ECG application

Github: https://github.com/huangjuite/ecg-id

Members

Student ID	Name	Work assignment
0510746	黃嘉伶	ML, LSTM
0510725	莊昱虹	MI diagnosis
0860078	黃瑞得	AE, classify

Introduction

Biometric technology has been developed for a very long time. Fingerprint identification and Iris recognition have already been proved to be very robust. But there is not much work using ECG to identify people. And previous work shows that it is difficult to identify people after exercise. So we decided to take this challenge and use different learning approaches to improve the performance of ECG identification, or even other applications.

The project aims to conduct the ECG signal study with wearable devices. The system

contains enrollment and recognition phases, which respectively means collecting users' ECG signals as the dataset of the classification model and identifying users by features extracted. To maximize the performance of the model, the key point is to simultaneously satisfy the inter-individual variability(originated from the human body) maximization and intra-individual variability(one user's HRV in different health conditions) minimization.

Related Work

Wang et al.[1] have applied existing methods to analyze the performance of rest-ex ECGID with the ECG database[3]. They have tried time-domain features like QRS complex, and transform analysis like short-time fourier transform or wavelet transform. They also experimented with machine learning or deep learning models like SVM or LSTM. However, they found that for those methods, the rest-rest ECGID accuracy can reach more than 95% while the ex-ex ECGID performance becomes around 85%; As for the rest-ex ECGID performance, most methods collapse to 10% or even worse.

Nurmaini et al., 2019[2] extract fixed length heartbeat sequences and use DNN and CNN models to build the self supervised autoencoder and identify various heart diseases. Their model achieved an accuracy of 99.73%. Outperformed other DNN, CNN based methods without autoencoder. Also outperformed traditional machine learning methods.

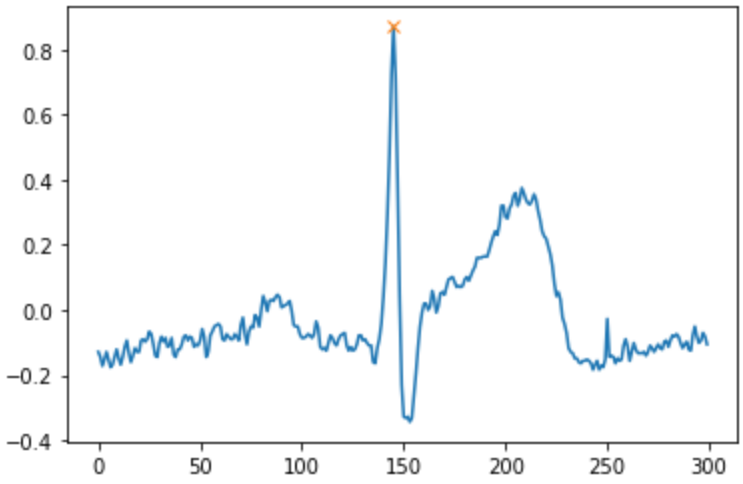
Liu et al, 2014[4] they extract the coefficients of ECG curve from each cycle and reach 94% in diagnosis of Myocardial Infarction. With the novel method of feature extraction, polynomial fitting, they avoid the risk of missing ST-elevation and steady the performance.

ECG identification

Dataset and preprocessing

ECGID-database[3]

