Spring 2020

Human Avtivity Recognition Using Smartphones

Chenhang Xu

Human Activity Recognition Using Smartphones

by

Chenhang Xu

A report submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Electrical Engineering

Program of Study Committee:
Joseph Zambreno, Major Professor

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation/thesis. The Graduate College will ensure this dissertation/thesis is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2020

DEDICATION

I would like to dedicate this thesis to my parents Xu and Qiu. Without their support I have no change to be a master student.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like to take this opportunity to express my thanks to those who helped me write this thesis. First and foremost, Dr. Zambreno for his guidance, patience and support throughout this research and the writing of this thesis. I would also like to thank my friend Kebei Wang, who taught me some machine learning algorithms. Additionally, I would like to thank Zhang, who assisted me with the thesis format.

ABSTRACT

Human activity recognition is a problem of classifying activity of a human using accelerometer and gyroscope data recorded by smartphones into well-defined movements. However, it is still a challenging problem to compute a large number of observations in each second and it is also hard to translate the measurements from smartphones into physical activity patterns. In the past few decades, researchers have designed different human activities recognition systems using different algorithms. In this paper, I investigate different approaches to HAR system based on smartphones. I provide a comparison of these approaches and provide recommendations of the best performing solution.

CHAPTER 1. OVERVIEW

1.1 Introduction

The aging population increased fast in recent years, which become a big challenge for societies [1]. The elderly population needs more healthcare than young people, for they are unable to take care of themselves [2]. It is hard for the family member to pay extra attention to them outside their daily care. If someone wants to have the elderly care in the hospital, the cost of the professional nurse is very high and result in financial pressure. The most effective way to solve this problem and reduce the cost is wearable devices [3].

"This field is the first component of the sequence for achieving Smarter Interactive Cognitive Environment together with data analysis, decision making and taking action, and our subject of research." [4]. As the author said in his article, human activity recognition is an active field for understanding peoples' behaviors by exploring the data.

There are several approaches used for human activity recognition. Recently, the most up to date researches on this field divided into two ways, such as vision-based recognition and sensor-based recognition [5, 6]. The vision-based recognition uses the 2D video recorder or a depth sensor. Human activities are recognized from RGB data, which collected from the camera and conjugate with the depth data. However, older people not only always stay in the home but also going outside for some reason. It is hard for vision-based HAR to monitor people's activity outside, and it is also very expensive to install the camera inside the room. The accuracy of vision-based HAR will decrease if the indoor environment is not as good as possible, such as the room is too dark, and the distance between people and sensors is too far [7].

In today's world, with the development of MEMS technologies, sensors are becoming less expensive and small [8]. More sensors have been built into smartphones to track peoples' movements and positions, such as accelerometers and gyroscope. So a huge amount of resources can be saved when these sensors can monitor the people all the time and report immediately if an unusual movement happened.

Data collection from accelerometers and gyroscope is time-series data, and it is a complex process to analysis on these data. First of all, we need to smooth and normalize the data. Then we need to segment the data and extract the feature from the data before we apply classification algorithms on it. It relies heavily on handcrafted feature extraction [9], and it needs strong background knowledge to do this. Domain features have mean, variance, median and so on. The frequency-domain features [10] have discrete cosine transform coefficients and fast Fourier transform coefficients. If the dimension of the data is too high, we need to consider principal component analysis(PCA) to reduce the dimension of the extracted features. Then we can apply support vector machine(SVM), K nearest neighbor(KNN), decision tree(DT), random forest to classify extracted features. However, these machine learning algorithms can only get a satisfying result in multiclass classification problems. Therefore, deep learning algorithms have been introduced to solve the HAR problem [11]. This kind of method can learn automatically and avoid the manual feature extraction step. However, deep learning also has a disadvantage of need a large dataset to train the model. In fact, it is sometimes difficult to access such a large dataset and inappropriate for the real-time classification problem for it needs high computational load [12].

In this paper, I will try to use several algorithms such as SVM, KNN, PCA, and decision trees to build the model and compare them together.

1.2 Data Exploration

The dataset I used for the paper is from the UCI Machine Learning Repository. The data was collected from 30 volunteers aged from 19 to 48 years old. They wore a Samsung Galaxy SII on their waist and used the accelerometer and gyroscope to record the data. There are six activities in this dataset: standing, sitting, lying, walking, walking downstairs, and walking upstairs. This dataset also included transitional postures such as stand-to-sit, sit-to-stand, sit-to-lie, lie-to-sit, stand-to-lie, and lie-to-stand. These experiments have been labeled manually and video recorded. The dataset has been randomly split into test and training sets at the ratio of 70:30.

