Supervised learning for heartbeat detection in photoplethysmography

Talha Agcayazi
Tanner Songkakul

Introduction and Motivation

Photoplethysmography (PPG) is an optical biosensing technique which uses interactions of photons with oxygenated and deoxygenated hemoglobin to determine blood volume in a vessel. This volume is related to heart rate because blood volume varies over time with the pumping of blood by the heart. This technique is used to measure heart rate in almost all modern wearable health trackers. However, commercial fitness trackers demonstrate inaccuracy in their measurement of heart rate [1]. In this project, we used a convolutional neural network (CNN) to identify heartbeat in PPG segments and compared its accuracy in identifying heart beats with other traditional clustering and classification techniques.

Overview of Supervised Learning Approach

Data from the TROIKA dataset was used as the source data for this project [2]. The TROIKA dataset contains 2 channels of wrist PPG data, electrocardiogram data (ECG), and accelerometer data (Figure 1). This data was collected from patients running on a treadmill over the course of a total of 5 minutes, varying in speeds of 1–2 km/h, 6–8 km/h, and 12–15 km/h at set intervals. The sampling rate was 125 Hz. During data collection, patients were encouraged to pull clothes, wipe sweat, push buttons, and otherwise mimic a real-world exercise situation.

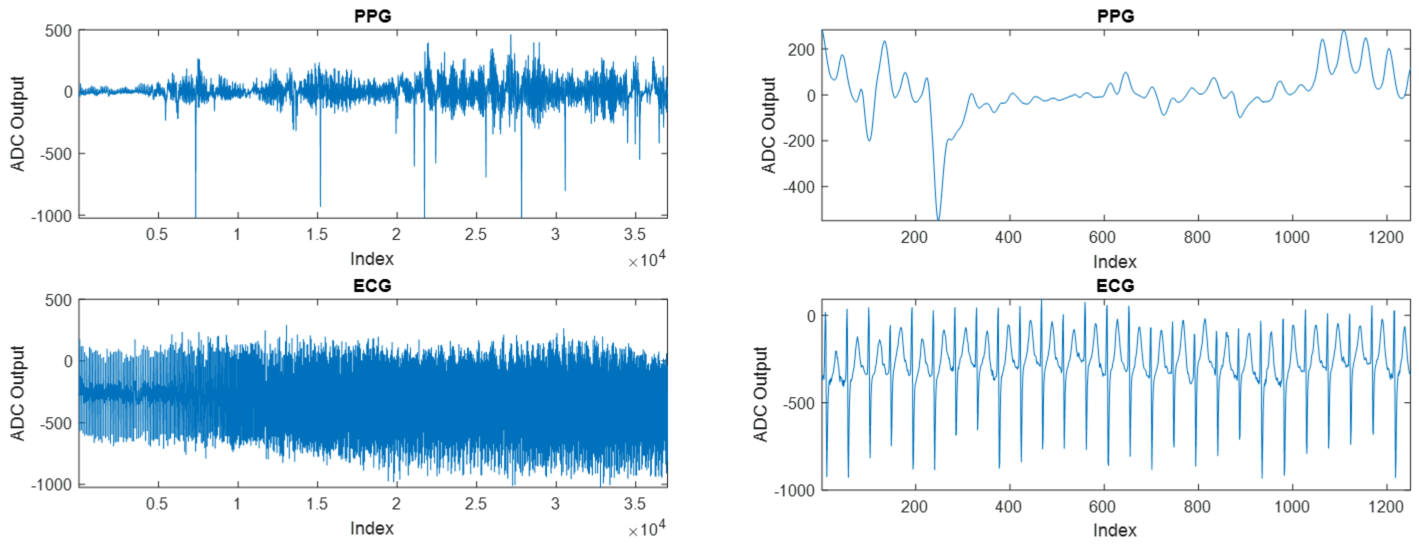


Figure 1: (Left) ECG and PPG signals for a single patient over the course of the study. (Right) 10 second window of the PPG and ECG signals

After PPG and ECG data from TROIKA is imported into Matlab, it is preprocessed to reduce noise and normalize the signal. The ground truth heartbeat data is identified from the ECG data and used to extract segments of the PPG data containing heartbeat (HB) vs no heartbeat (NHB). Features are extracted from the PPG segments, and various classifiers are applied to distinguish between heartbeat from no heartbeat segments.

