Final Project Report: Human Activity Detection and Classification

Chenqi Zhao, Bi Zhao, Si Zhang

May 1, 2018

Content

1. Introduction

Smart devices including smart phones, watches are intensively involved in human daily life today. With the advanced sensor technologies built in those devices, human activity data detecting and collection have become easy and accurate. Human activity recognition and analysis is impactful to a variety of modern industries including health care, music and ads, retail, and security, thus it has been an active research topic in recent years. In 2013, Davide Anguita el. conducted an experiment with a group of 30 volunteers with an age bracket of 19 – 49 years. Six activities were performed by every volunteer with a smartphone on the waist. The data is extremely valuable for commercial applications: medical companies can use sensor data and user activities to keep track of bio-signals, tech companies can feed news, music or ads when users are performing different activities, e.g. sitting, running or driving, retail companies can use customer behaviors combined with GPS data to guess users' propensity to consume.

The scope of our project consists of analyzing mobile phone sensor data in the context of activity recognition. We test different machine learning methods to predict the status of the human based on the sensor data of the smartphone. The status includes static and moving. We optimized our predictions and efficiency by implementing feature selection and reduction. We also benchmark the performances of different machine learning models and discuss the reason of difference performance. We tune the hyper parameters of models to improve the prediction accuracy and discuss the underlying reasons. Finally, we try to use artificial neural network (ANN) to classify the data and analyze the result from ANN.

2. Related research

As a popular topic in machine learning area, many research works have been done to recognize human activities. Ravi et al. used shallow classifiers, such as SVM, Naive Bayes, and SVM to classify eight daily actions, which achieved the accuracy of 95% in the result. Another work (Wang et al. 2012) implemented a Naive Bayes model to recognize six daily actions to offer music recommendation. However, these works suffer from one fact that their samples are single actions, which is not sufficient for human activities because an activity is much more complex than a linear combination of a sequence of actions. The connections between single actions is more important and valuable for recognition of human activities.

Some other works explored the continuous data of human activities and achieved good recognition results. Anguita et al, the team that performed the original experiment, focused on applying a support vector machine adapted for multiclass classification, using computational efficiencies that exploit fixed-point arithmetic. This computational efficiency would allow applications build using this model to perform better on smartphones, since the approach requires less memory, processor time and power consumption. This study can recognize six statuses of human activities but cannot recognize the static or moving. However, it is more

valuable to detect the moving status of human. For example, pedometer in iPhone. Our project mainly focuses on the moving status detection based on smart phone sensor data.

3. Methods

3.1 Overall algorithm

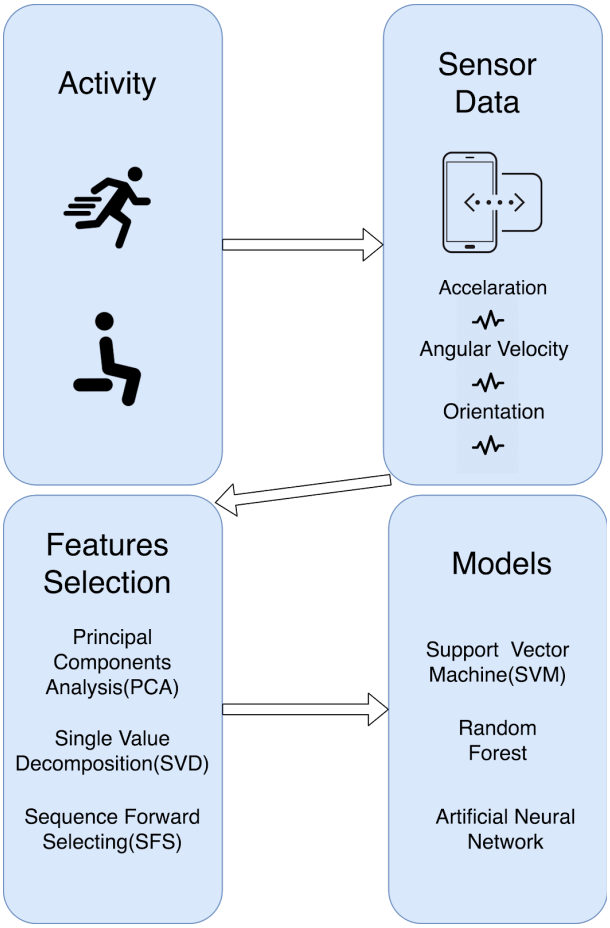


Figure.1 Flow of work

As shown in figure.1 this project mainly includes four steps. First we preprocess the data. The raw data is the collection of signals from smartphone with six classes. We binarize the data by re-classifying the data into two classes, and label them by 0 and 1, meaning static and moving. Second we train the classifiers with the original data and apply the classifiers to predict the test data. The classifiers includes SVM, random forest and artificial neural network, and calculate the accuracy of the classifiers. Third, we implement feature extraction methods to improve the performance of classifiers. PCA, SVD and SFS are applied to extract the features. Finally we

train the classifiers with the extracted features and evaluate the accuracy. The evaluation methods mainly include ROC curve and AUC.

3.2 Data Collecting and Preprocessing

The data set our project tests on is from the UCI Machine Learning Repository. It is a collection of data that generated by accelerometer and gyroscope readings from waist worn Samsung Galaxy S2 smartphone by 30 volunteers while performing standing, sitting, lying, walking, walking downstairs and walking upstairs. The smart phone captured 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz using the embedded accelerometer and gyroscope of the device. The experiments were video-recorded to label the response variables manually. Each individual observation of the data set is collection of sensor signals received over a 2.56 second sliding window and 50% overlap(128 readings per window). The signals from the window are applied different various filtering techniques to process. The feature variables were then constructed by summarites of those processed signals from time and frequency domain, including the mean, standard deviation, signal magnitude area, entropy, signal frequency, etc.

3.3 Data Information

The data contain over 10,000 observations. For each observation, there are 561 features in the data set. The response variable is the six activities, including ‘walking’, ‘walking_upstairs’, ‘walking_downstairs’, ‘sitting’, ‘standing’, and ‘laying’.

In preparation for our modeling, the dataset has been split into 70% training and 30% test data, with 21 of 30 participants in the training data and the remaining 9 participants in the testing data.

Table.1 is an overview of our training and testing data set, and the structure of our training data set, which can help us to analyze the results. We find that laying has lots of training samples, and down_stair may not have efficient number of training samples.

Data set	Number of observation
Training	7352
Testing	2947

Activity	Number of observation in training data
walking	1226
up_stairs	1073
down_stairs	986
sitting	1286
standing	1374
laying	1407

Table.1 dataset

3.4 Feature Engineering

The data set used in the project has 561 features. Large feature number can potentially lead to a very complex model, which tends to over-fit training data. Therefore it is important to use feature engineering method to shrink the feature space. There are many feature engineering methods attempt to approximate the data. Principal component analysis(PCA) is used as our feature extraction method since it explores the most useful subspace of dataset. Besides, Single value decomposition(SVD) and sequential forward selection (SFS) are also used as our feature engineering methods . The project has tested these three methods and compared the performance of them with each other.

