FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D COIMBRA

Pattern Recognition

Departamento de Engenharia Informática

Final Report

Smartphone and Smartwatch Activity Recognition

Francisco Miranda - n.º 2015250592
Miguel Arieiro - n.º 2014197166

Index

General Approach

In order to develop classifiers for human activity recognition we started by preparing data for analysis recurring to Matlab and the Statistical Pattern Recognition Tool (STPRtool). Using developed code, experimentation took place in order to identify the best techniques and parameters to be used on the generation of classifiers. After analysis of the best techniques, models were generated to create a multi model classification system based on majority voting. Each step of the work is further explained in the following sections.

Developed Code

Code was developed to load and normalize data from ARFF files and to parametrically execute Feature Selection, Feature Reduction and develop classifiers based on multiple techniques. In order to enable fast parameter tweaking and technique swapping, a Graphical User Interface (GUI) was also developed.

The GUI is composed of 3 tabs, each with options referring to a phase of the classifier development. On the first tab we're allowed to select which of the 3 proposed scenarios we want to test, as well as choose the classifier to be used and test how the existing models fare on a majority voting classifier. On the second tab we're allowed to check on and update the selected features. On the third tab we can select methods for feature reduction. At the time of delivery of this milestone.

On load the program will automatically load all files and organize them into structures ready for use with STPRtool and set the selected features for all those extracted from the phone accelerometer. It will also load previously stored models. In order to view the selected features update has to be pressed on the second tab. Each time feature selection parameters are changed, the update button must be pressed in order to make these changes effective.

When start button is pressed, the selected feature reduction technique will be applied over the selected features, then a classification model will be created using the selected classifier. When possible, the first 2 features will be utilized to create a 2d plot showcasing the projection of all individuals as well as the feature classification rule created into these feature dimensions. The calculated classification error will be showcased below the plot.

Feature Selection

For feature selection we created the option to choose between each of the 4 sensors. Then we can narrow down the features by removing the first 30 features which represent the values taken each second, as we understand the values isolated do not represent accurately the macroscopic activity occurring, and can be displaced in time over multiple samples.

In order to further narrow down features we allow another option. This option will use either Kruskal-Wallis or area under ROC curve method to rank and sort the features. Then we calculate the correlation matrix for all features. Finally we create a new set of selected features by adding the most discriminant one's first, and not adding those features, whose absolute correlation with at least one of the already added features, is above a certain threshold.

We're also allowed to manually select just a few of the features algorithmically selected.

Feature Reduction

Feature reduction is supported with 3 different methods, utilizing 2 techniques, Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA). All of the methods project the features selected into a smaller dimension.

Two of these methods are directly applying either LDA or PCA with a defined target dimension. The third method is to apply PCA to a calculated sub dimension which keeps a given percentage of the variance in the data.

Classifiers

We've implemented several types of classifiers, three linear and three other.
The linear classifiers consist of Euclidean Minimum Distance Classifier, Mahalanobis Minimum Distance Classifier and Fisher LDA classifier. Both minimum distance classifiers were fully implemented on the *Euclidean* and *Mahalanobis* functions, while Fisher LDA was implemented on *FisherLDA* function by using the Statistical Pattern Recognition ToolBox (STPRtool).

The remaining three classifiers are a Bayesian classifier, defined on the *Bayes* function, a K-nearest neighbour classifier (KNN) defined on the *KNN* function and a SVM classifier which is implemented on the *SVM* function in which we've used a linear kernel (SMO).

The last three classifiers were implemented by using the functions provided on the STPRtool.

Experimentation

In order to have comparable results and enough information to conclude which method was more effective on building a classifier experimentation took place with multiple variables. The variables to tweak were:

- Data set - Phone or Watch, accelerometer or gyroscope
- Classifier method - Fisher LDA, Euclidean Minimum Distance, Manhabolis Distance
- Removal of raw measurements - Boolean
- Feature ranking method - Kruskal-Wallis or AUC (area under curve)
- Feature absolute correlation threshold - Varying between 0 and 1
- Feature Reduction Method - PCA with fixed size, PCA with fixed variance, LDA
- Dimension after feature reduction - varying between 1 and the original number of features

A script was developed to test multiple combinations of the data in automated fashion. For each scenario, a total of 2112 parameter combinations were tested, combining all of the methods, data sets and feature selection options. Thresholds were tried for the values of 0.1 0.5 and 0.9, reduction techniques were tried for 2, 3, 5 and 10 dimensions, PCA was tried while keeping 50%, 75% and 95%. Tests that failed were not considered for analysis.