The training dataset has 7352 rows and 563 columns and the testing dataset has 2947 rows and 563 columns. In this dataset, the last two columns are subject and activities(Figure 1.1), and they are our labels. In this case, the feature size for this dataset is 561.

angle(X,gravityMean)	angle(Y,gravityMean)	angle(Z,gravityMean)	subject	Activity
-0.841247	0.179941	-0.058627	1	STANDING

Figure 1.1 561 feature size

The author has done the preprocessing steps. They were using a noise filter to preprocess the accelerometer and gyroscope. Each data in this dataset is a construction of sensor signals received a 2.56s fixed window with 50% overlap.

1.3 Data Visualization

The most important thing to preprocess the data is to make sure that all classes are almost the same size. This step is important so that the algorithms we apply to the dataset are not biased towards any of class. So I need to shuffle the dataset and count the number for each type of activity in this dataset.

WALKING	209
STANDING	179
LAYING	164
WALKING_UPSTAIRS	159
WALKING_DOWNSTAIRS	145
SITTING	143

Figure 1.2 Value count for training

On observing the Table 1.2, we can find that the number of each activity is almost the same. In this case, the dataset is ready to use and formed well.

To better understand our dataset and the distinctions between each activity. PCA is a handy tool to visualize the data. As I knew from the following work, when we decrease the feature size from 561 to 3, these three principal components can still contain 75% variance.

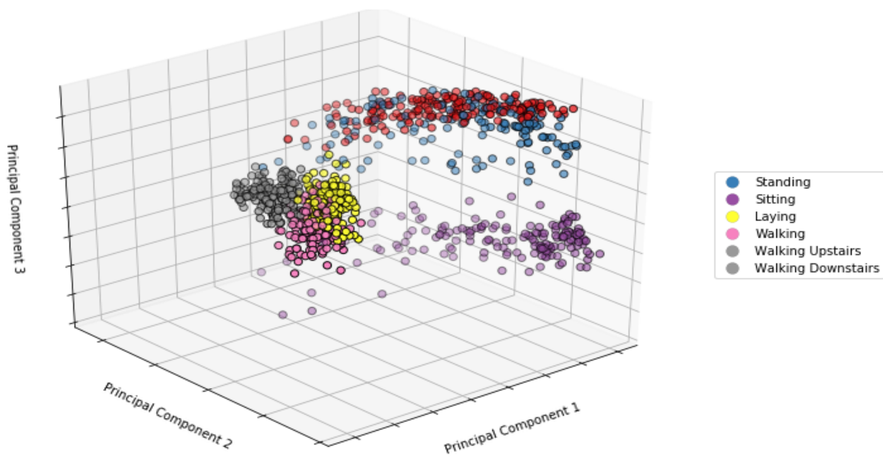

Figure 1.3 3D plot for traning

From the Figure 1.3, we can easily find that there are two groups (static and dynamic). These six activities can be reasonably isolated. For the static activities, only a little data has been overlapped. Within six activities walking upstairs and walking downstairs overlap most, which match the confusion matrix result for the following algorithms.

CHAPTER 2. REVIEW OF LITERATURE

In the following section, many different approaches to solving human activity recognition problems have been discussed. As we known from the dataset, this dataset contains six activities and each activity represents by 561 features. So it is a multi classes and high dimensional problem. Classical approaches to this problem are based on fixed-size windows and training machine learning models, such as random forest, decision trees [13], K nearest neighbor [14] and SVM [15, 16, 17]. However, in today's world, deep learning methods such as recurrent neural network and convolution neural network has been used to solve this problem.

The first approach for people to solve the problem is the support vector machine (SVM). Anguita [18] and his team performed the original experiment, and they used the SVM to solve the multiclass classification problem. They used the one-vs-all approach to implement SVM, and the hyper-parameters were selected through a 10-fold cross-validation procedure. The accuracy of their experiment is 96%, which is much higher than other algorithms at that time. At that time, they still need to consider power consumption. By using this kind of approach, they can apply this method on smartphones because it needs less memory and power consumption, but the accuracy remains the same as other algorithms at that time.

The second traditional approach to solve the multiclass classification problem is the decision tree. Bao and Stenphen [19] developed the algorithms to detect human's everyday tasks by collecting data from 5 accelerometers. They test several algorithms on it, and the decision trees get the best result, which is 84% accuracy. The results for this experiment show that some activities are classified correctly with a subject-independent training dataset; others still need a subject-specific training dataset. Different from other researchers using several accelerometers to collect the data.

Because 561 features represent each activity, if we keep using the full dataset, it will cost us more time to compute the result. When we faced the high dimension problem, it is very important for us to reduce the dimension using principal component analysis (PCA). Jin and He [20] build the system only based on a single tri-axis accelerometer to monitor human activities. This system used the discrete cosine transform, PCA, and SVM to explore the data and classification of human activities. The accuracy is around 96%, which is higher than other approaches. Kishor [21] also combined PCA with an artificial neural network(ANN). The purpose for them to use PCA is to reduce processing time. They reduce 561 features of raw data to 70 features, and the dataset still keeps the most important information. From my research, if we keep 100 principal components, the dataset will keep 95% variance, and the variance will decrease to around 85% if we keep 75 principal components. Then they test the multi-layer perceptron classifier on these principal components. The accuracy for this approach is around 96% and the computational time reduced from 658s to 128s.