3.4.1 Principal Components Analysis

Principal Components Analysis (PCA) attempts to identify the directions in feature space–called 'principal components' or 'PCs'–along which the data vary the most[1]. Each column of the data matrix (*X*) is centered and scaled, and the PCs are the eigenvectors of the empirical covariance matrix.

$$\Sigma = \frac{1}{m} \sum_{i=}^{m} x^{(i)} x^{(i)T}$$

For a d-dimensional summary of the data that captures as much variance as possible, we represent the data in the basis of the first d PCs, {u1, ..., ud}.[2] Here, we can compute the entries of Z as following:

$$Z_{ij} = u_j^T x^{(i)} \; for \; j = 1, \ldots, d \; and \; i = 1, \ldots, m$$

The entries are referred to as 'PC scores' and the j-th column of Z is the projection of the data onto the j-th PC.

We obtain the PCs of the train-val data, after centering and scaling the original features. We then compute the PC-based features that we will use as model input by representing the train-val data and the test data in the basis of those PCs. In Figure.2, we graph the PCA scatter plot for the data set, colored by six different activities. The result is interesting as we can see the movement (walking, walking up stairs and down stairs) is clearly separated from static activity(sitting, standing and laying). However it can be seen that the data of sitting and standing overlayed with each other therefore it will be difficult to separate them by classification.

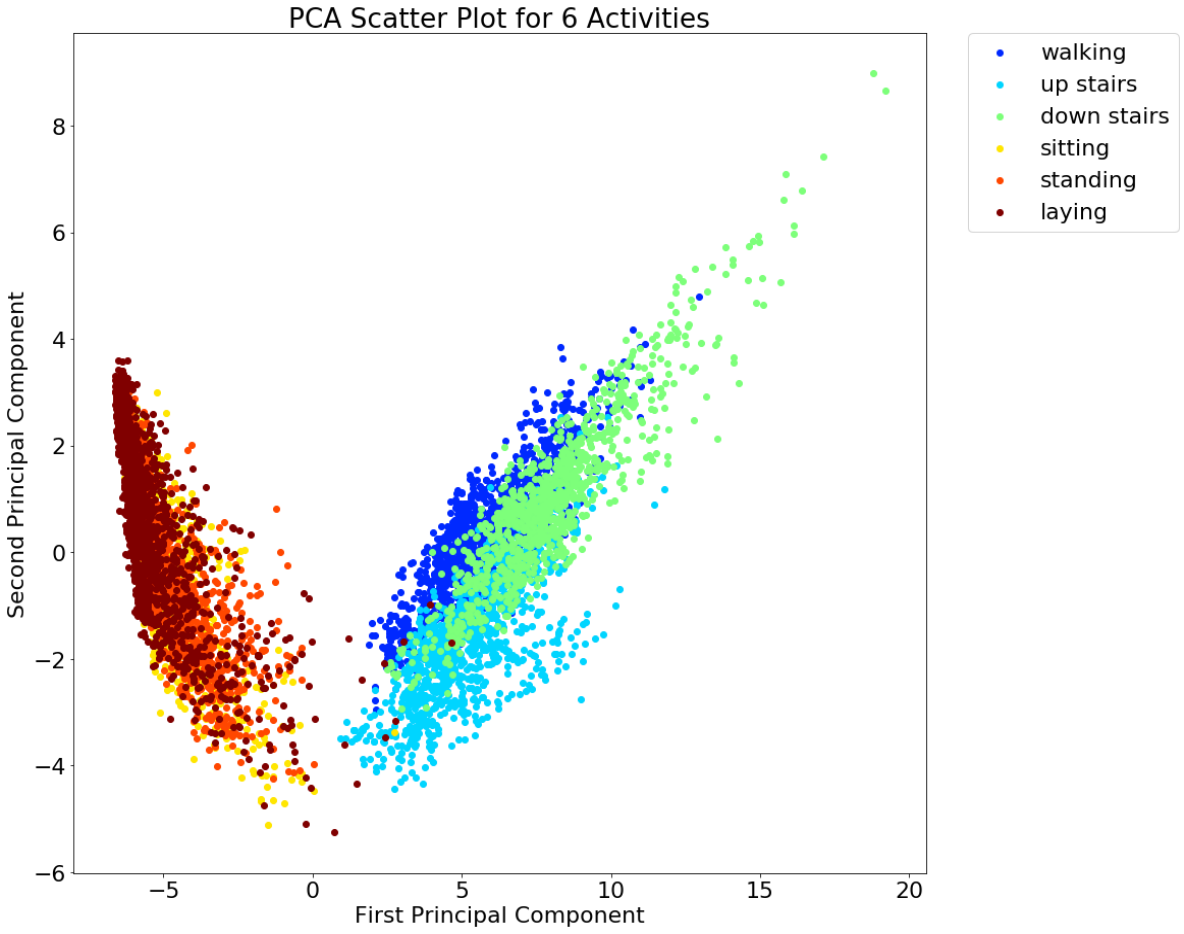


Figure.2 PCA scatter plot for 6 activities

3.4.2 Single Value Decomposition

Single value decomposition (SVD) is one of the feature extraction methods that less used compared with PCA. In our project we have tested it and this section will explain how SVD works.

The idea of SVD is to use a rank reduced approximation of a dataset to generalize some of the properties/structures. SVD factories a matrix into a product of three matrices:

$$M = U\Sigma V^*$$

U is a rotation matrix, Σ is a scaling matrix, and V^* is another rotation matrix. The eigenvalues of the original matrix are stored on the diagonal of matrix Σ. The values on the diagonal are in descending order so that the most significant eigenvalue is at the top left corner of the Σ. Therefore, SVD performs an initial rotation V^* at the first step, and then scaling the rotated matrix Σ. At last performing another rotation U to produce an arbitrary matrix.

There are two ways to reduce the feature dimension by SVD: [3]

- An approximation A is used to represent original space ($A \approx A_k = U_k \Sigma_k V_k^*$), K is smaller than the rank of E, which means only counting the first k orthogonal vectors in the original space.
- Since U is a unitary matrix, we can represent original space E by a new space $E' = U^t E = \Sigma V^*$. In this way, we can shrink dimensionality of the space. We use this method in the project.

3.4.3 Sequential forward search

The goal of feature selection is to select most significant feature from the original set. Sequential forward search(SFS) is a heuristic method to help overcome the complexity of exhaustive search. Our project use it as the feature selection method to compare it with feature extraction methods mentioned above. To put it simply, it has following steps:

1. Use k-nearest-neighbor-classifier(KNNC) as the classifier, and the leave-one-out(LOO) test for recognition rate estimate.
2. Pick one feature which has the highest LOO recognition rate among all features.
3. Select the feature among all unselected features, together with the selected features that gives the highest recognition rate.
4. Repeat the previous process until you have selected enough number of features.