The script ran each combination of parameters 25 times and calculated the average error and standard deviation of the error for the created classifier. Results were then written to a text file and sorted by increasing average error for further analysis.

Results Analysis

Statistical calculations

Average error and average error deviation was calculated for each possible value for each experimental variable. This allowed us to get an overview of the parameters' contribution to the quality of the model, which values worked best and how consistent they were when used with a wide variety of other parameters. Calculated values are displayed on the following tables.

Analysis of calculated statistics

Average error and average error deviation was calculated for each possible value for each experimental variable. This allowed us to get an overview of the parameters' contribution to the quality of the model, which values worked best and how consistent they were when used with a wide variety of other parameters. Calculated values for each scenario are displayed in tables 1, 2 and 3 at the end of this document.

Scenario A

For this scenario the most consistent data came from the Smartphone Accelerometer's dataset. Removing the first 30 features, keeping a low correlation threshold, ranking features with KruskalWallis and using pca reduction with parameter 0.95 all led to better results. The best classifiers were KNN and Bayes, largely surpassing other classifiers.

Scenario B

For this scenario the most consistent data came from the Smartwach Accelerometer's dataset. Removing the first 30 features, keeping the correlation threshold at 0.9, PCA reduction with parameter 0.95 and KNN or Bayesian models all led to better results.

Scenario C

For this scenario the most consistent data came from the Smartwach Accelerometer's dataset. Keeping the most features and KNN classifier led to overall better results. Due to the high number of features Bayes model creation and LDA feature reduction were unfeasible.

Model choice

Scenario A

Being the simplest scenario we could easily achieve an error below 1% with many different models. In the majority voting system, with just 3 models, we were able to achieve an error of **0.09%**.

The chosen models were from the Smartwatch accelerometer's dataset. The parameters of the chosen models were as follows:

Model	Feature selection	Feature Reduction	Classifier	Average Error
1	Removed first 30 features	PCA keeping 95% variance	KNN	0.33%
2	Removed first 30 features and kept correlation threshold at 0.5, ranking features with KruskalWallis	PCA keeping 95% variance	KNN	0.36%
3	All features	LDA to 3- dimensional space	Bayes	4.0%

Below can be found the confusion matrix for the model generated in this scenario.