Another approach for the classification problem is K nearest neighbor. As we know from the study, the accuracy and time consume depend on the quality of the feature representation used for this dataset. In Sadiq Sani' s [22] paper, they used different feature representation approaches for the HAR dataset, from shallow handcraft features to frequency transform features. They choose five different features ranging from shallow to deep, and found the deep features that can produce the best result. Almontazer [23] also given a paper related to KNN, which showed this algorithm is easy to apply and the accuracy is very high.

For the deep learning methods, one of the significant research using the convolutional neural network was published by Ming Zeng [24]. He designed a CNN model for accelerometer data, and the data was collected by the accelerometer in three axes (X, Y, Z). These data will be fed into the convolutional layers and pooling layers. Before they were interpreted by the hidden layers, they will concatenate together.

By reviewing several papers related to CNN, there are many ways to solve the HAR problem using CNN. In Sang' s [25] paper, they built a one-dimensional convolutional neural network for HAR. They divided the activities into two groups: static(sitting, lying, standing) and dynamic(walking, walking upstairs, walking downstairs). In this case, the problem became two-stage learning of multiple 1D CNN problem. The first binary classifier will recognize whether it is a static activity or dynamic activity. Then the second multiclass 1D CNN will recognize the individual activities.

The recurrent neural network (RNN) is designed to learn from the sequence data. Long short term memory network is one of the RNN, which can be used to the problem of HAR. In Abdulmajid's [26] paper, they explore the use of LSTMs on many architectures(unidirectional, bidirectional and cascaded). LSTM will predict the activity based on the subsequence of the sensor data in each time step. The results show that the LSTM model is better than SVM and KNN, but compared with CNN, the performance is almost the same.

CHAPTER 3. Machine Learning Algorithms

3.1 Principle Component Analysis (PCA)

The reason for me to use the principal component analysis is the dataset has 561 features. It will cause high data error if we try to use the whole dataset and increase the computational complexity. So I am going to use PCA to reduce the number of features in this dataset [27]. From my point of view, the PCA problem can be known as a probabilistic problem. The data are random vectors from a distribution with zero means and some covariance matrix. We are going to find the best rank-k subspace to fit the data. If we have a d-dimensional random vector. The mean and covariance matrix for it is 0 and $\sum = [xx^T]$. Then the eigenvalue decomposition for this matrix should be $\sum = U \wedge U^T$. In this formula, is a diagonal matrix and every eigenvalue in this matrix arranged in decreasing order. Our goal is to find the top-k eigenvectors, which can represent most of the dataset [28].

Principal components are uncorrelated and they are ordered by the variation present in all of the original variables. The following figure (Figure 3.1) shows the result. From the plot, we can easily find over 95% variance can be represented by the first 100 components. After we get this result, we can believe that using these 100 principal components, we can recover most of the critical characteristics of the dataset. In this case, we can only choose the first 100 principal components to calculate the result. By ignoring other components, we can reduce the computational complexity and we will not affect the accuracy a lot. Then I zoom in on the first 100 components to have a closer look of the relationship. From Figure 3.2, we can find the first 3 components can represent over 75% variance. In this case, we know that 561 features can be reduced to 100 to get almost the same result. If we still want to get less computing time, we can decrease the feature sizes to 3.

Then I applied the KNN algorithm to the PCA model. As we can see from the Table 3.1, if we train the model with the full dataset, the accuracy should be 86.8% and the running time should be 1320ms, which takes a long time to compute the result. After we decrease the feature size to 100, the accuracy almost remains the same, but the computational time decreased to 568ms. I also tried to decrease the feature size to 3 and the accuracy is 77.9%. The accuracy decreased a lot, but it was still acceptable and it used the least computational time(134ms). From my point of view, PCA is very useful in high dimensional dataset because it can decrease the running time but the accuracy is still very high.