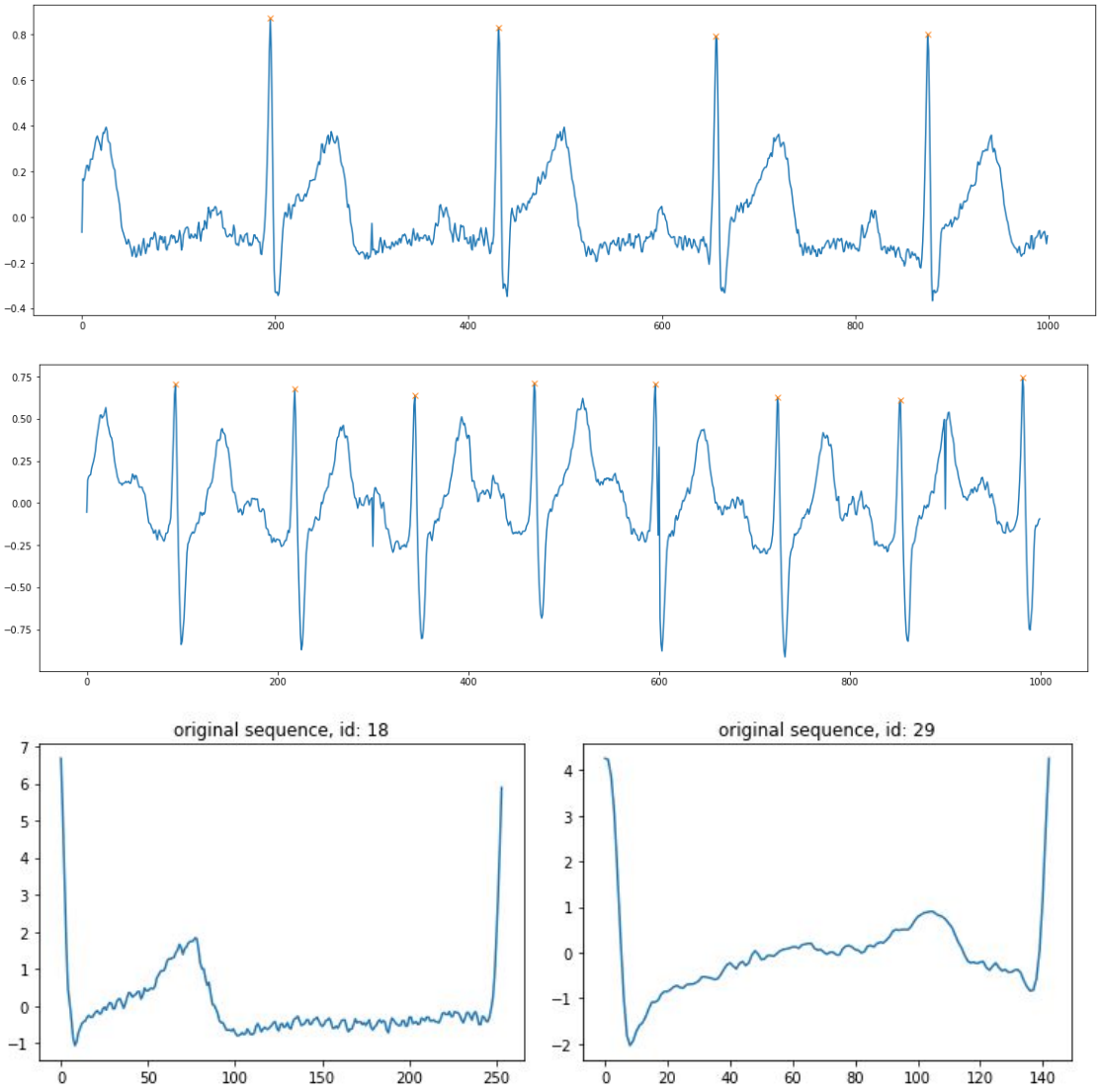
The database is collected in a lab at the South China University of Technology. The database includes pre and post-exercise recordings for 45 subjects. There are 33 males and 12 females whose age is between 18 to 22. Each subject in this set performed a few basic structural workouts such as steady running and climbing the stairs. The ECG signals are recorded from the wrist and the length of recordings in rest condition and post-exercise conditions are around 5 minutes and 150 seconds respectively. The heart rate in the post-exercise condition is the range from 90 to 150 compared with around 70 in rest condition. The ECG signal was captured in lead II and the sampling frequency is 300 Hz.



Data preprocessing

To identify each heartbeat. First, we use the scipy tool "signal.find_peaks" to find out peaks in recorded heartbeat signals. Then we cut out each heartbeat from peak to peak. Due to the record condition could be during exercise or rest. The length of each heartbeat is ranging from 80+ to 200+ data points. To filter out the corrupted heartbeat data. We remove the data that has a length shorter than 80 or longer than 300. We also standardized all the heartbeats data to have a standard range of all data and make a copy of data resampling to 128 data points for later reconstruction. We split each person's exercise and rest heartbeats to train and test datasets using a split ratio of 0.3/0.7(test/train). The total heartbeat number is in the table below.

Datasets	Number of heartbeats
train	**19032**
test	**8210**
rest	**16628**
exercise	**10614**



Resting heartbeat from 18th person and Exercise heartbeat from 29th person.

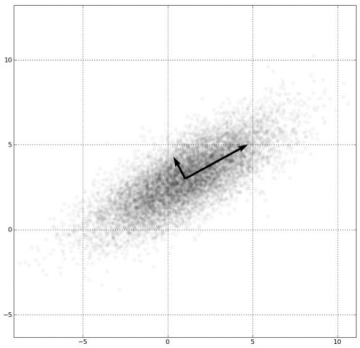
Machine learning

Model design

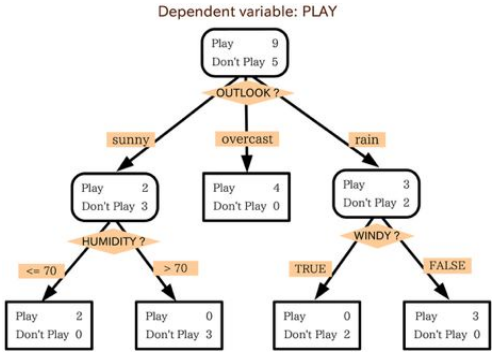
Principal Component Analysis(PCA) is unsupervised learning, which can be used to effectively reduce dimensionality. Therefore, we can show the data with better simplicity and cleanliness while losing little to no important information. Decision Tree is supervised learning which makes a decision intuitively and clearly. It is suitable for regression analysis and classification prediction. Shorter execution time is one of its advantages over other machine learning models.