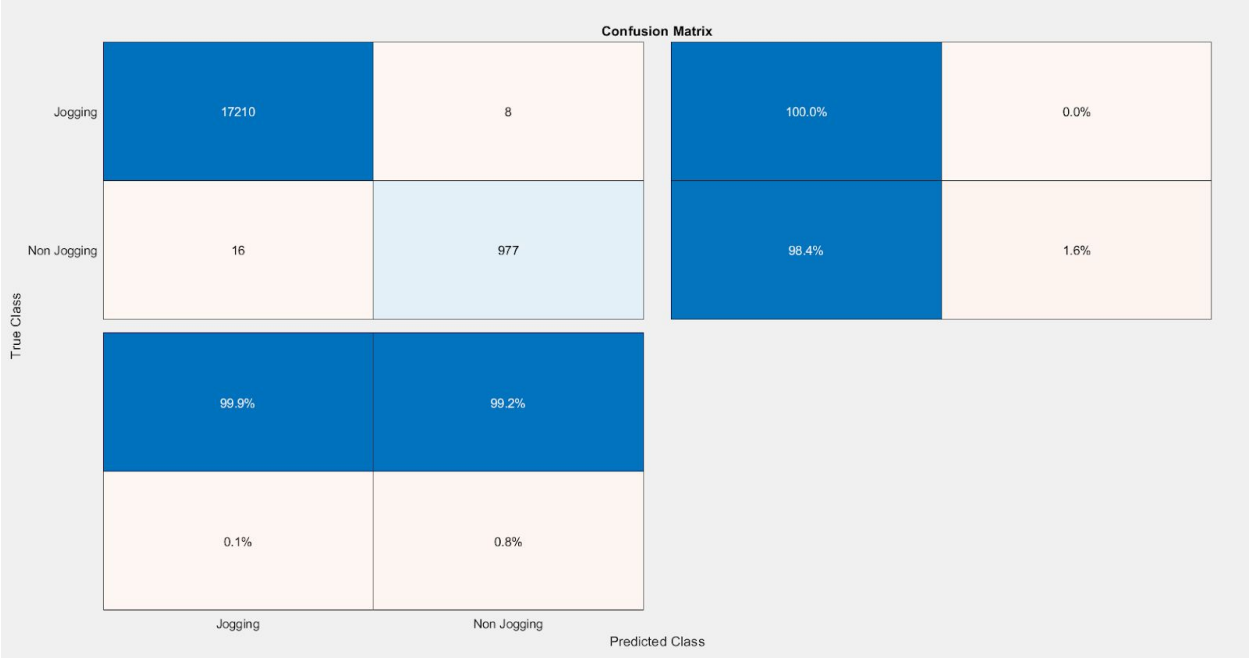

Figure 1: Confusion Matrix for Scenario A

Scenario B

This scenario led to much worse errors than on the previous one, best models were created with the KNN method, and while overall Bayes performance was close to that of KNN, it didn't provide any particularly reliable model. Best model had an error of 7.4%, but with just 4 models we were able to achieve an error of **4.9%**.

The chosen models were from the Smartwatch accelerometer's dataset and the classifier used was always KNN. The parameters of the chosen models were as follows:

Model	Feature selection	Feature Reduction	Average Error
1	All features	PCA keeping 95% variance	7.4%
2	Removed first 30 features and kept correlation threshold at 0.9, ranking features with KruskalWallis	PCA keeping 95% variance	8.3%
3	Removed first 30 features and kept correlation threshold at 0.9, ranking features with AUC	PCA keeping 95% variance	8.3%
4	All features	PCA to 10-dimensional space	9.2%

Below can be found the confusion matrix for the model generated in this scenario.

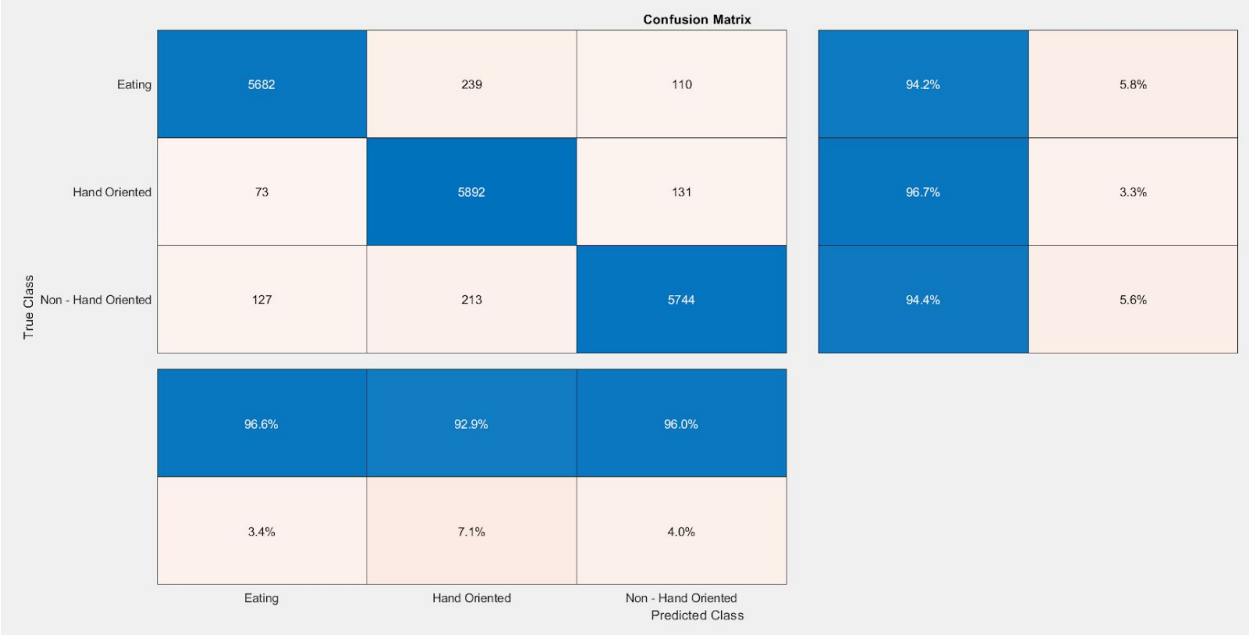

Figure 2: Confusion Matrix for Scenario B

Scenario C

This scenario was by far the most complex one and as such had a much higher error rate. As such the best individual model had an error of 24.7%, after using 6 different models and deciding by majority voting we were able to attain only **20.7%** error. The chosen models were from the Smartwatch accelerometer's dataset, the classifier used was always KNN and reduction always was done by PCA keeping 95% variance. The parameters of the chosen models were as follows:

Model	Feature Selection	Average Error
1	Kept all features	24.7%
2	Removed first 30 features, kept correlation threshold at 0.9, ranked features with KruskalWallis	26.7%
3	Removed first 30 features, kept correlation threshold at 0.9, ranked features with AUC	26.3%
4	Removed first 30 features	26.7%
5	kept correlation threshold at 0.5, ranked features with KruskalWallis	28.7%
6	kept correlation threshold at 0.5, ranked features with AUC	29.1%

Below can be found the confusion matrix for the model generated in this scenario. It is observable that the hardest to identify activity was "Eating a sandwich" presenting a 35.0% specificity, with all other activities presentir a sensibility and specificity above 64.7%.

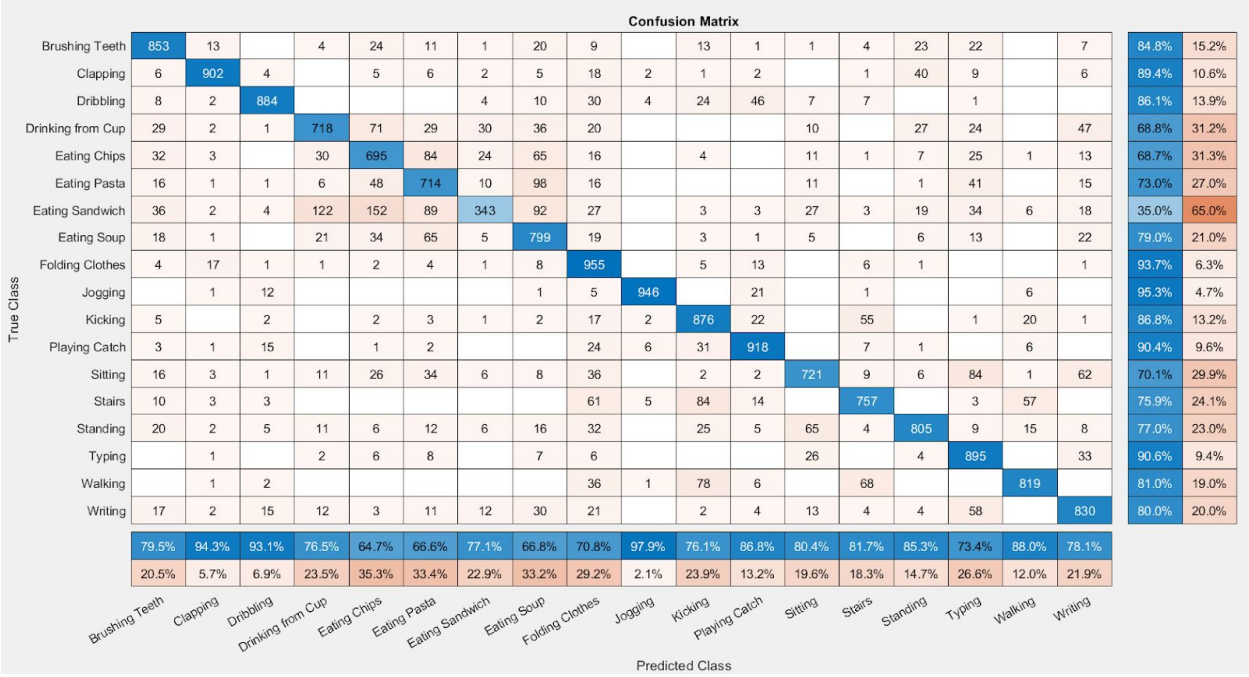

Figure 3: Confusion Matrix for Scenario C

Conclusion

Once our study was complete we decided to look for other works which had used the same data set for comparison purposes. In the study referenced in [2] we found that impersonal models were created, matching our work.

For the watch accelerometer, in the scenario C, the models had an accuracy varying from 59.3% to 70.3%, which was worse than that of our final model (79.3%). Although we had slightly better results, we agreed that the most difficult to recognize activity was “Eating a sandwich” and that the best data set was from the watch accelerometer.