3.4.4 Isomap

Isomap is a technique that resolves this problem by combining Geodesic (or curvilinear) distances between data points with MDS. Geodesic distance is the distance between two points measured over the manifold. [5] In Isomap, the geodesic distances between the data points xi are computed by constructing a neighborhood graph G, in which every data point xi is connected with its k nearest neighbors xij in the dataset X. Dijkstra's or Floyd's shortest-path algorithm could be applied to calculate the shortest path between two points and then for the Geodesic distance. [4] With the Geodesic distances between all data points, a pairwise geodesic distance matrix is constructed. By feed this distance matrix into the classical MDS, the data points xi in high-dimensional space X can be mapped to yi in the low-dimensional space Y. [4]

The general implementation of Isomap algorithm includes two steps:
First is to obtain a matrix of distance:

- Identify the nearest neighbors for each data point. The nearest neighbors can be defined as all points with the Euclidean distance falls under a certain threshold, or the closest k neighbors.
- Construct a neighbor graph by connecting all data points to their neighbors found by the first step.
- Assign weight to each edge based on Euclidean distance; estimate the geodesic distance dij between data point i and j, by calculating the sum of arc lengths along the shortest path between i and j in the graph. This can be calculated by Djikstras algorithm. The matrix of distance can be constructed by D = {dij2}

Second is to feed the matrix of distance to the classical MDS to complete the dimensional reduction. The core idea here is to replace the Euclidean distance with the Geodesic distance.

3.5 Classification

Our object is to achieve a highly accurate classifier to make prediction on data from new individuals. For this purpose, it is important to test performance of different models on individuals not in the training set. While a model trained and test on the same dataset could be more accurate, this is not objective of our project. Therefore, we have tested the performance of different classifiers on test set, and analyzed why some models performed well while others performed poorly. Models and algorithm are implemented, include: Support vector machine(SVM), random forest(RF), and artificial neural network(ANN)

3.5.1 Support vector machine

The goal for support vector machine is to design a hyperplane that classifies all training vectors in two classes. The best choice will be the hyperplane that has largest distance to the nearest training data points of any class, also called function margin. Given training vector , and a label vector , SVM solves

$$min(\tfrac{1}{2}w^T w + C\sum_{i=1}^{n} k_i),\ where\ y_i(w^T\phi(x_i) + b) \geq 1 - k_i$$

SVM use kernel function map training vectors into a higher dimensional space.

$$K(x_i, x_j) = \phi(x)_i^T \phi(x)_j$$

The kernel function has lots of types, including linear, polynomial and rbf. To find the best kernel function, we need to minimize the weight factor . During the experiment, we test SVM model with three different kernel functions. In result (table.2), we find linear kernel function gives the highest accuracy. Beyond accuracy, figure.3 shows the roc curves for three kernel function. ROC curve also shows linear kernel gives the best result. Hence, we decide to use linear kernel function to build SVM model.

Kernel function	Accuracy
polynomial	0.9539
rbf	0.9491
linear	0.9640

Table.2 accuracy of three kernel function

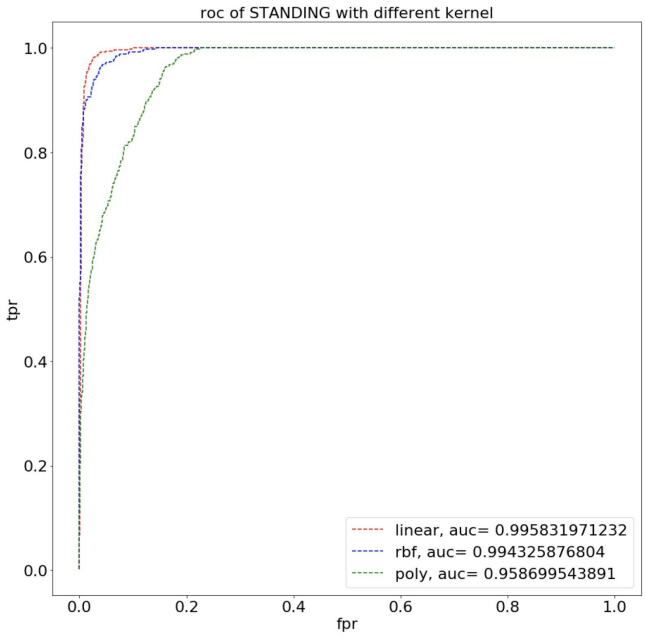


Figure.3 ROC of standing with three kernels

Support vector machine can only work for binary classification. In order to make 6 classes classification, we train 6 different support vector machines instead, and each of SVM model serves to distinguish one class from the rest of classes.

3.5.2 Random forest

Random forest is an ensemble learning method for classification, regression, etc. It constructs a multitude of decision trees at training time, and outputs classification model. Different from decision tree algorithm, random forest corrects overfitting to their training set.

During training process, each tree in the ensemble is built from a sample drawn with replacement from training set. When splitting a node during the tree construction, we no longer choose the best split. We choose a split among the random subset. Because of randomness, the variance decreases a lot, even though we have some bias on forest.

During the experiment, we test the random forest model with different number of tree during training stage, and choose the best number of trees. The result is as follow (figure.4):

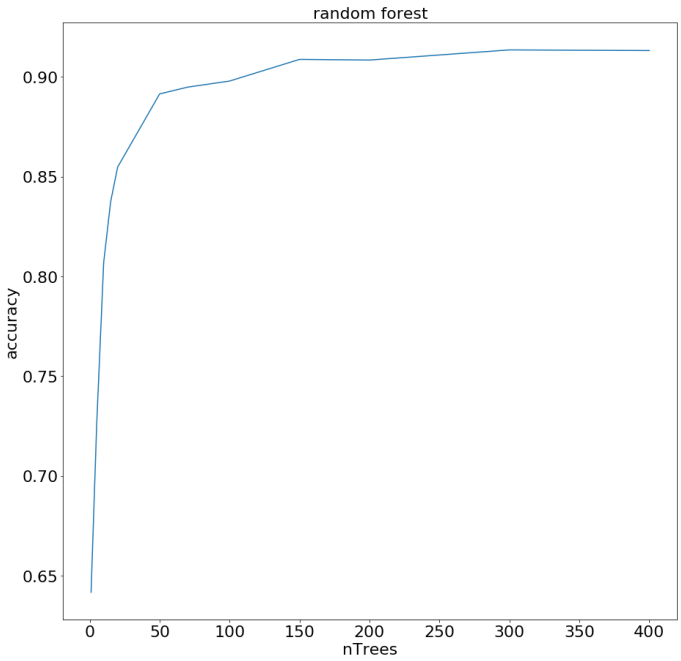


Figure.4 nTrees against accuracy

During the experiment, we find the ideal number of trees is 70. When we incresear number of tree from 1 to 70, the accuracy of classification improve a lot. However, after 70, the accuracy remain relative stable.