Preprocessing PPG Data

Preprocessing was carried out as defined by Grisan et al [3]. The preprocessing stage is necessary to reduce dynamic range of the PPG signal from motion, remove any trends, and reduce noise. Mean and standard deviation is found for windows of length F_sN for N=8. These means and standard deviations are used to normalize the PPG signal (Equation 1).

$$PPG_{norm}(t) = \frac{PPG_{raw}(t) - \sigma_{PPG}(t)}{\mu_{PPG}(t)}$$

Equation 1: Calculation of normalized PPG from mean and standard deviation.

The normalized PPG signal trend is found by a moving average filter of length F_sN for N=2. This trend is then subtracted from the normalized PPG signal, resulting in a detrended signal. The final step is a 7th order Butterworth filter with Fc=18 Hz. The output of this filter is the preprocessed data, which has been normalized, filtered, and detrended (Figure 2).

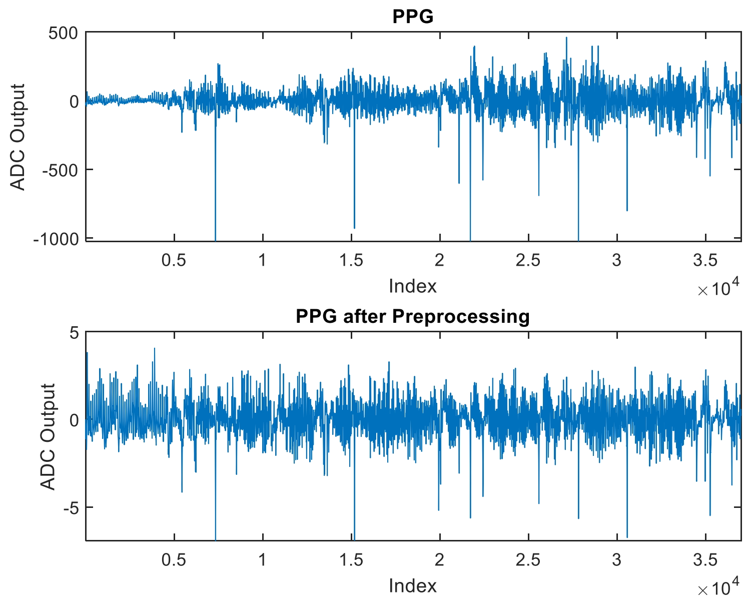


Figure 2: Sample PPG data before and after preprocessing.

Ground Truth Extraction

Grisan et. al approach was also followed for ground truth extraction [3]. Peak locations in the ECG signal were identified using the Pan-Tompkins Algorithm implementation built into Matlab (Figure 3).