Table 3.1 Feature Size VS Accuracy

Feature Sizes	**Accuracy(%)**	**Time(ms)**
561	87.2	1320
100	86.7	568
3	77.9	134

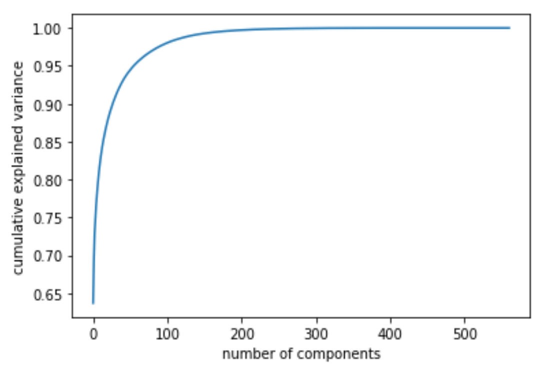

Figure 3.1 Variance versus the number of components

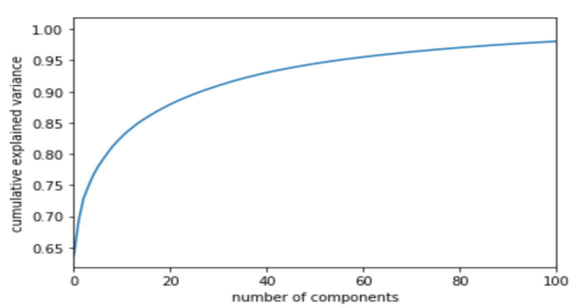

Figure 3.2 Variance versus the number of components - zoomed into 100 components

3.2 Support Vector Machine (SVM)

In the past few years, there are many machine learning algorithms, which can be used for classifiers such as Markov Chain [29], Naive Bayes, SVM and so on. While it is not clear that SVM is the best solution for human activity recognition, it has been used in many related fields such as handwriting recognition, speech recognition, and gesture recognition. Consequently I used the SVM algorithm to build the model. SVM is based on decision hyper-planes to define decision boundaries. As Krause [30] said in his article, SVM is trying to maximize the decision boundary between hyper-planes. By exploring the dataset, I find some clusters fully overlap, and other clusters overlap a little. In this case, we need to maximize the margins when classifying these activities to get a good result.

I also used the linear kernel for the SVM classifier. The reason for me to use the kernel is the system is non-linear in the lower dimension. But we know the SVM is a linear classifier, so to build a linear classifier, we need to input the low dimensional data into high dimensional.

Then I will also introduce the threshold to ignore the noise for the dataset. If there are two groups red and green, a red point is in the group green. The whole system cannot be built if we don't ignore the noise point. So for my model, I accept some samples, which distance is less than 1. Because I will not count these samples.

SVM can only solve binary problem, for the multi class classification problem, we have two ways to solve it(One VS. One and One VS. All). I will choose to implement SVM using One VS. One approach that trains a separate classifier for each different pairs of labels and it always outperforms than One VS. All approach.

3.3 K-Nearest Neighbour (KNN)

KNN is another simple supervised machine learning algorithm to solve classification problems. It is easy to use and understand but the speed can be very slow if we use this algorithm on a large dataset because it only tests on the memory. KNN is an instance-based classifier, and it works by finding the instance between a query and all samples in the dataset. It will select K examples closest to the query. In this paper, I used the Euclidean distance function as my learning function.

So one of the most important things for the KNN algorithm is how to choose K. If K = 1, the model will fit the training set perfectly, but it will get very high test error. Because at K=1, we were over-fitting the boundaries. When we increase the K value, the boundaries are more consistent and reasonable. The error rate will keep decreasing and reach a minimum value at a K value. After we keep increasing K, the model will under-fit the test data.

As we can see from Figure 3.3, I had determined the value of K by plotting graph test data accuracy vs. the number of K(from 0 to 50). From the figure, we can find when the number of K is around 40 to 45, this KNN model will achieve the best result, which is around 78% accuracy. Although this algorithm is straightforward to apply, it still takes a long time to compute the result. This computational time is much higher than other algorithms.