- Feature extraction
    - PCA

- Classification
    - Decision tree



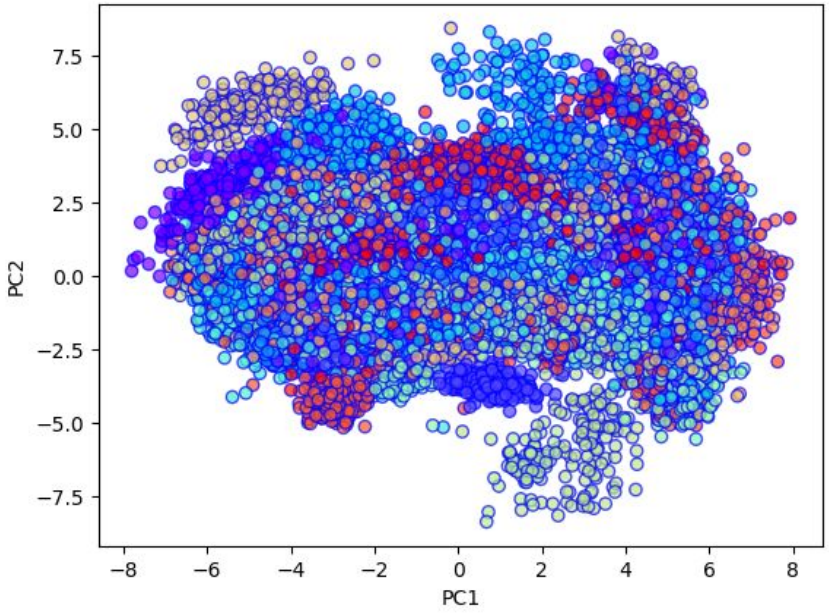
PCA



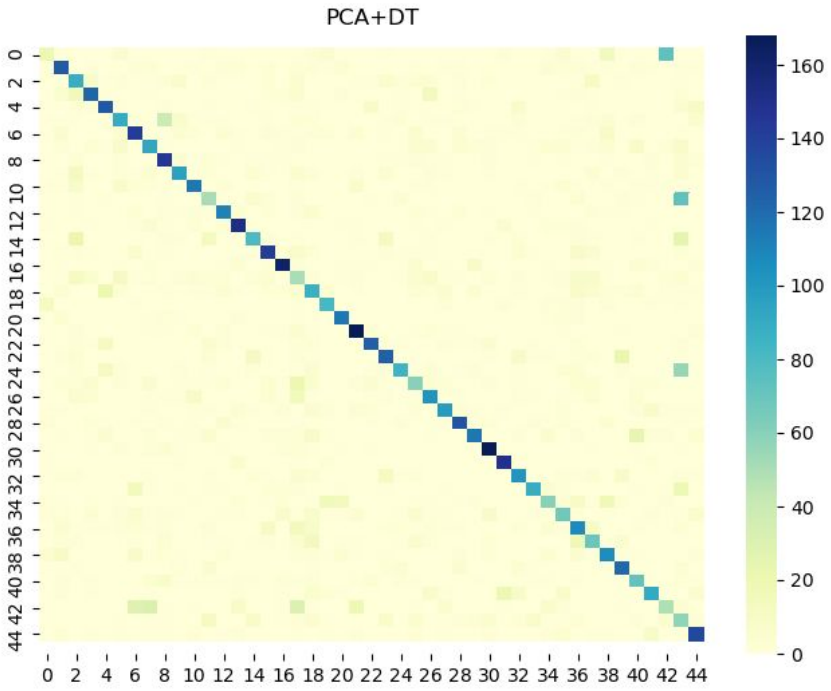
Decision Tree

Experiment result

After reducing 128 features to 44 features with PCA, we can see some obvious partitions from the distribution of two most significant features. Using the PCA and Decision Tree, we've achieved an accuracy of **0.6273** with the test dataset.



Distribution of two features



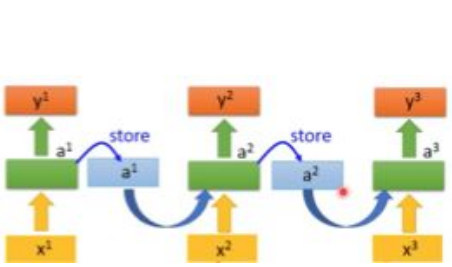
DNN model with LSTM cell

Model design

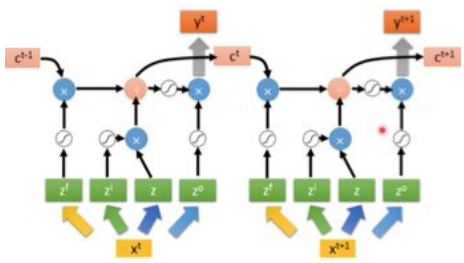
Since the ECG signal is a consecutive sequence, Recurrent Neural Network (RNN) is suitable for this dataset. However, RNN has a long-dependency problem because it keeps any information no matter what. Hence, we design the Long Short-Term Memory (LSTM) model with the hidden state containing 128 nodes. Then connect to a 32 dimension fully connected layer as the model output. LSTM has specific gates to decide how much information to remember or forget, which enables the neural network to have better performance.