3.5.3 Artificial neural network

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information.

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A well trained neural network can simulate the thinking process of human brain and make decisions based on provided information. The advantages of ANN include [6]:

- Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
- Self-Organisation: An ANN can create its own organisation or representation of the information it receives during learning time.
- Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
- Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

ANN is widely applied in pattern recognition. The nodes in the network are the computing units that process the input data and generate output data. During training, the network is trained to associate outputs with input patterns. The training process of ANN is to adjust the weights of different features in the input data so that it can adapt to the output. The power of neural networks comes to life when a pattern that has no output associated with it, is given as an input. In this case, the network gives the output that corresponds to a taught input pattern that is least different from the given pattern. [6]

Architecture of ANN:

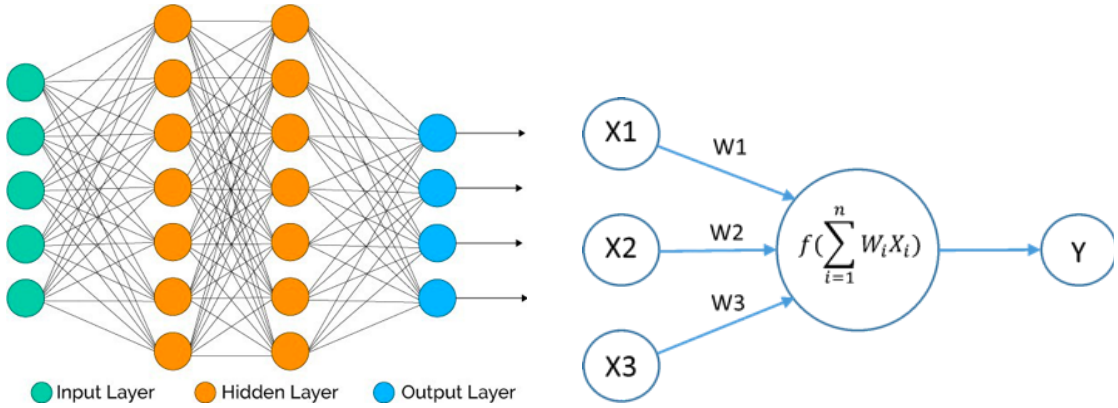


Figure.5 architecture (ANN)

An ANN if consist of input layers, hidden layers and the output layers. The lines between different layers represent the weights of each feature. The input layer takes in the input observations. The hidden layers update the weights of different features. Each node is computing unit that calculate how impactful a feature is.

Learning process of ANN:

1. initialize the weights to small and random numbers
2. set up the input nodes by input the observations to the nodes, each feature should be input to one node.
3. forward propagation: starting from the input layer, the neurons are activated by calculating the product of input feature and the associated weight in the activation function f in the neuron. The output will be the input for the next layer.
4. using the loss function to calculate the error by compare the result to the actual result in test set.
5. staring from the output layer, calculate the error by the loss function, and update the weights in each layer by lowering their impact to the error.
6. repeat the above steps and update the weights. In the reinforcement learning, the weights are updated after each observation, while in batch learning, the weights is updated after a batch of observations.
7. Repeat the step 1 through step 6, each cycle is called an epoch.

4. Results

4.1 Feature Engineering

As mentioned above there are four feature selection methods used in the project, SFS, PCA, SVD and Isomap. The figure.6 shows that the best performance of accuracy is gradually reached when the number of component of PCA is after 200 principle components for each class. It can be seen that recognition results of laying and walking are best. Walking down stairs and walking up stairs can be mixed with each other. There is also a little bit difficulty to separate standing and sitting. However, the overall results is good enough. From figure.7, we can see that the overall accuracy for six different activities can reach to the peak 95% at 250 principle components. After the peak the performance goes down hill. The reason is because curse of dimensionality causes the accuracy to decrease when the the number of selected feature is large.

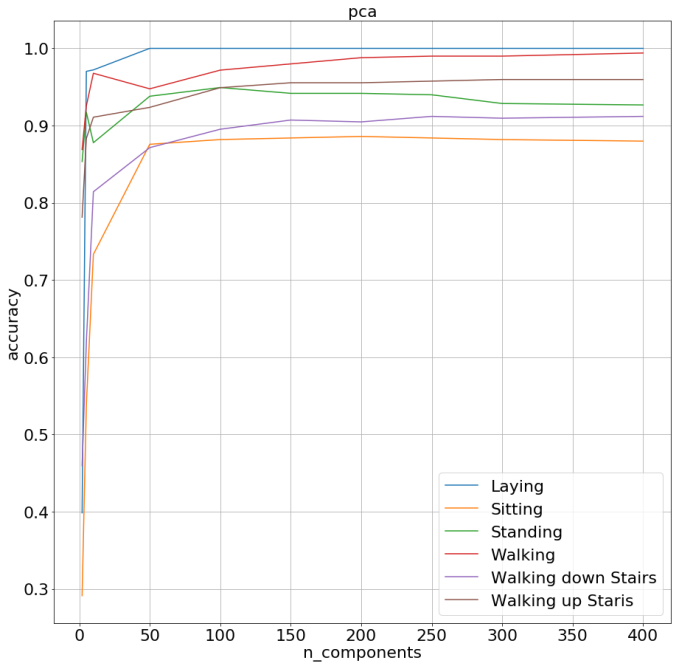


Figure.6 accuracy against number of components (PCA) for each activity

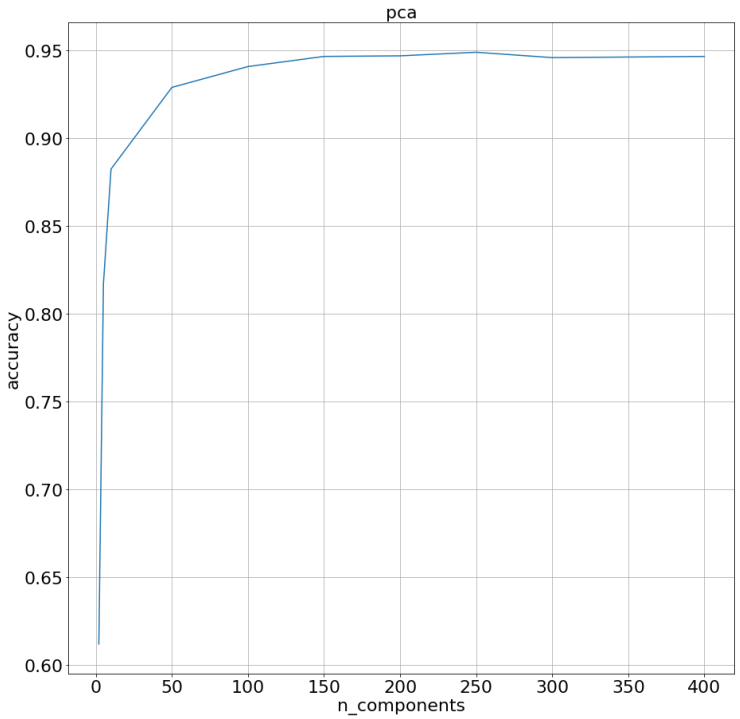


Figure.7 average accuracy against number of components (PCA) for six classes , Final choice = 250