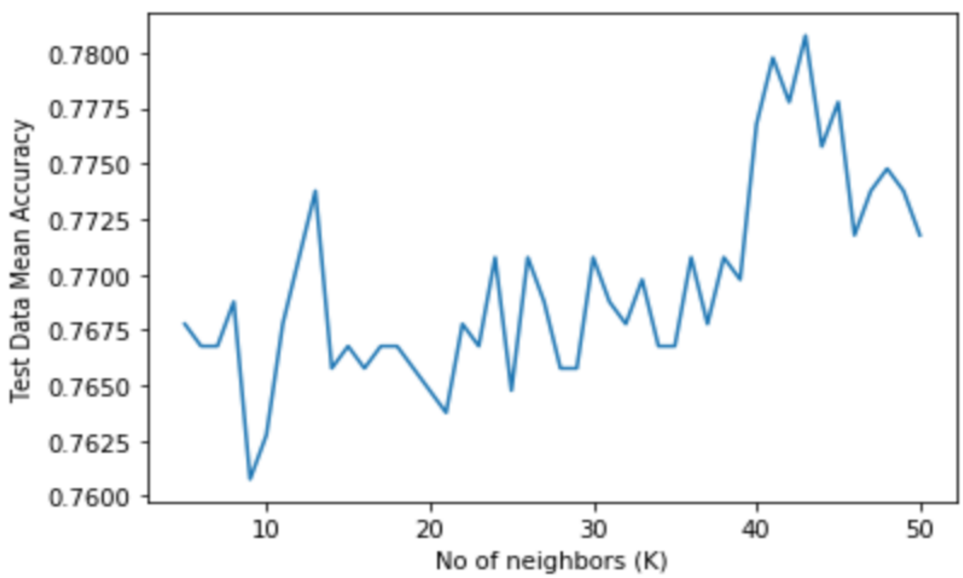

Figure 3.3 KNN plot

3.4 Decision tree

The decision tree is a supervised machine learning algorithm, which covers both classification and regression problems. The internal node on the decision tree represents the test feature, and the leaf node represents how many classes we have. Because there are six activities in our dataset, so the leaf node should be 6. Each branch of the decision tree represents a classification rule. This algorithm is easy to understand and implement. There is a non-linear relationship between parameters and it will not affect tree performance.

So how to use decision tree algorithm to solve the multi class classification problems? We know the decision trees is greedy splits on the each feature of the data at a specific threshold. This algorithm wants to find the maximum difference between the loss of the parent node and the sum of the losses of the child nodes to choose the split. In this case, the loss function I use in this algorithm is Gini Loss function.

We also need to know when to stop splitting? There are tow ways, one way of doing this is to set a minimum number of training inputs to use on each leaf and another way is to set maximum depth of my model. So the maximum depth refers to the length of the longest path from a root to a leaf. For my paper, I choose to regularize using the maximum tree depth.

CHAPTER 4. Result

4.1 Accuracy without PCA

Table 4.1 Accuracy comparison of the three algorithms

algorithms	**accuracy (%)**
DecisionTreeClassifier	78.18
KNN classifier	87.2
SVC	88.4

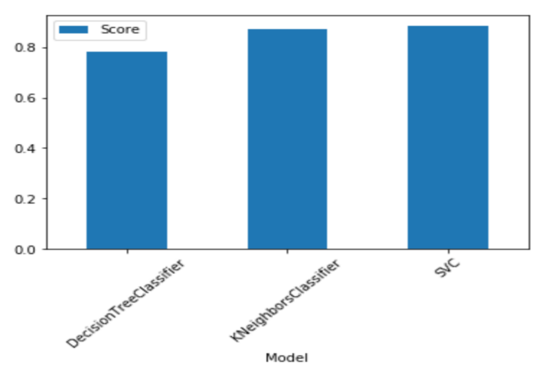

Figure 4.1 Accuracy comparison of the three algorithms

From the above table(Table 4.1) and graph(Figure 4.1), we compared the test accuracy with three algorithms(Decision tree, K-nearest neighbor and support vector machine). The result shows the support vector classifier with a linear kernel got the best result, which is 88.4% accuracy. K-nearest neighbor got almost the same accuracy (87.2%). The decision tree got the lowest accuracy, which is 78.18% accuracy. I will also use a more robust method to examine the accuracy with a confusion matrix and make sure where the errors come from.

The following graph is the confusion matrix for the decision tree(Figure 4.2). From the graph, we got the perfect result in laying and the most mis-classified activity is walking downstairs. The error rate for it is 32%, and most of the errors are identified as walking upstairs. Standing is the most mis-classified static activity, which told us the error always happened during the transition from one activity to another. It is clear, the accuracy for identify static activities are higher than dynamic activities.

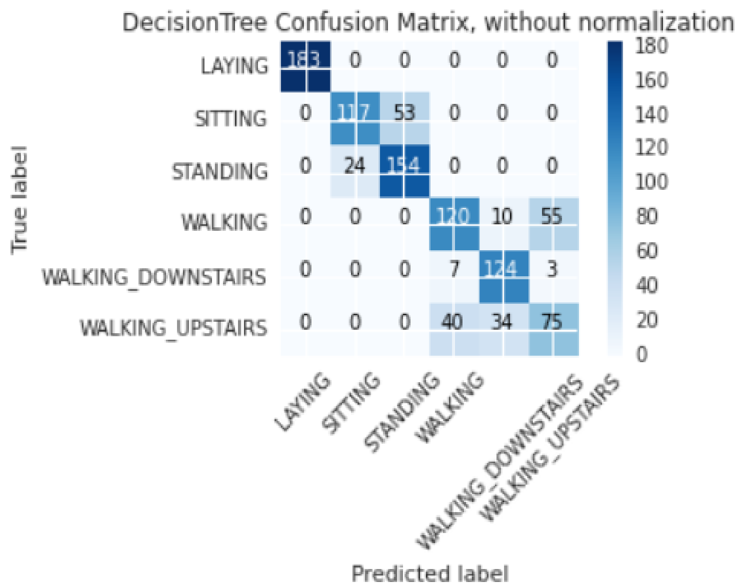

Figure 4.2 DecisionTree confusion matrix

The second confusion matrix is for support vector machine with a linear kernel(Figure 4.3). From my point of view, the HAR dataset has an imbalanced problem, and only the support vector classifier can explore deeper in it. The reason is there is build-in features in SVC and it can penalize mistakes on the minority classes. From figure SVC confusion matrix, it shows the most errors happened at standing. This activity was mis-classified as sitting and the error rate is around 36%. From this, we can also find the error was occurred monitoring transitioning activities such as sitting to standing and walking to walking upstairs.

