Sanjit Sarda
Krish Shah

EE189: Final Report

Introduction

Remote photoplethysmography (rPPG) is a non-invasive technique that uses a video camera to virtually measure a person's vital indicators such as heart rate, respiration rate, and blood oxygen saturation. The rPPG technique is based on the fact that hemoglobin absorbs light differently depending on whether it is oxygenated or deoxygenated in the blood, which can be observed by a camera. This footage can be processed using algorithms that extract the coloring variations produced by the pulsing blood flow, resulting in a signal that represents the person's vital signs. To train these algorithms and test their accuracy, a person's vital signs are measured using a clinical device such as a pulse oximeter.

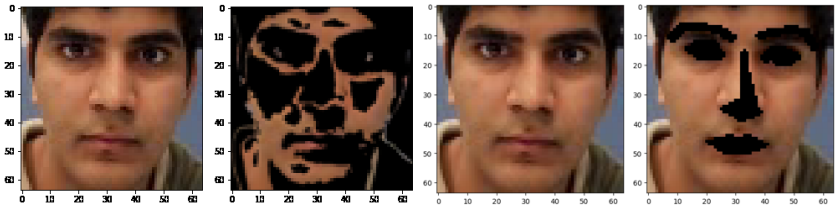
This report will highlight some of the techniques that we have used to improve the quality of our rPPG signal and achieve a mean absolute error (MAE) close to clinical standards. We will also discuss the use of 2D and 3D convolutional neural networks (CNNs) as well as other signal and image processing techniques, such as skin segmentation and linear color reduction.

Model Description

Preprocessing

Skin Segmentation

Skin segmentation allows the model to focus on the skin pixels without being distracted by non skin pixels. We made two attempts at skin segmentation. First, SemanticSegmentation by WillBrennan and second, LBFmodel.



Since the first method was too long we could never implement it with a neural network model, however the second model adversely affected the MAE of the model. Pictured to the left are the two models: SemanticSegmentation and LBFmodel.

Linear Color Reduction

Since the change in blood volume changes the RGB colorspace along a certain vector we can reduce color channels from 3 to 1 by taking the dot product with respect to that vector. To visualize the changes in the color space, here are two plots showing why this observation is useful. The first(linked) one shows a 3D scatter of the average RGB value across all pixels of every frame, and the second(linked) is a video plot showing how the traversal of this average value across time.

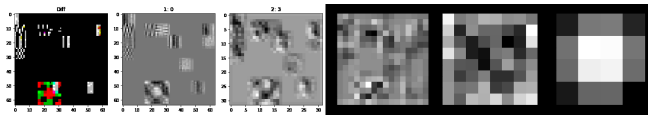
Dataset

Our final model was trained on videos with 1600 frames and a frame length of 512 and 20 samples per second.

CNN Models

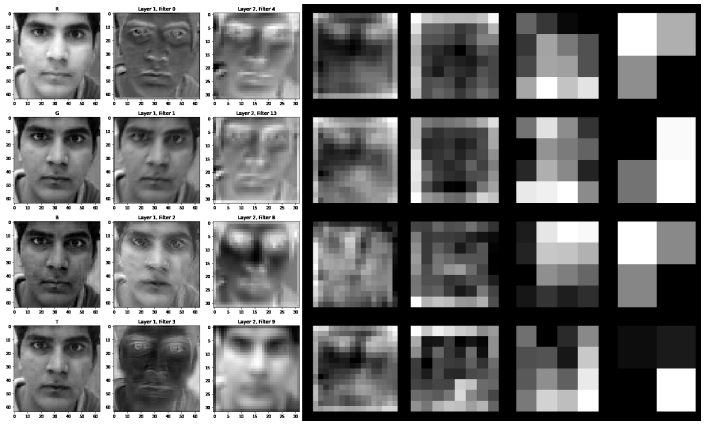
2D Convolutional Neural Network

Our 2D Convolutional Neural Network follows a standard Computer Vision architecture. It has 5 Convolution layers followed by a Dense layer. Each convolution layer consists of a 2D Convolution, a TanH activation and a 2D MaxPool. The model takes in the derivative of the PPG signal as an input, to keep the signal smooth. Each convolution layer extracts features from the image, and after a number of epochs, the model picks up on features that extract the heart rate signal. With this approach we were able to achieve an MAE of 5.6 on the test set. To better explain and visualize the model, here are some selected filters from each convolution layer. The first image is the time differential of one of the frames. As it progresses through the model, the layers apply different filters and reduce the resolution.While applying the linear color reduction to the model, we added a learnable vector to the model and found that reducing the color channels improved the speed of the model and decreased the MAE of the model by 8 times.