LSTM

- hidden state 128
- Fully connected 32



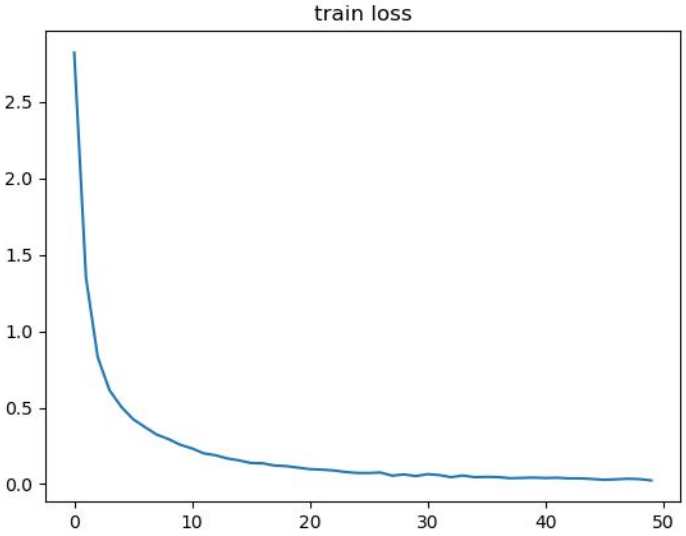
RNN structure



LSTM structure

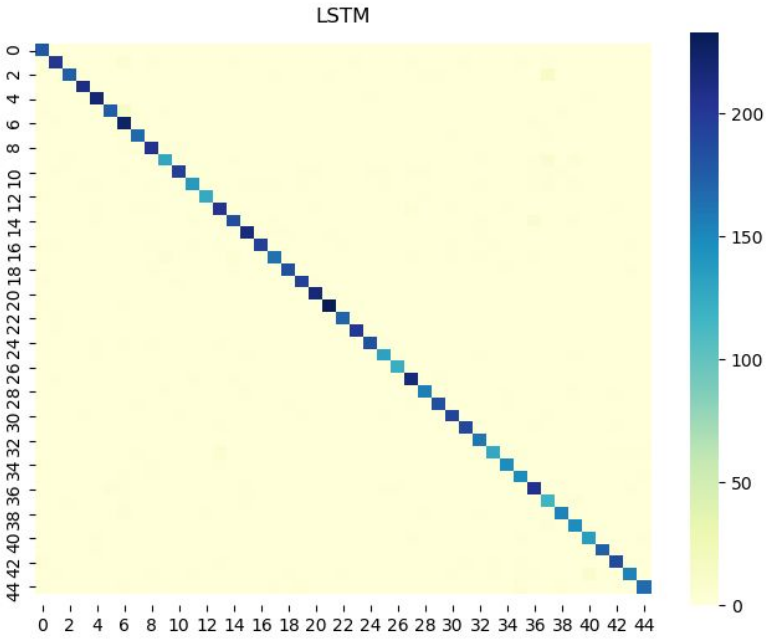
Train LSTM

We trained the LSTM with the training dataset for 50 epochs on an RTX2070 Nvidia GPU. The loss function is cross-entropy loss as it is a classification problem.



The learning curve of the LSTM

For sequence signals like ECG, networks with memory are more likely to lead to a better result. Using the LSTM, we've achieved an accuracy of **0.9621** with the test dataset. The confusion matrix shown below is nearly diagonal.



Autoencoder

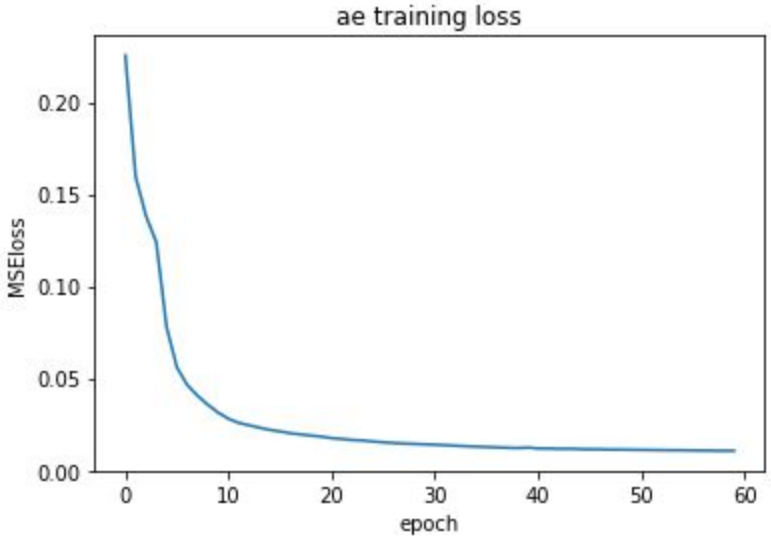
Model design

In order to deal with different lengths of the input heartbeats. We design the model encoder using the LSTM cell with the hidden state containing 128 nodes. Then connect to a 32 dimension fully connected layer as the encoder output. The decoder simply is 2 fully connected layers with 128 dimensions. For each heartbeat sequence input, the output is 128 data points. All layers use ReLu as activation function except the last decoder output layer.