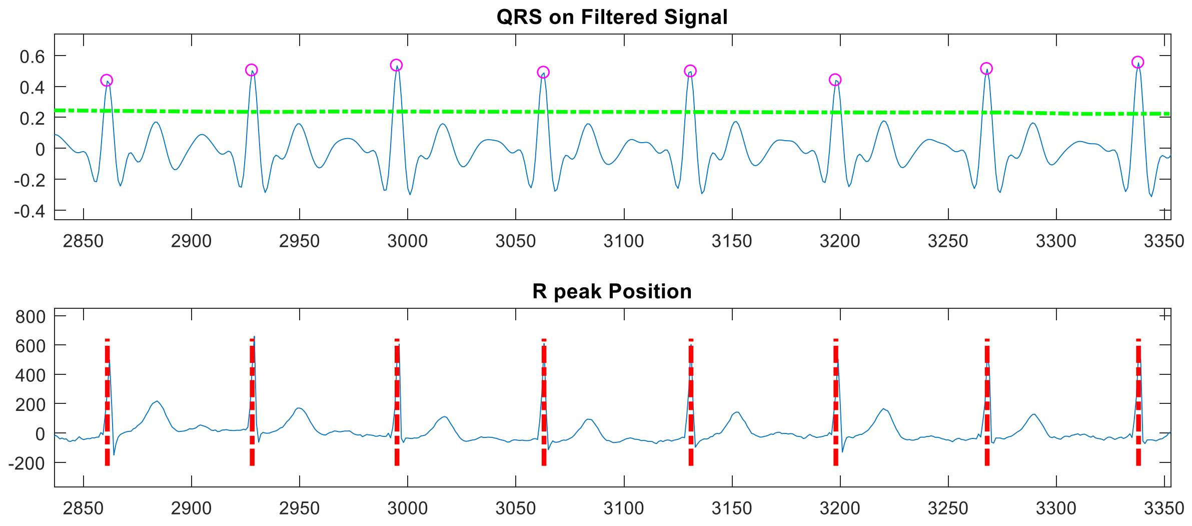


Figure 3: Output of Pan Tompkins

These peak locations were used to find the heartbeat (HB) segments contained in the PPG signal starting at R_i and spanning $\Delta R_i = R_{i+1} - R_i$. These same locations were used to find the no heartbeat (NHB) segments by offsetting by half of the interval, therefore centering this the NHB PPG signal midway between beats using $R_i + 0.5\Delta R_i$. The HB and NHB segments of the PPG signal were then resized using stretching and interpolation to the maximum segment size of length 140. We extracted around 32k segments from the PPG data.

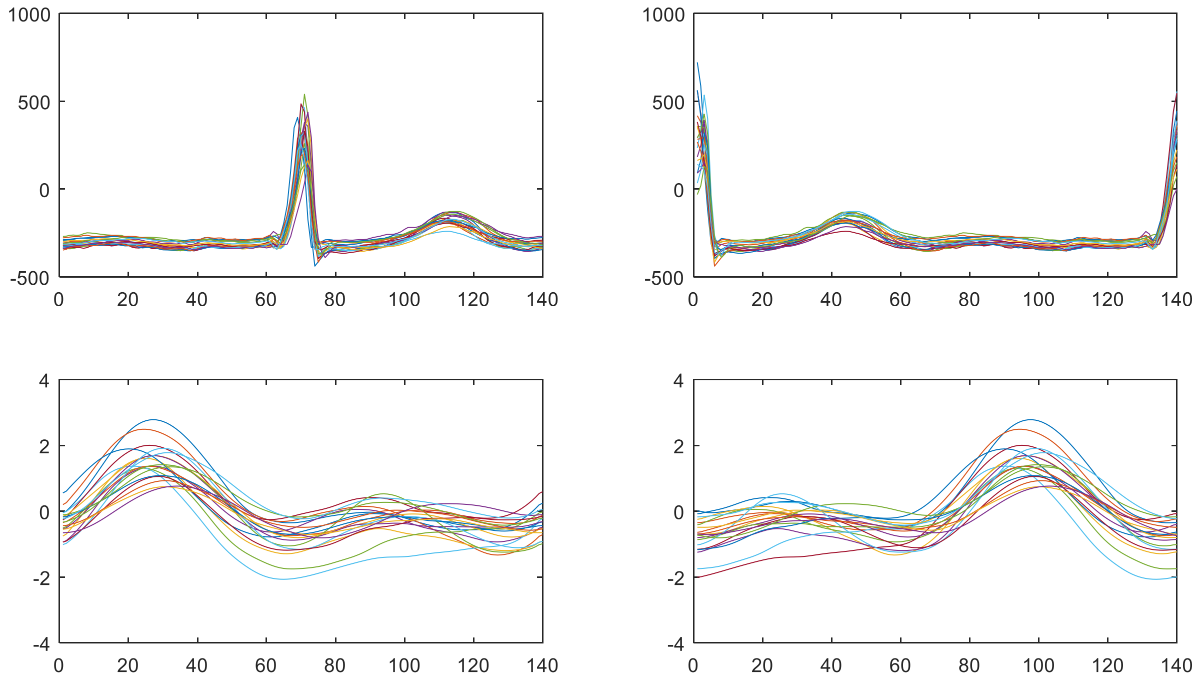


Figure 4: (Left) Segmented and stretched HB signals. (Right) Segmented and stretched NHB signals. Segmented ECG signals are given on top for reference.

Feature Extraction

The features in Table 1 were extracted from each segment of PPG data and labeled with HB or NHB. The 3-level quantized version is derived in Equation 2. Principle component analysis was carried out to find 12 PCA eigenvectors. Segment energy was calculated by finding the frequency domain energy spectrum from the FFT.