Figure.8 shows our test result of feature selection by SFS. SFS has a good starting point compare with PCA which is around 79% with only first two features and gradually increasing to 95%, however, SFS selection is time consuming and after reach to peak the accuracy drops slightly while the accuracy by pca can still increase. The reason behind this is because after selecting best set of features from original dataset by SFS, redundant features are added which can provide irrelevant information.

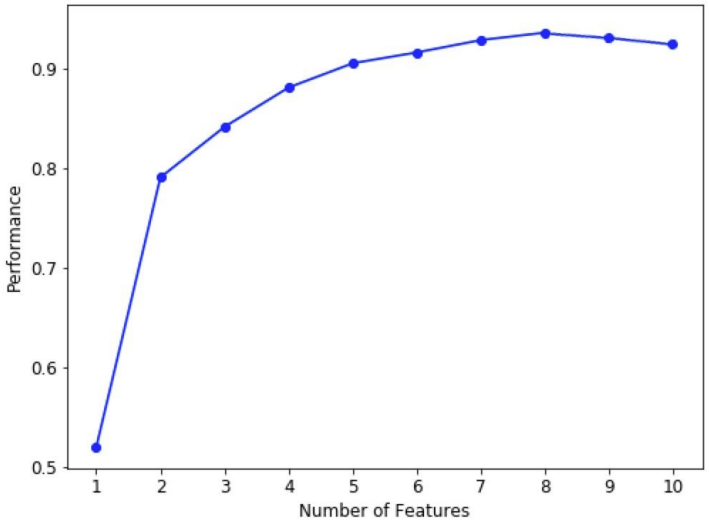


Figure.8 SFS Feature Selection

Figure.11 indicates SVD and PCA has similar results as SVD and PCA are two eigenvalue methods to map the data to lower dimensions. From the figure we can see that when the number of features selected less than 50 SVD perform worse. However, PCA is not as efficient to compute as the SVD hence there is a trade off to choose from them. In the following classification part we use PCA since it performs in term of accuracy.

We also test the isomap and the result can be clearly seen in the figure.9. There is a rise at the beginning but the accuracy stays at 91% as the numbers of features increase. In the figure.11, we compare isomap, SVD and PCA. As the figure demonstrates, the accuracy of isomap is not as good as PCA and SVD. This results can indicate that our data is not strongly non-linear as PCA and SVD can perform well on linear and isomap can generate better results when deal with strongly non-linear data like swiss roll. During the test we also find it is time consuming to run isomap algorithm which is expected as isomap needs Dijkstra's algorithm or Floyd–Warshall algorithm to calculate shortest path between its nodes‘ neighbors. Therefore, isomap is not recommended as a data dimension reduction method for our data.