By the way, SVC also allowed us to do Platt scaling(Figure 4.5). A separate cross-validated logistic regression will be trained to map the SVC output into probabilities of each activity. In this case, I will use the cross-validation to random sampling the dataset for this algorithm. So what I am going to use is five fold cross-validation. The test fold will start from the first fold and it will move to the next fold after the iteration. This process will stop until each fold has been used as a test fold. I will use the random forest cross-validation to train the model. From Figure 4.4, we know the optimal number of features is 159 and the processing time is 883s.

So we know there are 159 important features in full 561 features. Then I tried to plot the scatter plot to visualize the best features. The plot will be very complex if we plot all 159 features. So I will only display only five best features and find the cross-correlation.It is very clear that there is some high degree of positive correlation, and there are also some places without correlation.

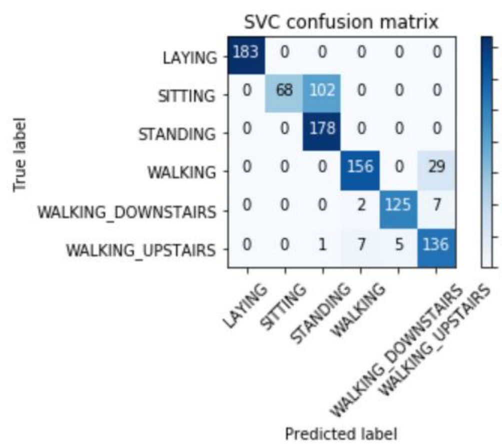

Figure 4.3 SVC confusion matrix

The last confusion matrix is for K nearest neighbor(Figure 4.6). The confusion matrix perform almost the same as SVC. The most errors also happened at standing and it was mis-classified as sitting with an error at 37%. The only different I should say is KNN take more computational time than SVC for it is a brute force method to solve the problem. This can happen because the K I choose is too small.

4.2 Accuracy and running time with PCA

As we known from the Table 4.1, KNN and SVC got the same accuracy when using the full dataset, which is around 88%. So I cannot say SVC is much better than KNN by comparing the accuracy. In this case, I will comparing them using accuracy and computational time.

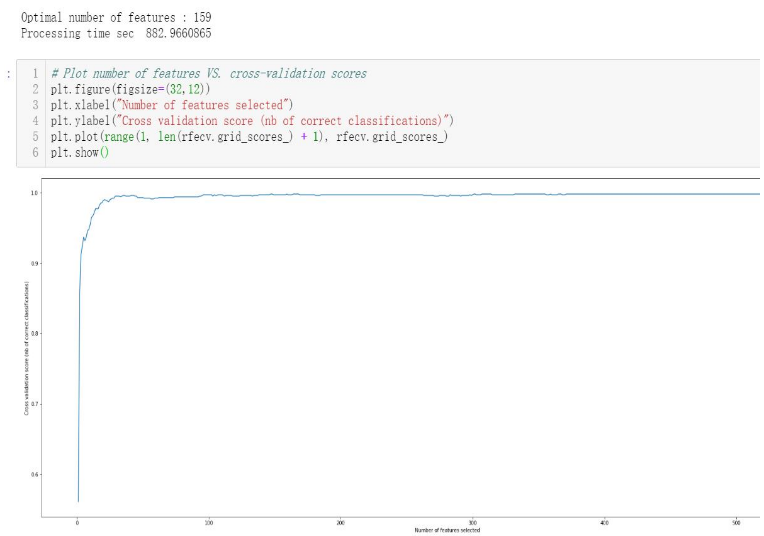

Figure 4.4 Cross validation score VS number of feature selected

Table 4.2 Accuracy and Computational time with PCA

Feature Sizes	DT	SVC	KNN
561	78.18%, 682ms	88.4%, 882ms	87.2%, 1320ms
100	69.5%, 201ms	83.2%, 312ms	86.7%, 568ms
3	52.1%, 78ms	74.4%, 83ms	77.9%, 134ms

As we can see from Table 4.2, decision trees take the least computational time compared with the other two algorithms. But the accuracy of this solution was not acceptable, which is much lower than the others. When I decreased the feature sizes to 3, I could only get 52% accuracy. From the above paragraph, we know there are two ways to solve the multi-class classification problems, which are One VS. All and One VS. One. Decision trees are very similar to OVA solution, it will take a shorter time in training than OVO solution. But the accuracy will lower than OVO. As we can see from the result, SVC is much better than decision trees.