- Encoder
    - LSTM (hidden state 128)
    - Fully connected 32
- Decoder
    - Fully connected 128
    - ReLu()
    - Fully connected 128

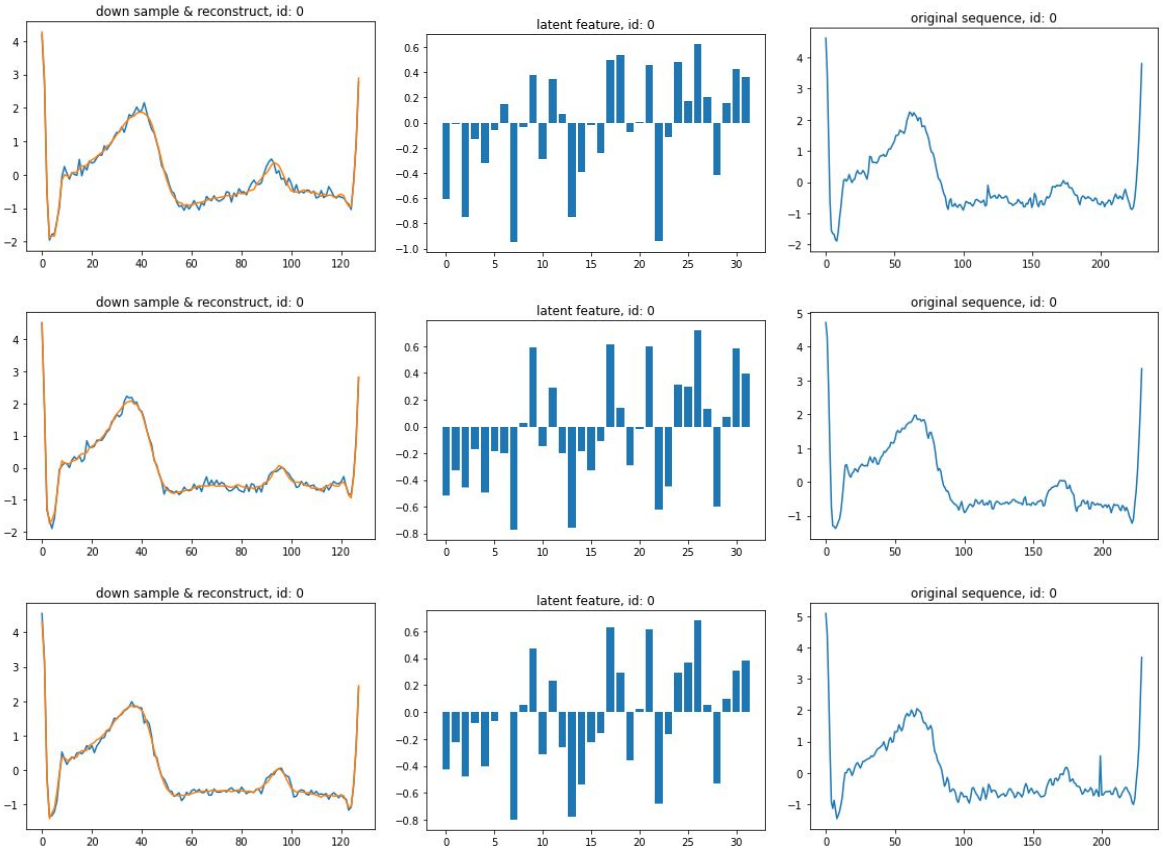
Train autoencoder

We trained the autoencoder with the training dataset for 60 epochs on an RTX2070 Nvidia GPU. To train the autoencoder to reconstruct to the resampled heartbeat. We use a mean square error as the loss function.