Feature Description	Features
Sample	140
Sample Mean	1
Sample Standard Deviation	1
3 level quantized version of sample	140
Principle Component Analysis	12
Maximum Intensity Position	1
Minimum Intensity Position	1
Time difference between maximum and minimum position	1
Segment energy in [0.04, 0.09], [0.09,0.15], [0.15,0.60] Hz bands	3

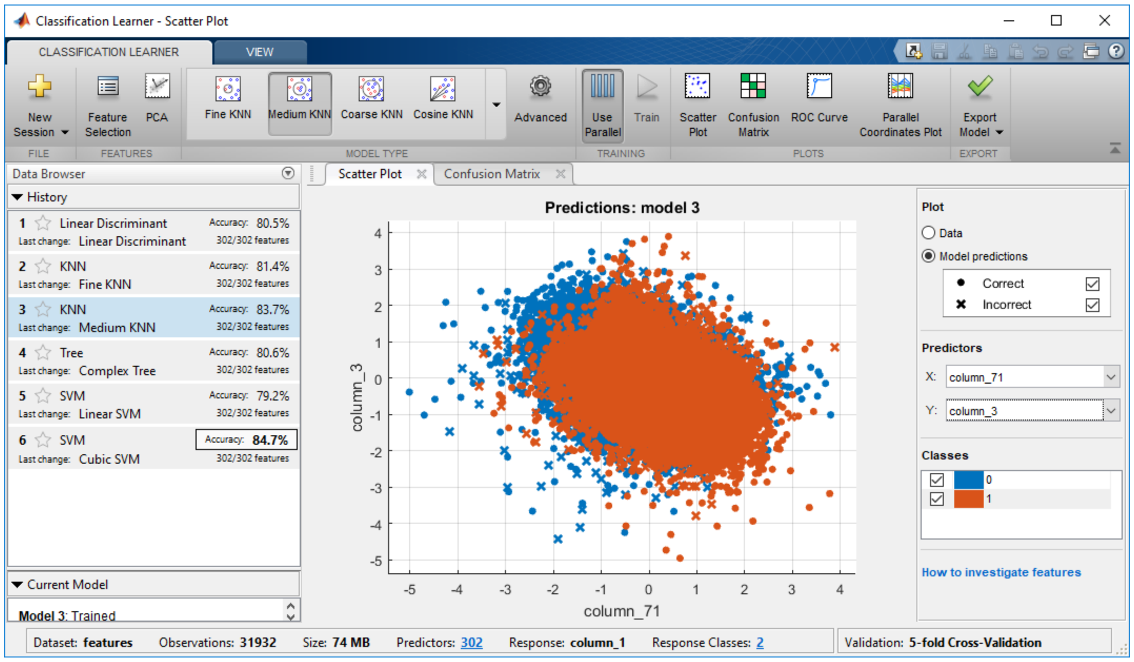
Table 1: Features extracted from normalized PPG data.

$$PPG_{quant}(t) = \begin{cases} -1, & PPG(t) < \mu - 0.5\sigma \\ 0, & \mu - 0.5\sigma \leq PPG(t) \leq \mu + 0.5\sigma \\ 1, & PPG(t) > \mu + 0.5\sigma \end{cases}$$

Equation 2: Quantized version of PPG Signal

Machine Learning Techniques

To classify between HB and NHB segments we decided to first try traditional classification techniques as in Grisan et al's work [3]. We tested linear discriminant, k-nearest neighbor (1 and 5-nn), decision tree and support vector machine (linear and cubic) classifiers using Matlab's Classification Learner Applet. For these tests, we used all 302 of the features with 5-fold validation. A screen shot of the Classification Learner Applet with our final classifiers selected is shown in Figure 5.



History

	Model	Accuracy	Features
1	Linear Discriminant (Last change: Linear Discriminant)	80.5%	302/302 features
2	KNN (Last change: Fine KNN)	81.4%	302/302 features
3	KNN (Last change: Medium KNN)	83.7%	302/302 features
4	Tree (Last change: Complex Tree)	80.6%	302/302 features
5	SVM (Last change: Linear SVM)	79.2%	302/302 features
6	SVM (Last change: Cubic SVM)	84.7%	302/302 features

Current Model: Model 3: Trained

Plot: Data [✓ Model predictions]; ● Correct ✓; ✖ Incorrect ✓

Predictors: X: column_71; Y: column_3

Classes: ✓ 🟦 0; ✓ 🟧 1

Dataset: **features** Observations: **31932** Size: **74 MB** Predictors: 302 Response: **column_1** Response Classes: 2 Validation: **5-fold Cross-Validation**

Figure 5: A Screenshot of the Matlab's Classification Learner App with the final list of classifiers.

Convolutional Neural Network Implementation

Convolutional Neural Networks (CNN) have become increasingly popular because of their ability to learn important features during training without the need of feature engineering. Our CNN structure, shown in Figure 6, takes the segment samples as inputs without any of the generated features. The network contains many layers of convolution (3x1 kernel), pooling and flattening. We implemented this network by using the popular TensorFlow and Keras libraries written for Python [4].