Compared with KNN and SVC, they can get almost the same accuracy for 561 feature size. After we applied the PCA to the dataset, KNN performed 3 per cent accuracy better than SVC. From this point we know KNN perform better than SVC in low dimension data and SVC can perform better in high dimension data. If KNN works better than SVC, it indicates our dataset is not easily

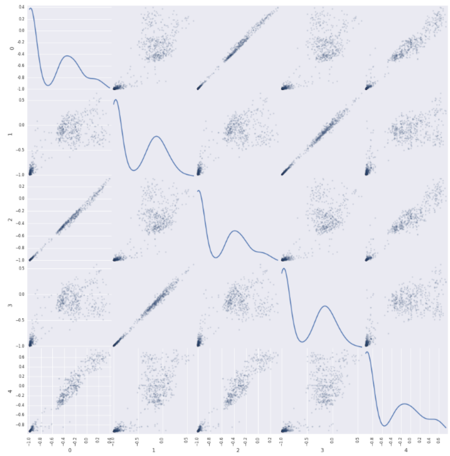

Figure 4.5 Scatter plot for SVC

separable using the decision planes that you have let SVC use. SVM is a binary classifier, which use linear hyper planes to separate classes and KNN will generate a highly convoluted decision boundary as it is driven by the training dataset. SVM will use the restricted parametric approximation of the decision boundary, which get bad performance but use small data storage space and less running time. From my result, KNN performed better than SVC in low dimension dataset, so I can say my dataset is separable in some classes. As we discussed in data exploration, we knew our dataset is separable in static activities but overfitted in dynamic activities.

After we computer the computational time for these two different algorithms, we could find the training time for SVC is much less than KNN. 638ms difference for 561 features sizes, 256ms difference for 100 feature sizes and 51ms difference for 3 feature sizes. From this point of view, I could say SVC is better than KNN to solve the HAR dataset.

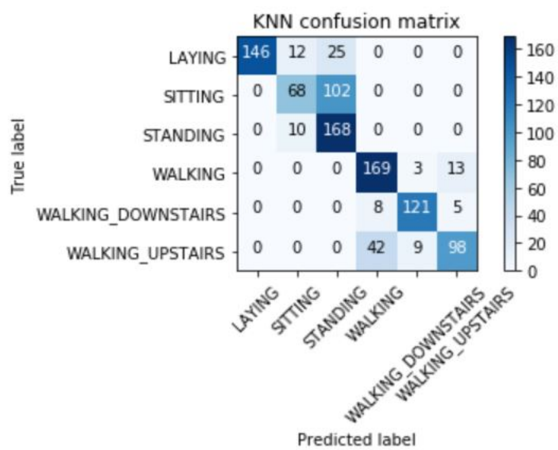

Figure 4.6 KNN confusion matrix

CHAPTER 5. Conclusion

In this paper, I used the support vector machine, K-nearest neighbor and decision trees to study human activities recognition. Compared with three algorithms, SVM gets the best accuracy (88.3%) because I use the Gaussian kernel to reduce the dataset. KNN can get the same result as SVM but it will take long time to compute. From my point of view, KNN is the least complex and I only need to compute the distance between the query and his neighbors. KNN only tests on the memory. I also try to apply the PCA algorithm on the dataset first and use KNN to solve the problem. I had reduced the feature size from 561 to 100, the accuracy did not change after that, but the training time decreased a lot. After I reduced the feature size to 3, the accuracy decreased from 87.2% to 77.9%. It decreased a lot, but it was still acceptable. After we applied PCA to SVM, we can find KNN perform better than SVM in low dimension dataset but it still needs more time to compute the result.

Future work may consider the neural network and implement the real-time system on the smartphone, which can monitor human activities as they are happening.

BIBLIOGRAPHY

[1] J. Beard, S. Biggs, D. E. Bloom, L. P. Fried, P. R. Hogan, A. Kalache, S. J. Olshansky *et al.*, "Global population ageing: peril or promise?" Program on the Global Demography of Aging, Tech. Rep., 2012.

[2] A. Yazar, F. Erden, and A. E. Cetin, "Multi-sensor ambient assisted living system for fall detection," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'14)*, 2014, pp. 1–3.

[3] M. Morris, J. Lundell, E. Dishman, and B. Needham, "New perspectives on ubiquitous computing from ethnographic study of elders with cognitive decline," in *International Conference on Ubiquitous Computing*. Springer, 2003, pp. 227–242.

[4] J. L. Reyes-Ortiz, A. Ghio, X. Parra, D. Anguita, J. Cabestany, and A. Catala, "Human activity and motion disorder recognition: towards smarter interactive cognitive environments." in *ESANN*. Citeseer, 2013.