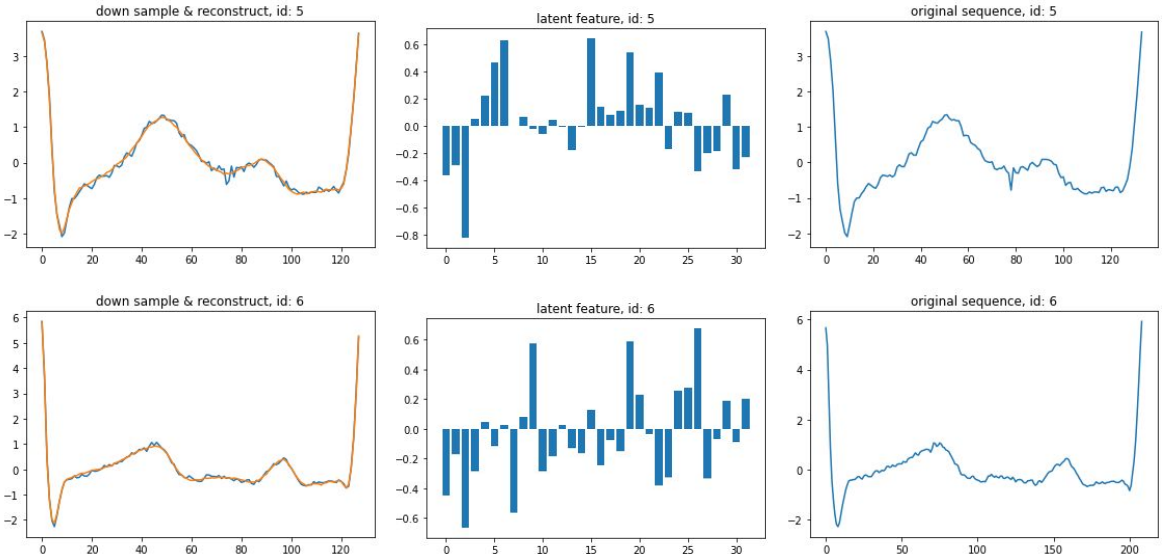


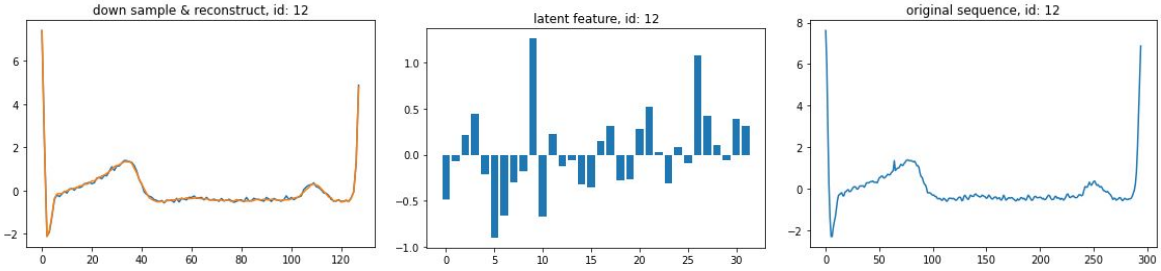
The learning curve of the Autoencoder.

samples of the **same** person's heartbeat reconstruction using an autoencoder.



samples of the **different** person's heartbeat reconstruction using autoencoder.

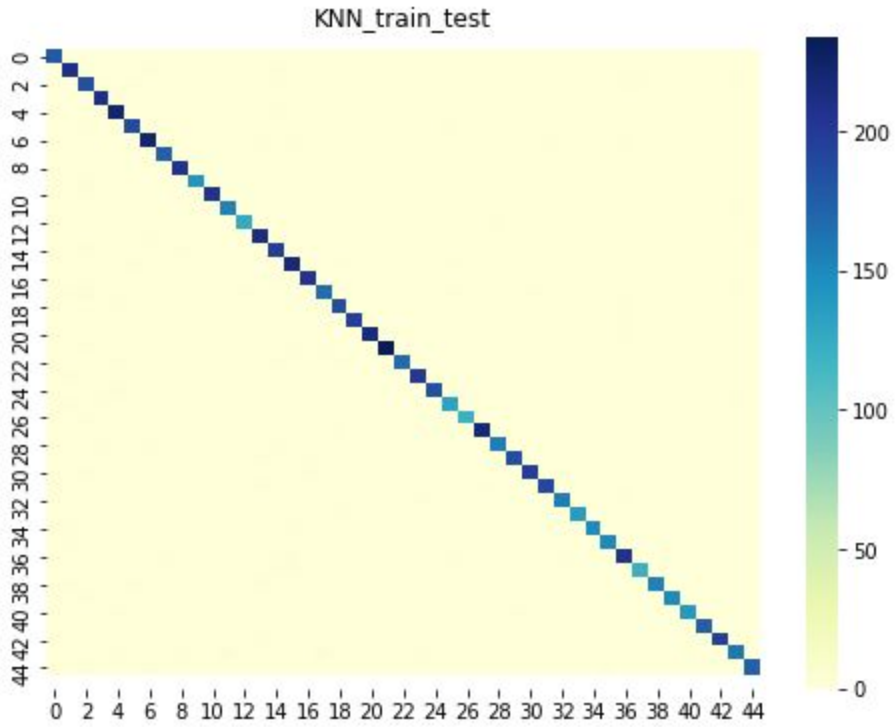




Classify heartbeat

- **KNN**

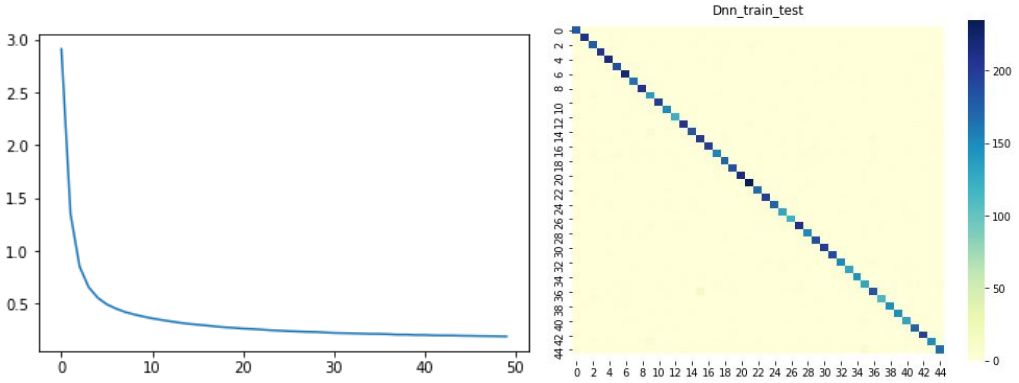
  We extract the latent feature from each heartbeat using the trained encoder. Then use a kth nearest neighbor with k=5 to identify each heartbeat. We've achieved an accuracy of **0.9854** with the test dataset.



Confusion matrix of the classification result

- **DNN**

  We also try to classify each heartbeat using the latent feature as input to a simple deep neural network. We design the neural network with 2 fully connected layers with 64 dimensions and an output layer of 45 classes. And we use a simple cross-entropy as the error function. The network was trained using latent features generated from train-dataset for 50 epochs. This method achieved an accuracy of **0.9487** which is slightly worse than the KNN.

