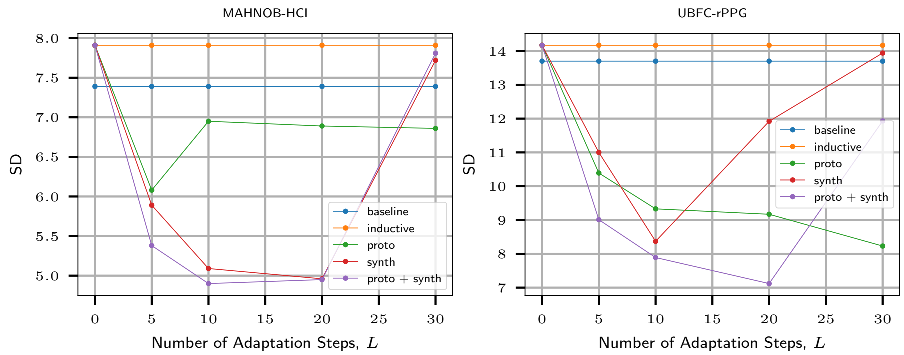

Fig. 2: Standard deviation of error, SD, obtained using different rPPG estimation methods. Demonstrates how the number of adaptation steps, L, affects performance.

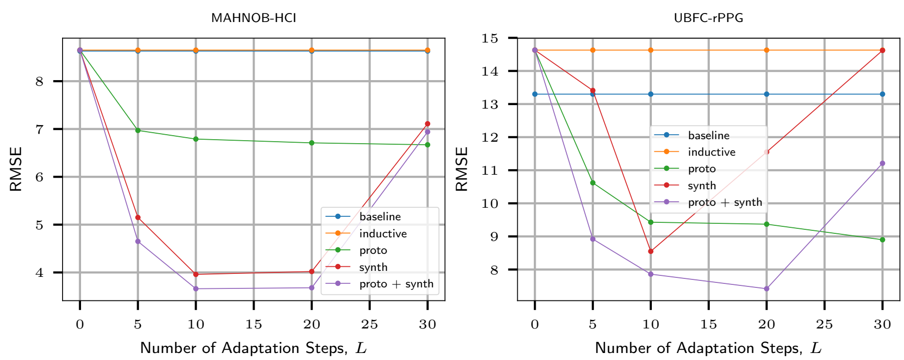

Fig. 3: Root mean squared error, RMSE, obtained using different rPPG estimation methods. Demonstrates how the number of adaptation steps, L, affects performance.

Table 9: Results of average HR measurement on MAHNOB-HCI comparing the difference between joint adaptation of *both feature extractor and rPPG estimator* and updating the *feature extractor only*.

Method	HR (bpm)		
	MAE	RMSE	R
Joint Adaptation	4.25	6.09	0.81
Extractor Only	**3.01**	**3.68**	**0.85**

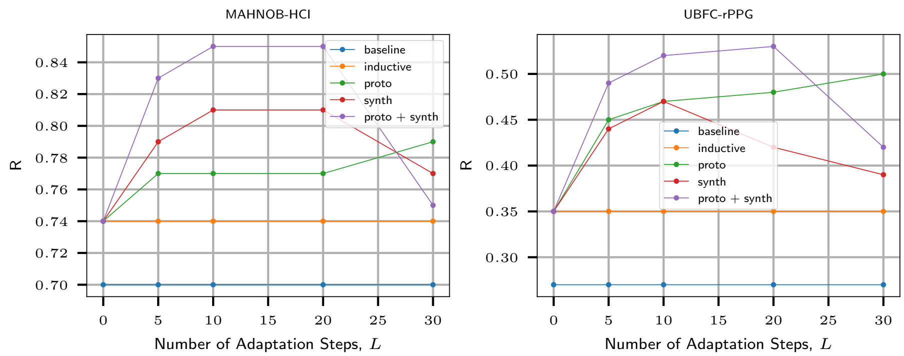

Fig. 4: Pearson correlation coefficient, R, obtained using different rPPG estimation methods. Demonstrates how the number of adaptation steps, L, affects performance.

More subjects are also shown here to give a better understanding of the importance of transductive inference for rPPG estimation using a deep learning model.

5 Demonstration Using Video

We show the implementation of our algorithm on videos extracted from MAHNOB-HCI to show its performance during deployment. We demonstrate our algorithm on videos of 3 subjects and a snapshot of a single frame from one of the video is shown in Fig. 5. From the video attached in the supplementary materials, it can be observed that the feature activation maps corresponding to our transductive inference method is relatively consistent when compared to a model trained in an end-to-end fashion. Please refer to our video for a better understanding on the improvements brought by the introduction of transductive inference to rPPG estimation.