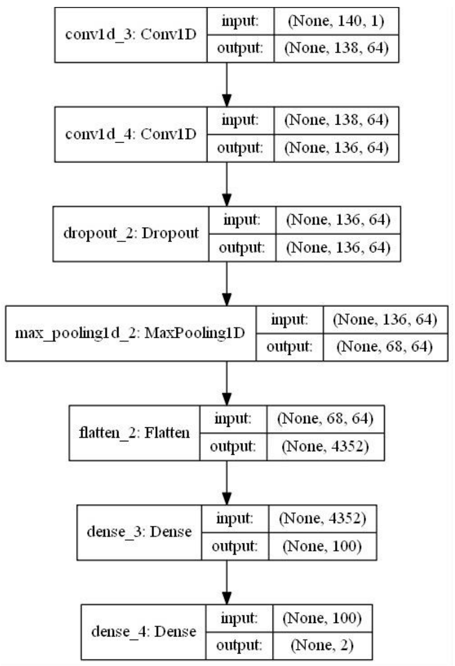


Figure 6: Structure of our CNN implementation

Results and Discussion

Aside from prediction accuracy, we also measured the training and testing time to evaluate our classifiers. The results of our experiments are given in Table 2. Among the traditional classification techniques, we observed the best accuracy from the Cubic SVM classifier. The CNN classifier performed better than the Cubic SVM classifier without any of the computed features. In terms of training time, the linear discriminant classifier performed the best with a training time of 50s. The CNN algorithm went through 10 epoch cycles of training in 6 minutes which is still fast for the accuracy it achieved. Prediction timings are measured by predicting the full dataset of about 32k segments. Among all of the classifiers, the decision tree classifier was the fastest predictor followed by linear discriminant and CNN classifiers.

Classifier	Accuracy	Training Time (mm:ss)	Prediction Time (mm:ss)
Linear SVM	79.2%	36:00	01:18
Decision Tree	80.5%	02:00	00:01
Linear Discriminant	80.6%	00:50	00:01
1-NN	81.1%	36:50	02:12
10-NN	83.5%	36:40	02:12
Cubic SVM	84.5%	59:00	00:58
CNN	86.6%	06:00	00:12

Table 2: Comparison of classifiers in terms of accuracy, training time, and prediction time

Conclusions

In this project, we carried out preprocessing, feature extraction and classification using the PPG and ECG signals from the TROIKA dataset [2]. Raw PPG data was normalized, filtered, detrended, and mapped to peaks detected in the ECG signal. A total of 302 features were extracted, and used as inputs to various classifiers. Among the classifiers that used features, the cubic SVM provided the best results. The CNN performed better than the cubic SVM without the computed features. This convolutional neural network achieved a peak performance of 86% correct heartbeat identification.

Future Work

There are a few possible ways to extend our work in this project. To increase the classification accuracy, we can train separate classifiers based on the activity levels of the user. This would allow us to have a cascade of classifiers each one optimized for different activities as indicated by the accelerometer data in the dataset. The accelerometer data could also be used in adaptive filter to vary the filtering parameters based on the amount of motion. We could also try adding new features such as wavelet and coefficient of skewness. To increase the accuracy of the CNN, we can try different hyperparameters. After the classification has been refined, there is also the matter of using these peaks to calculate the patient's actual heart rate. There are also integration challenges that we can work to solve in order to use the classifiers on real-time wearable systems such as power and memory limitations.

Sources

[1] S. Benedetto, C. Caldato, E. Bazzan, D. C. Greenwood, V. Pensabene, and P. Actis, "Assessment of the Fitbit Charge 2 for monitoring heart rate," PLoS One, vol. 13, no. 2, pp. 1–11, 2018.

[2] S. Zhu, K. Tan, X. Zhang, Z. Liu, and B. Liu, "TROIKA: A mixed approach for heart rate monitoring during intensive physical exercise using wrist-type PPG Signals," Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS, vol. 2015–November, no. 2, pp. 2347– 2350, 2015.

[3] E. Grisan, G. Cantisani, G. Tarroni, S. K. Yoon and M. Rossi, "A supervised learning approach for the robust detection of heart beat in plethysmographic data"; 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Milan, 2015, pp. 5825-5828.

[4] https://machinelearningmastery.com/cnn-models-for-human-activity-recognition-time-series-classification/