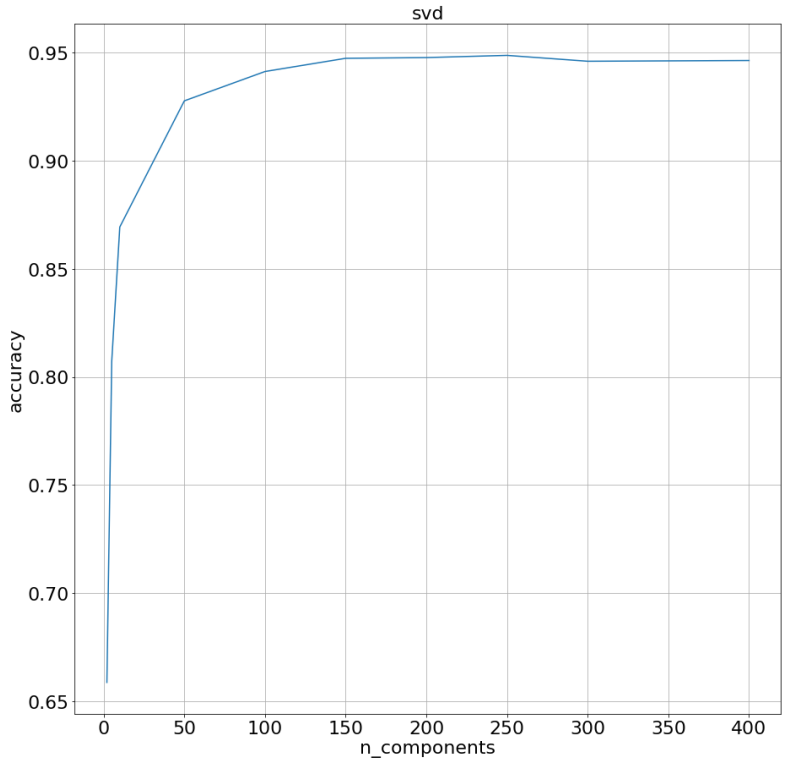


Figure.9 Single value decomposition (SVD), final choice 260

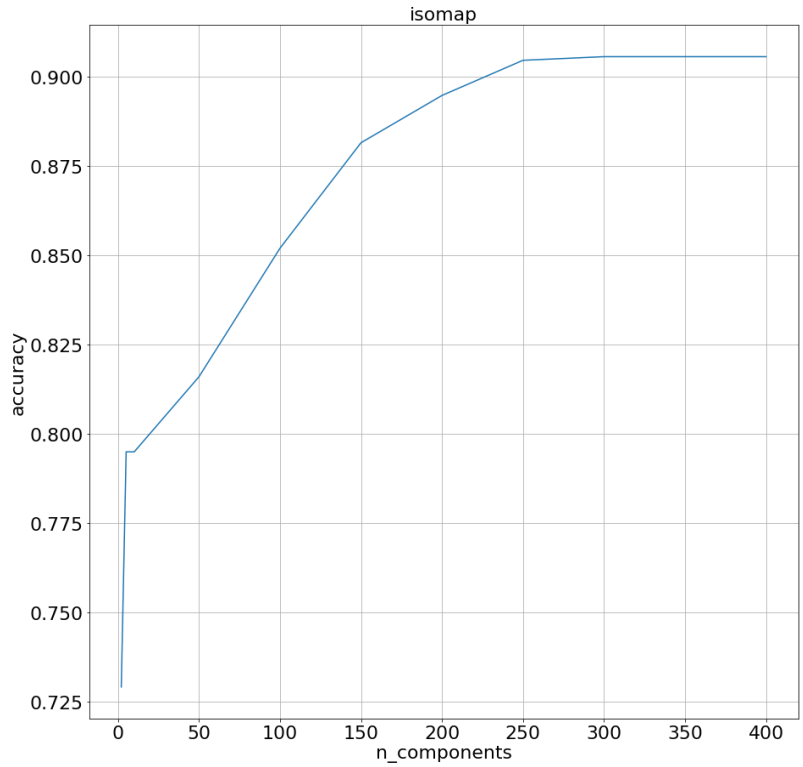


Figure.10 Isomap

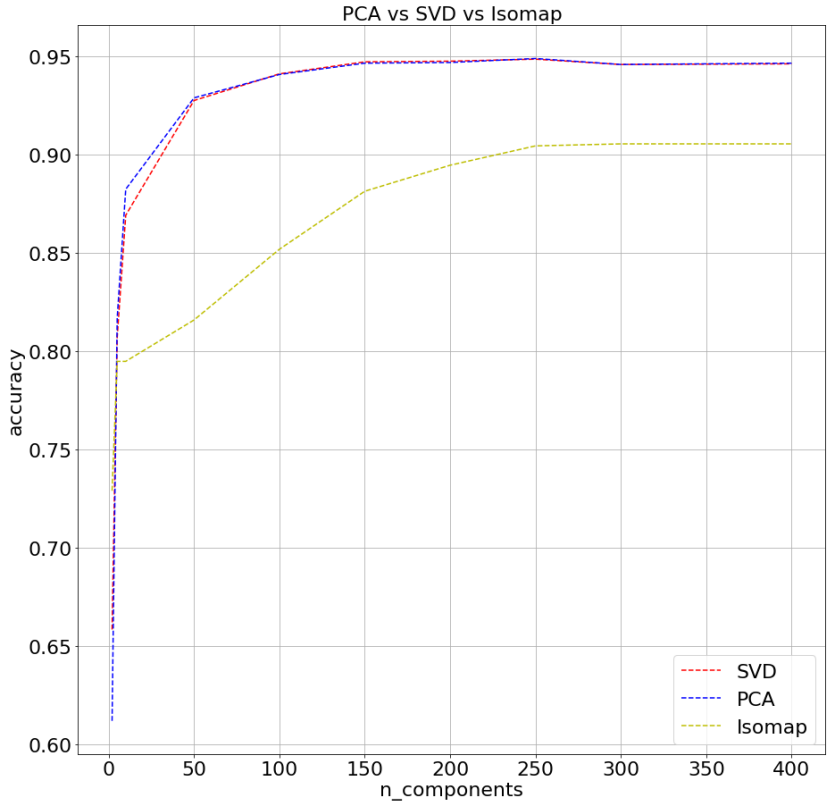


Figure.11 Compare svd and pca

4.2 Classification

4.2.1 Support vector machine

In figure.12, we present ROC curves for six classes. We can find that sitting and standing activities are the most difficult problems, especially sitting. Combining the confusion matrix result (figure.13), we can find the classifier always classify sitting into standing, and standing into sitting. We can draw a conclusion that sitting and standing is difficult to distinguish from each other, which makes sense, since the difference of cell phone activity between sitting person and standing person is small.

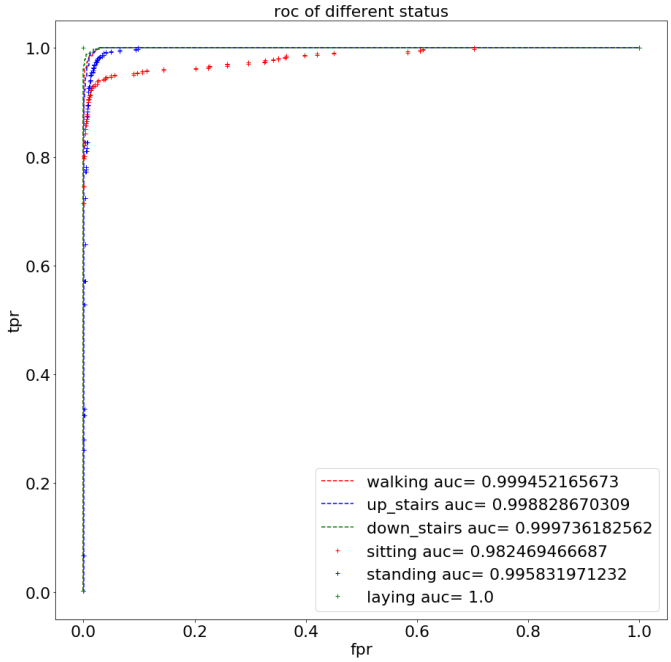


Figure.12 ROC of six classes (svm)

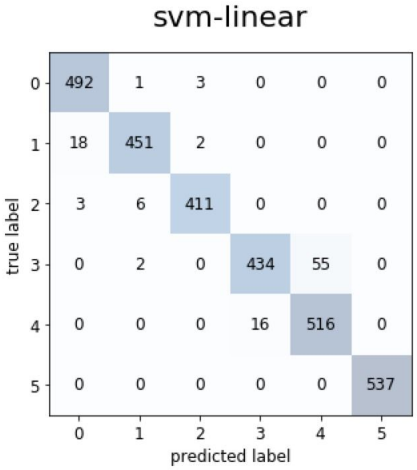


Figure.13 Confusion matrix

Table.3 is Area under curve (AUC) corresponding to six classes. It indicates that laying is the easiest problem to solve, since its AUC equals to 1. This result can be caused by two reasons. First, the movement of laying is quite different other activities. Second, laying has the largest number of training samples.

Class	Area under curve
Walking	0.9994
Up_stairs	0.9988
Down_stairs	0.9997
Sitting	0.9825
Standing	0.9958
Laying	1.0

Table.3 AUC of six classes (svm)

4.2.2 random forest

Figure.14 shows down_stairs is a trouble class. We believe this is because of insufficient training, since down_stairs has the smallest number of training sample compared to the others. What's more, the overall performance of random forest is worse than support vector machine, based on roc curves and confusion matrix. Figure.15 shows the moving activities, including walking, up_stairs, and down_stairs are confused for each other. Some static activities, including sitting and standing, are also difficult to recognize. According to the confusion matrix, we can have another conclusion that moving activities can separate well from static activities.