[5] E. A. Heinz, K. S. Kunze, M. Gruber, D. Bannach, and P. Lukowicz, "Using wearable sensors for real-time recognition tasks in games of martial arts-an initial experiment," in *2006 IEEE Symposium on Computational Intelligence and Games*. IEEE, 2006, pp. 98–102.

[6] L. Wang, "Recognition of human activities using continuous autoencoders with wearable sensors," *Sensors*, vol. 16, no. 2, p. 189, 2016.

[7] N. Ikizler and P. Duygulu, "Histogram of oriented rectangles: A new pose descriptor for human action recognition," *Image and vision computing*, vol. 27, no. 10, pp. 1515–1526, 2009.

[8] O. Tunçel, K. Altun, and B. Barshan, "Classifying human leg motions with uniaxial piezoelectric gyroscopes," *Sensors*, vol. 9, no. 11, pp. 8508–8546, 2009.

[9] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019.

[10] Y. Xue and L. Jin, "A naturalistic 3d acceleration-based activity dataset & benchmark evaluations," in *2010 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2010, pp. 4081–4085.

[11] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019.

[12] Z. Chen, C. Jiang, and L. Xie, "A novel ensemble elm for human activity recognition using smartphone sensors," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 5, pp. 2691–2699, 2018.

[13] L. Fan, Z. Wang, and H. Wang, "Human activity recognition model based on decision tree," in *2013 International Conference on Advanced Cloud and Big Data*. IEEE, 2013, pp. 64–68.

[14] S. J. Preece, J. Y. Goulermas, L. P. Kenney, and D. Howard, “A comparison of feature extraction methods for the classification of dynamic activities from accelerometer data,” *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 3, pp. 871–879, 2008.

[15] M. M. Hassan, M. Z. Uddin, A. Mohamed, and A. Almogren, “A robust human activity recognition system using smartphone sensors and deep learning,” *Future Generation Computer Systems*, vol. 81, pp. 307–313, 2018.

[16] J.-L. Reyes-Ortiz, L. Oneto, A. Samà, X. Parra, and D. Anguita, “Transition-aware human activity recognition using smartphones,” *Neurocomputing*, vol. 171, pp. 754–767, 2016.

[17] N. D. Nguyen, P. H. Truong, and G.-M. Jeong, “Daily wrist activity classification using a smart band,” *Physiological measurement*, vol. 38, no. 9, p. L10, 2017.

[18] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine,” in *International workshop on ambient assisted living*. Springer, 2012, pp. 216–223.

[19] L. Bao and S. S. Intille, “Activity recognition from user-annotated acceleration data,” in *International conference on pervasive computing*. Springer, 2004, pp. 1–17.

[20] Z. He and L. Jin, “Activity recognition from acceleration data based on discrete consine transform and svm,” in *2009 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2009, pp. 5041–5044.

[21] K. H. Walse, R. V. Dharaskar, and V. M. Thakare, “Pca based optimal ann classifiers for human activity recognition using mobile sensors data,” in *Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 1*. Springer, 2016, pp. 429–436.

[22] S. Sani, N. Wiratunga, and S. Massie, “Learning deep features for knn-based human activity recognition.” in *CEUR Workshop Proceedings*, 2017.

[23] A. Mandong and U. Munir, “Smartphone based activity recognition using k-nearest neighbor algorithm,” in *Proceedings of the International Conference on Engineering Technologies, Konya, Turkey*, 2018, pp. 26–28.

[24] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang, “Convolutional neural networks for human activity recognition using mobile sensors,” in *6th International Conference on Mobile Computing, Applications and Services*. IEEE, 2014, pp. 197–205.

[25] H. Cho and S. M. Yoon, “Divide and conquer-based 1d cnn human activity recognition using test data sharpening,” *Sensors*, vol. 18, no. 4, p. 1055, 2018.

[26] A. Murad and J.-Y. Pyun, “Deep recurrent neural networks for human activity recognition,” *Sensors*, vol. 17, no. 11, p. 2556, 2017.

[27] I. Jolliffe, “Principal component analysis for time series and other non-independent data,” *Principal Component Analysis*, pp. 299–337, 2002.

[28] L. Balzano, Y. Chi, and Y. M. Lu, "Streaming pca and subspace tracking: The missing data case," *Proceedings of the IEEE*, vol. 106, no. 8, pp. 1293–1310, 2018.

[29] A. Krause, M. Ihmig, E. Rankin, D. Leong, S. Gupta, D. Siewiorek, A. Smailagic, M. Deisher, and U. Sengupta, "Trading off prediction accuracy and power consumption for context-aware wearable computing," in *Ninth IEEE International Symposium on Wearable Computers (ISWC'05)*. IEEE, 2005, pp. 20–26.

[30] A. Mannini and A. M. Sabatini, "Machine learning methods for classifying human physical activity from on-body accelerometers," *Sensors*, vol. 10, no. 2, pp. 1154–1175, 2010.