Due to lack of processing power other methods such as support-vector machines were not explored, however given comparison to the study previously referenced, we were satisfied with our results.

Tables

Table 1: Parameter Quality for Scenario A

data set	avg error	std error
1	16.15%	27.59%
2	21.18%	28.39%
3	18.32%	27.56%
4	22.56%	30.29%
remove first	**avg error**	**std error**
0	20.25%	28.97%
30	18.83%	28.16%
correlation threshold	**avg error**	**std error**
0.1	16.80%	23.41%
0.5	19.03%	29.59%
0.9	21.20%	32.56%
1	20.98%	27.71%
feature selection method	**avg error**	**std error**
ROC	21.38%	28.94%
KruskalWallis	17.64%	28.06%
feature reduction method	**avg error**	**std error**
pcaReduceFeatures	8.99%	7.48%
ldaReduceFeatures	39.91%	40.72%
feature reduction parameter	**avg error**	**std error**
0.95	7.56%	6.80%
0.75	8.48%	7.08%
0.5	9.76%	7.65%
2	17.61%	25.26%
3	22.51%	30.90%
5	26.61%	34.99%
10	28.67%	36.68%
classifier	**avg error**	**std error**
KNN	3.76%	2.47%
Bayes	4.07%	1.55%
Euclidean	30.74%	32.07%
Mahalanobis	31.77%	34.32%

Table 2: Parameter Quality for Scenario B

data set	avg error	std error
1	41.25%	16.20%
2	47.86%	11.53%
3	36.58%	16.99%
4	46.21%	14.24%
remove first	**avg error**	**std error**
0	44.25%	16.54%
30	41.68%	14.49%
correlation threshold	**avg error**	**std error**
0.1	42.67%	9.35%
0.5	43.65%	16.83%
0.9	41.93%	17.41%
1	43.38%	17.11%
feature selection method	**avg error**	**std error**
ROC	43.13%	15.78%
KruskalWallis	42.68%	15.33%
feature reduction method	**avg error**	**std error**
pcaReduceFeatures	38.97%	10.29%
ldaReduceFeatures	50.47%	20.41%
feature reduction parameter	**avg error**	**std error**
0.95	34.22%	11.34%
0.75	37.29%	10.71%
0.5	42.37%	9.44%
2	44.48%	10.10%
3	43.75%	14.63%
5	45.17%	18.82%
10	45.59%	21.41%
classifier	**avg error**	**std error**
KNN	33.53%	13.29%
Bayes	36.26%	7.14%
Euclidean	49.66%	13.64%
Mahalanobis	49.73%	16.63%

Table 3: Parameter Quality for Scenario C

data set	avg error	std error
1	74.95%	20.22%
2	83.83%	10.47%
3	70.72%	21.73%
4	76.56%	15.74%
remove first	**avg error**	**std error**
0	75.89%	18.85%
30	77.14%	17.52%
correlation threshold	**avg error**	**std error**
0.1	81.99%	11.58%
0.5	76.80%	18.30%
0.9	73.29%	20.29%
1	73.99%	20.02%
feature selection method	**avg error**	**std error**
ROC	76.68%	18.22%
KruskalWallis	76.35%	18.19%
feature reduction method	**avg error**	**std error**
pcaReduceFeatures	76.51%	18.20%
ldaReduceFeatures	#N/A	#N/A
feature reduction parameter	**avg error**	**std error**
0.95	71.53%	21.91%
0.75	74.56%	19.61%
0.5	79.78%	15.43%
2	82.64%	10.94%
3	79.02%	14.83%
5	75.45%	18.51%
10	72.62%	21.10%
classifier	**avg error**	**std error**
KNN	57.95%	16.22%
Bayes	#N/A	#N/A
Euclidean	77.10%	7.90%
Mahalanobis	94.50%	0.06%

References

[1] WISDM Smartphone and Smartwatch Activity and Biometrics Dataset

[2] G.M. Weiss, J.L. Timko, C.M. Gallagher, K. Yoneda, and A.J. Schreiber (2016). Smartwatch-based Activity Recognition: A Machine Learning Approach, Proceedings of the 2016 IEEE International Conference on Biomedical and Health Informatics (BHI 2016), Las Vegas, NV, 426-429