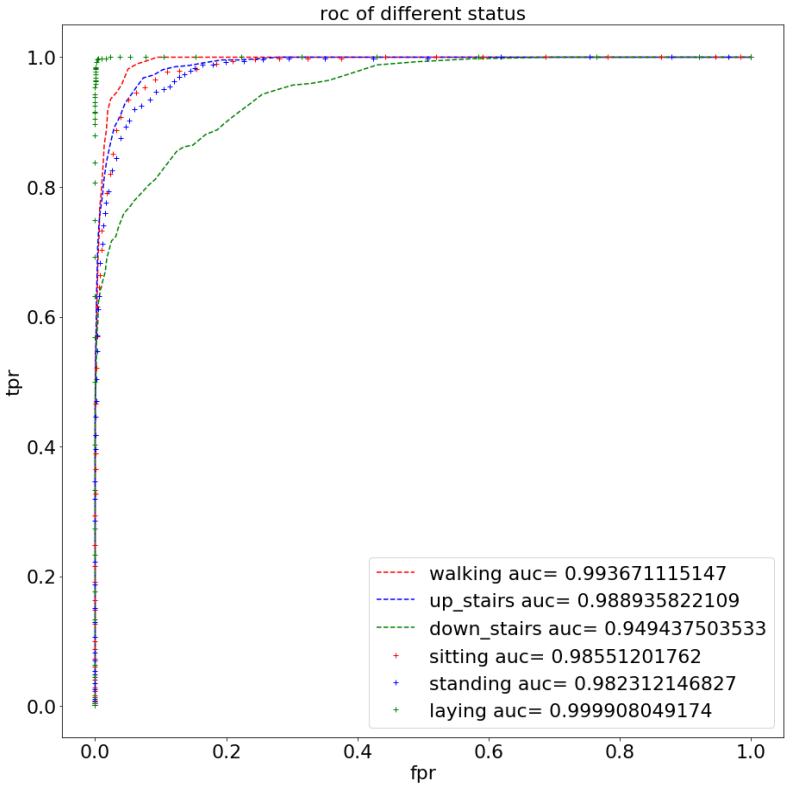


Figure.14 ROC of six classes (RF)

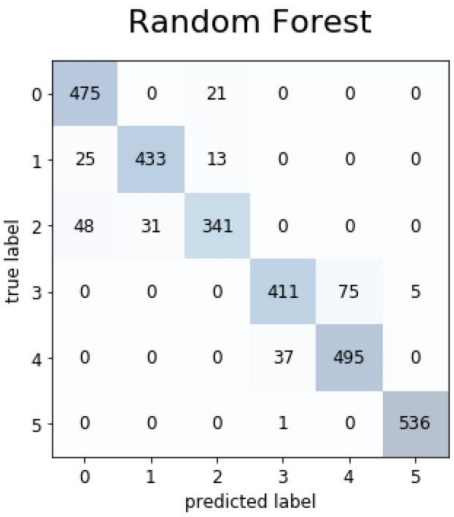


Figure.15 Confusion matrix

Table.4 shows the AUC for each class. We can still find that laying class has the best result compared to other classes, which is similar to SVM results.

Class	Area under curve
Walking	0.9936
Up_stairs	0.9889
Down_stairs	0.9494
Sitting	0.9855
Standing	0.9823
Laying	0.9999

Tabel.4 AUC of six classes (RF)

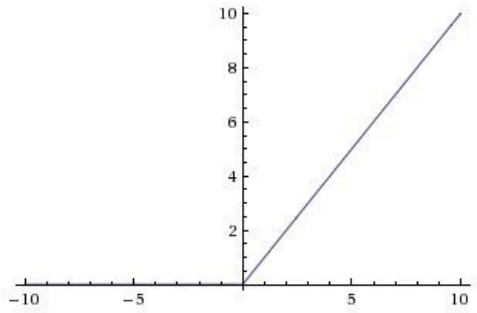
4.2.3 Artificial neural network

Activation functions:

In our experiment, we implemented an broad network with one hidden layer, the activation function in hidden layer is rectified linear unit (ReLU). [10]

ReLU function is a popular activation function. The expression of the ReLU is:

$$f(x) = x \mathrel{+}= max(0, x)$$



It gives an output x if x is positive and 0 otherwise. Obviously this function non-linear, and the combain of ReLUs is also non-linear. The expression is simple enough to approximate most of function outputs by combining. Some other advantages of *ReLU* includes: []

- The networks is lighter: if the half of the outputs from previous layer are 0, 50% of neurons in the current layer will be activated, which is much lighter compared with sigmoid activation (sigmoid activate almost all of the neurons).
- Less computation: the computation of ReLU is much easier than other functions like sigmoid. Therefore it will be faster and requires less computation resources.

At the output layer, *softmax* function is applied as the activation function. [9]

$$f(x) = \frac{e^{xi}}{\sum_{j=1}^{n} e^{xj}}$$

The softmax function determine the output by calculating the the probabilities of each target class over all possible target classes. For the multi-classes data, the output will be the

probability of each class, and the sum of the probability will be 1. Thus the class with the highest probability will be the predicted result. This activation function works very well for multi-class classifications.

In order to explore the impact of neural network structure to the prediction accuracy of the network, we implemented ANNs with different number of hidden layers (depth) and different number of nodes(breadth). The broadness varies from 6 to 200, with step of 3, and depth varies from 1 to 10, with step of 2. The result of prediction result vs. broadness and depth are listed below.

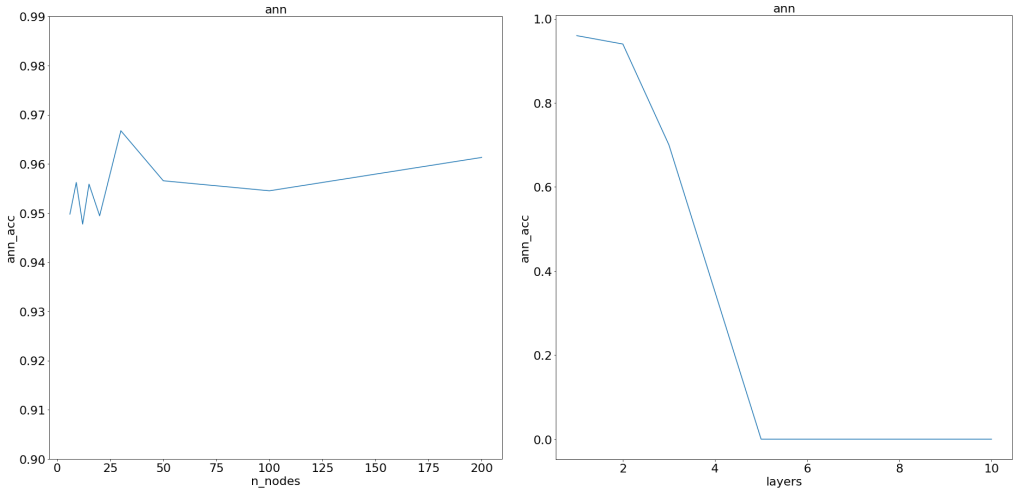


Figure.16 left: relation between ANN accuracy with number of nodes in each hidden layer. right: relation between ANN accuracy with number of hidden layers.