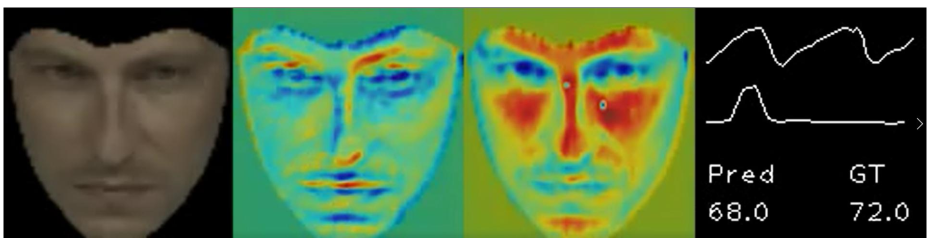

Fig. 5: A frame extracted from a video from MAHNOB-HCI. From left to right: 1. Pre-processed face image (zero-ing of pixels outside facial landmarks), 2. feature activation maps corresponding to end-to-end trained model, 3. feature activation map corresponding to Meta-rPPG (proto+synth) and 4. plots containing rPPG signal (top), power spectral density of rPPG signal (middle), predicted (bottom-left) and ground truth heart rate (bottom-right) in beats-per-minute (BPM).

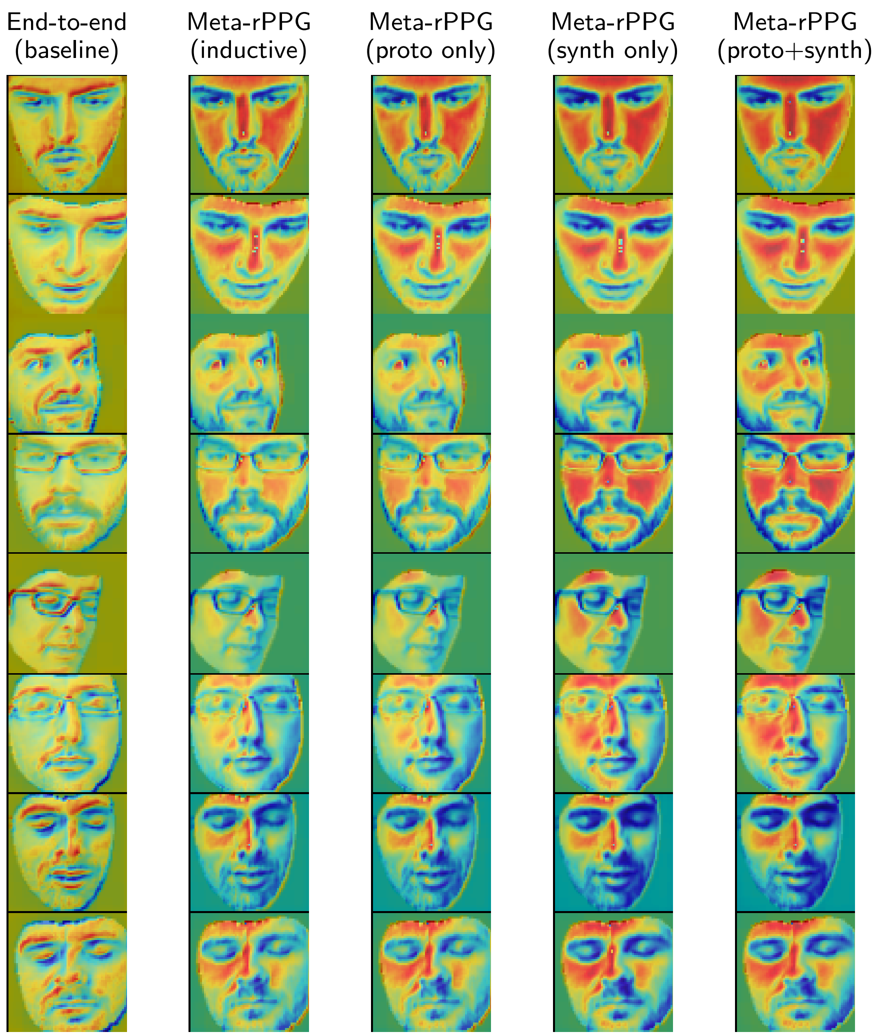

Fig. 6: Feature activation map visualization of 8 subjects (each row corresponds to one subject) using different training methods. Ablation study is performed by inspecting the feature map activation the results upon the application of every proposed transductive inference method. Usage of transductive inference results in activations of higher contrast and covers larger region of facial features that contributes to rPPG estimation.