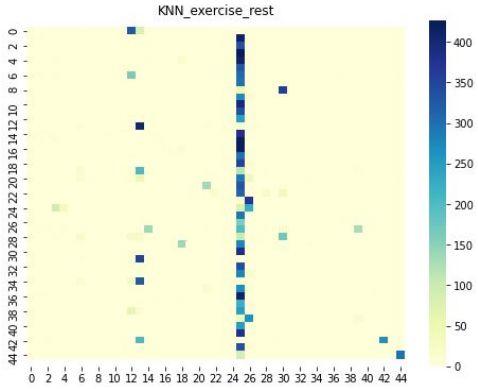


Learning curve and confusion matrix of the classification result using DNN

Data modality

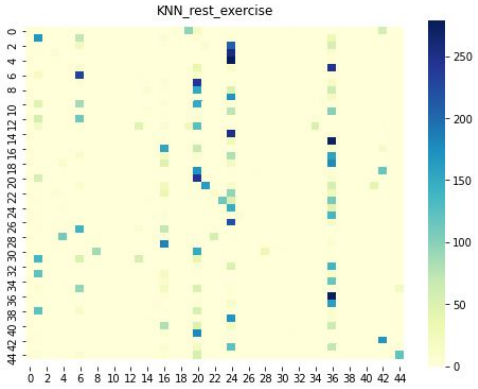
To study the data modality between rest heartbeat and exercise heartbeat. We try to use only one type of data to fit the KNN model and test the other one. And we found a similar result as the previous work shows. That these two types of heartbeat have a lot of different features. Despite the fact that they are recorded from the same person.

accuracy:0.1756



Fit exercise-dataset and test rest-dataset

accuracy:0.0868

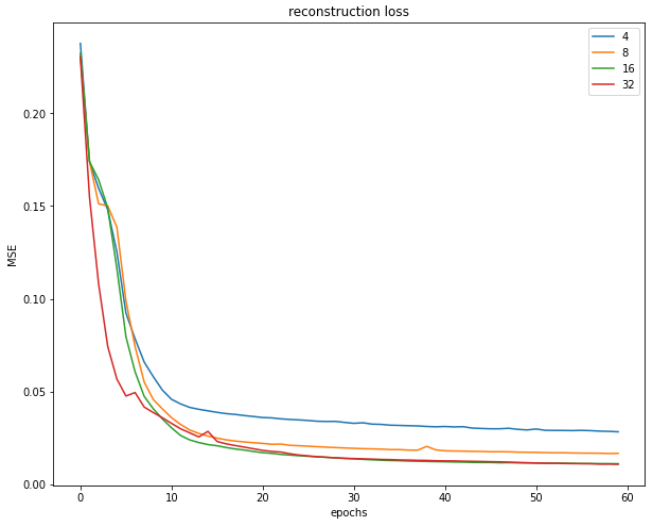


Fit rest-dataset and test exercise-dataset

Modify latent space

We also try to modify the dimension of latent space and try to find out how big the dimension needs to be to successfully reconstruct and classify the heartbeat sequence.

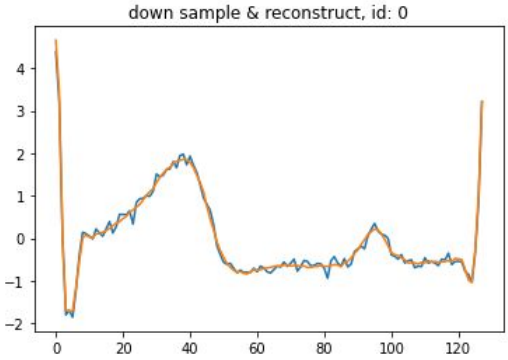
We start from narrowing down the latent space to the dimension of **32, 16, 8, 4**. We also try to adjust latent space with a dimension of 2. But the reconstruction error is too high to compare.

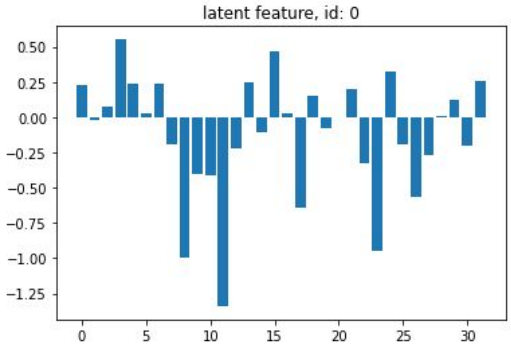


The reconstruction error increases when the latent space dimension decreases.

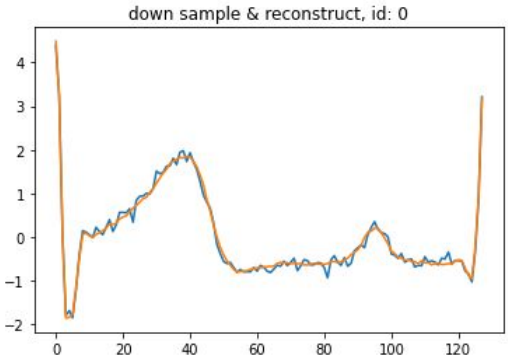
examples of the different latent space reconstruction of the same heartbeat.