From the result of figure 16, we find that for this dataset, the single hidden layer with 25 nodes structure can achieve the highest accuracy of 0.967. The accuracy of deep structure with more than two layers drops significantly with increasing number of layers. We may conclude that for this dataset, the broad network works much better than deep network. The reason could be the broad structure with more nodes in hidden layer generates better classifiers with more combinations of ReLU functions. While the deep structure can extract the complex pattern from data, it is easy to over-fit the training data. In our experiment, for the network with more than 4 layers, the accuracy in training stayed at 0.18 and did not increase after each epoches. One of the reasons could be the network trys to fit into one pattern as perfect as possible. The calculated error is very large at the output layer, and the network tries to reduce the error by fitting into another pattern. Within the limited epoches, the network fails to find the really good pattern for all features. We believe that is the reason of the extremely low accuracy for the structures with more than 4 layers.[7]

Figure.17 shows ANN performs well in most of classes except for sitting and standing, which is very similar to the result of SVM. The reason could be that the some of features of sitting and standing overlaps and this confused the classifier.

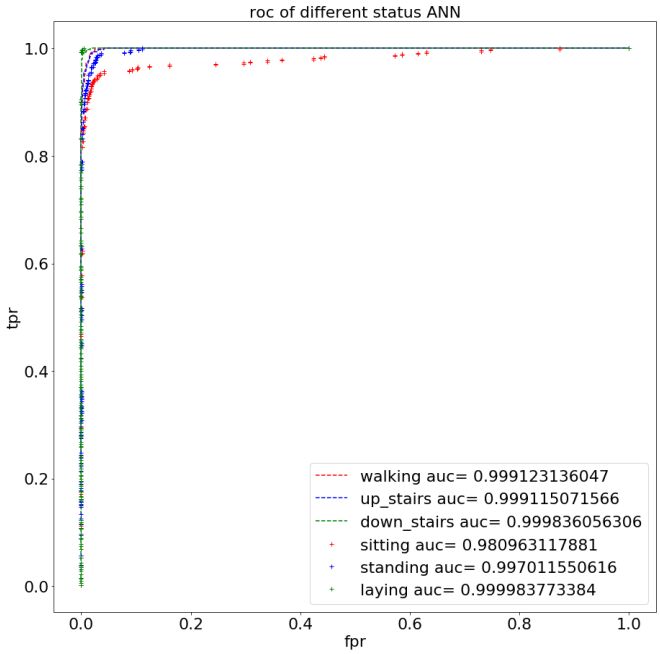


Figure.17 ROC of six classes (ANN)

Table.5 shows the AUC for each class. We can still find that laying class has the best result compared to other classes, which is similar to SVM results.

Class	Area under curve
Walking	0.999
Up_stairs	0.999
Down_stairs	0.999
Sitting	0.981
Standing	0.997
Laying	0.9999

Tabel.5 AUC of six classes (ANN)

5. Conclusion

In this project, a set of human activity data collected from a real experiment is studied. The raw data is interpreted and transformed to fit our classifiers. The data is analyzed and preprocessed by feature engineering methodologies including PCA, SVD, isomap and SFS. The performances and efficiency of these three methods are compared.

- **Data:** Laying is the easiest activity. Down_stair, standing and sitting are much harder activities for detection. We believe it is because of two reasons.
    - Sufficient training in laying, and insufficient training in other activities. For difficult activities, we need to increase the number of training sample and also training diversity.
    - Most static activities are difficult to detect, because it looks similar to each other via cell phone system.
- **Feature engineering**: Feature extraction methods (PCA, SVD) significantly improved the performance of classifiers because the large number of features of our data.
- **Classifier implementation and optimization**: Three kinds of classifiers, SVM, Random Forest, and Artificial Neural Network are implemented and optimized.
- **Accuracy**: Applying ANN, we are able to achieve 96.69% accuracy while the accuracy from Random Forest and Support vector machine are 91.34% and 96.40% respectively.

Classifier	Accuracy	Running time(s)
SVM (linear)	0.9640	2.14
RF	0.9134	5.67
ANN	0.9667	55.3

Table.5 accuracy of three classifiers

- **Running time**: The running time of Artificial Neural Network is the longest due to its cyclic learning process, and SVM is slightly faster than Random Forest. From the results above

In conclusion, we believe that ANN is the best classification model for this dataset, if we do not take running time into consideration. However, if people want to concentrate on both accuracy and running, we suggest using support vector machine.

6. Future work

The project might be improved in the following ways:

Data: Currently the dataset has 7352 observations with 6 classes. This dataset is relatively small to retrieve the information that is really valuable. If the dataset has more observations with more segmented classes, we could retrieve more interesting information. For example, we might be able to recognize that a person goes hiking regularly, and we could send more sport equipment to him/her.

Feature engineering: Feature engineering significantly improved the performance of our classification, but we find that most of misclassification occurs between “sitting” class and

“standing” class. We can improve the feature engineering work to reduce the misclassification by retrieving more specific feature of these two class, separately.

Classification models: In this project we applied SVM, random forest and artificial neural network classifiers, and achieve relatively good prediction result. However, there are some hyperparameters that have not been tuned. Optimizing those hyperparameters may result in better predictions.

We tried broad neural network and deep network, but we did not try broad and deep network. This structure combine the strength of both broad and deep networks by jointly training a wide linear model (for memorization) alongside a deep neural network [8], and it might give us better prediction.

REFERENCES

[1] T. H. G. James, D. Witten and R. Tibshirani, Introduction to Statistical Learning with Applications in R, 6th ed. (Springer, 2013).

[2] A. Ng and J. Duchi, CS229: Machine Learning - Course Notes. (Autumn 2016).

[3] Francesca, Fallucchi, and Fabio Massimo Zanzotto. "SVD feature selection for probabilistic taxonomy learning." Proceedings of the workshop on geometrical models of natural language semantics. Association for Computational Linguistics, 2009.

[4] Van der Maaten, L. J. P. "An introduction to dimensionality reduction using matlab." Report 1201.07-07 (2007): 62.

[5] Tenenbaum, Joshua B., Vin De Silva, and John C. Langford. "A global geometric framework for nonlinear dimensionality reduction." science 290.5500 (2000): 2319-2323.Liu, Nian, and Nayyar

[6] https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

[7] A. Zaidi. "Artificial Neural Network: Deep or Broad? An Empirical Study." Australasian Joint Conference on Artificial Intelligence. Springer, Cham, 2016.8

[8] Cheng, Heng-Tze, et al. "Wide & deep learning for recommender systems." Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. ACM, 2016.

[9]http://dataaspirant.com/2017/03/07/difference-between-softmax-function-and-sigmoid-function/

[10] https://en.wikipedia.org/wiki/Rectifier_(neural_networks)