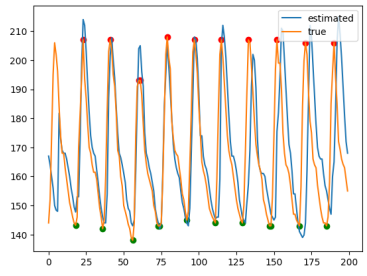


3D Convolutional Neural Network

Our final model is a 3D Convolutional Neural Network with 6 convolutional layers, each having a 3D CNN followed by a batch normalization, a ReLU activation and a MaxPool. Unlike the 2D convolution layer, the 3D CNN keeps a consistent 32 channels throughout the convolutions, which gives better results. The ReLU function was found to improve computational efficiency and counter overfitting by dropping out layers. This time, we found that the linear color reduction decreased our performance. However, adding this to the existing 3 colors at the beginning of the model improved the MAE of our model on the test set from 2.65 to 2.18. Here is a visual of 4 filters from each convolution layer. In the first layer we see the RGB filtered image along with the image with the linear color reduction. Similar to the 2D CNN model, as the layers progress, the model applies different filters and the maxpool reduces the resolution.



Optimizer and Loss Function

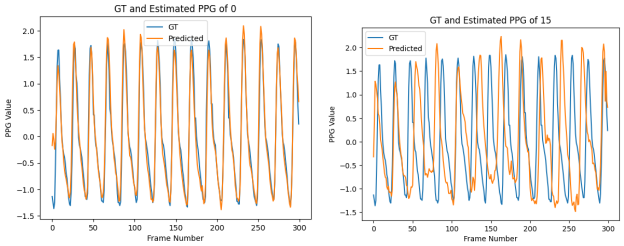


Our final model used an Adam optimizer with 1E-3 learning rate and 1E-4 weight decay. We tried a number of different loss functions including Neg Pearson, MSE and a custom Peak comparing loss function. We hoped that our color reduced channel magnified the vectors where color change was significant, and that the model would learn to look along those changes, especially in peaks and troughs of the ground truth. To guide the model with this, we developed a loss function to compare the values of the ground truth and estimated ppg signal where the ground truth had peaks and troughs. In this plot, the dots represent the peaks and troughs, where the ground truth and estimated signal are compared. While implementing this loss function, we tried combinations of MAE, Peak and Trough and Neg Pearson. Ultimately, the Neg Pearson guided our model well without overfitting the data, so we decided to try permutations

of each loss function. Ultimately, Neg Pearson outperformed all the other loss functions, since it is efficient at handling multi-channel data with a broad variety of intensities. Neg Pearson loss is capable of handling such data because it normalizes it to have a zero mean and unit variance, reducing the influence of intensity differences between channels. Furthermore, the Neg Pearson loss is affected by the linear connection between the predicted and ground truth signals, which is the case for our rPPG model.

Results

	Train Set	Test Set	Out of Distribution Set
MAE	1.76	2.18	8.8



GT and Estimated PPG of Videos 0 and 15.

Limitations and Future Work

Even though our model performs quite well on our test and train set, there are some limitations in the models abilities to predict PPG signals. First, there is a significant difference between the MAE on the test set and the MAE on the train set which shows that our model overfits the train data. This is mainly because we did not test our results on a validation set after every epoch to check overfitting. Furthermore, our model was limited to a small window looking at only the face of the person, and without a background, it was difficult to evaluate the PPG signal based on changing lighting conditions.

We tested many techniques not present in our final model that have potential. One such technique would be skin segmentation: If we could find a way to improve the computational efficiency of skin segmentation and perform it on those videos, we believe that would significantly improve the performance of the model. We also wanted to experiment with a more guided model architecture by adding additional filtered channels to the input and creating a more guided loss function that worked with these filters. Additionally we wanted to experiment with sectioning off parts of the face and taking the dot product of vectors based off of the region of the face the pixel fell on. We believe this may improve results since different parts of the face are affected by the blood volume in different intensities. Furthermore we would like to look into changing lighting conditions and experiment with filters that understand those lighting changes based on background information.