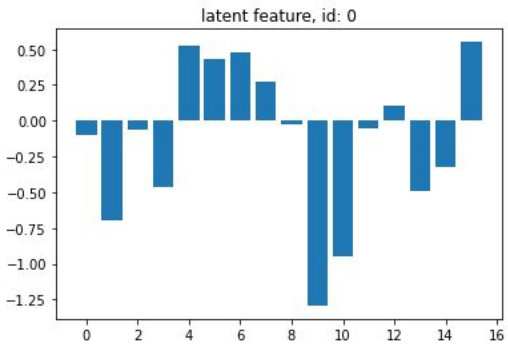
- **32**



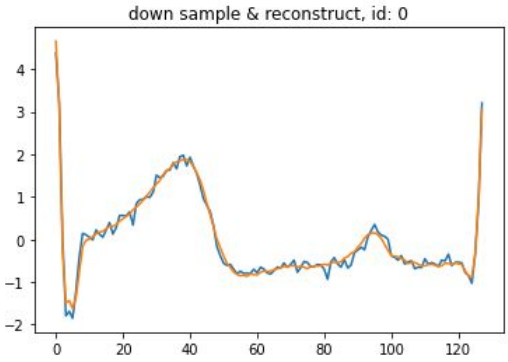


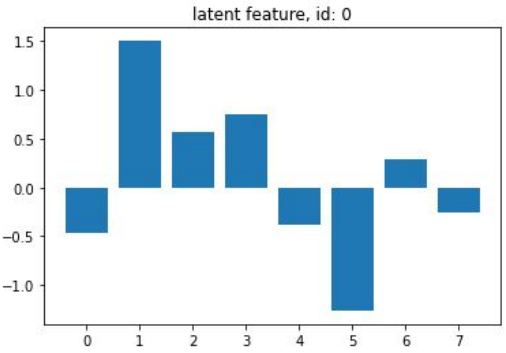
- **16**



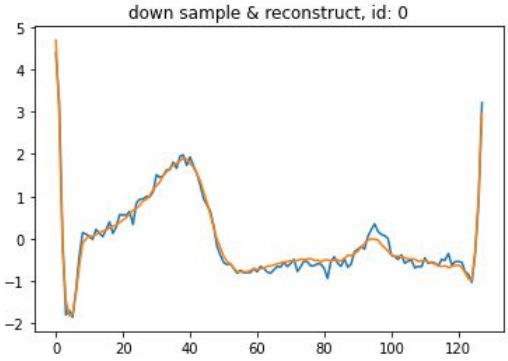


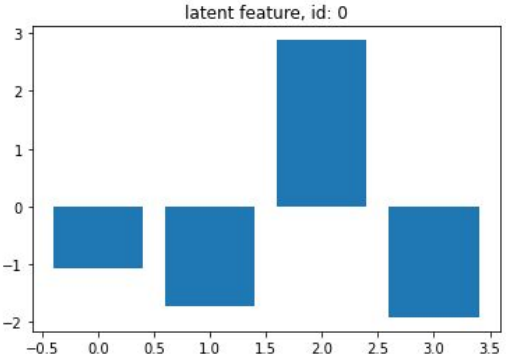
- **8**



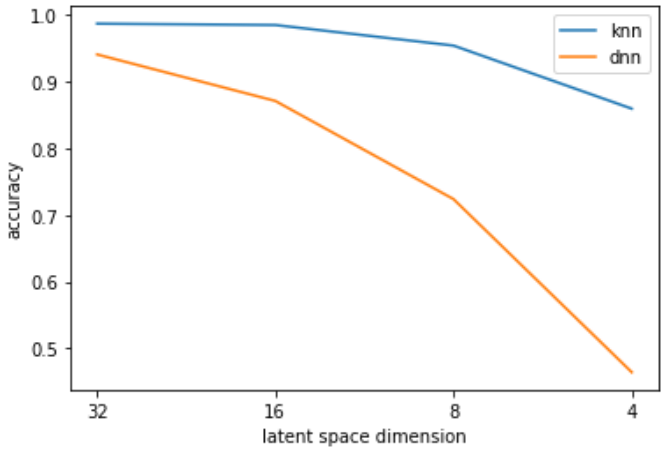


- **4**





Classify different size of latent space using KNN and DNN



Discussion

ECG signal is a time-sequence that it is not expected to perform well with machine learning models. Not to say that we cut each heartbeat into different samples according to its heart rate. While PCA is extracting the features, it may be confused with the length or some other features. Compared with the previous research by Wang et al.[1], we get quite the same good performance with the machine learning model when the condition is training with post-exercise(last 70%) and testing post-exercise(first 30%). As for LSTM, since LSTM is a kind of recurrent neural network that is suitable for this kind of signal, we get the same or even better performance than Wang et al. after slightly modifying the model design.

Our method adopts Nurmaini et al., 2019[2] designs. Use a self supervised autoencoder to extract the features. But our encoder design has changed to a LSTM model which can deal with various lengths of heartbeat sequence. Our rest-exercise, exercise-rest classification is not as good as Nurmaini et al., 2019[2]. But using our system architecture. The autoencoder can be pre-trained before applied to the real time application. After that, the KNN for identification is simply a data structure management problem. The entire system doesn't need to be retrained or finetuned. The only thing needed to do is register some samples of your heartbeat during exercise and rest. Then the system can identify your heartbeat from others by the distance threshold of the KNN.