DATA MINING

Project Report

Bremen Big Data Challenge - Edition 2019

By Gari Ciodaro, Diogo Cosin, and Ralph Florent

May 17, 2019

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Laoreet non curabitur gravida arcu ac. Et molestie ac feugiat sed lectus vestibulum mattis ullamcorper velit. Turpis egestas pretium aenean pharetra. Amet justo diam vulputate ut pharetra sit. Elit scelerisque mauris pellentesque pulvinar. Mauris in aliquam sem fringilla. Id ornare arcu odio ut sem nulla. Nunc lobortis mattis aliquam faucibus purus. Eget velit aliquet sagittis id consectetur purus ut faucibus pulvinar. Nulla aliquet enim tortor auctor urna nunc id cursus. Varius quam quisque id diam vel quam elementum pulvinar.

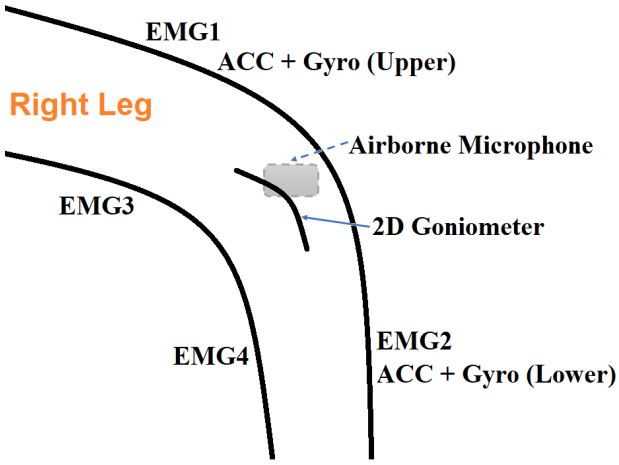

Figure 1: Wearable sensor placement for data measurements. (source: part of the provided data)

1 Introduction

2 Background of the Data

2.1 Data Source

Provided by *"The Bremen Big Data Challenge 2019" Organizers*, the collected data are based on daily athletic movements [**bbdc**]. Using wearable sensors above and below the knee (See Figure 1) of the individual (athletic), a dataset of 19 individuals, mainly identified as *subjects*, has been recorded. And as the competition requires, the data of 15 out of the total number of subjects are used as the training dataset and the remaining part as the testing dataset. The dataset is publicly available online on the official website: BBDC or by simply browsing through the following URL: *https://bbdc.csl.uni-bremen.de/index.php/2019h/28-aufgabenstellung-2019*.

2.2 Description and Format

The data comprise the following 22 movements:

- Race ('run')
- Walking ('walk')
- Standing (standing)
- Sitting ('sit')
- Get up and sit down ('sit-to-stand', 'stand-to-sit')
- Up and down stairs ('stair-up', 'stair-down')
- Jump on one or both legs ('jump-one-leg', 'jump-two-leg')
- Run left or right ('curve-left-step', 'curve-right-step')
- Turn left or right on the spot, left or right foot first ('curve-left-spin-Lfirst', 'curve-left-spin-Rfirst', 'curve-right-spin-Lfirst', 'curve-right- spin-Rfirst ')
- Lateral steps to the left or right ('lateral-shuffle-left', 'lateral-shuffle-right')

Subjects	Datafile	Label
Subject02	`Subject02/Subject02_Aufnahme000.csv`	*curve-left-step*
Subject02	`Subject02/Subject02_Aufnahme001.csv`	*curve-left-step*
Subject02	`Subject02/Subject02_Aufnahme002.csv`	*stand-to-sit*
...	...	...
Subject19	`Subject19/Subject19_Aufnahme438.csv`	*curve-right-step*
Subject19	`Subject19/Subject19_Aufnahme439.csv`	*curve-right-spin-Rfirst*

Table 1: Tabular visualization of the "***train.csv***" dataset

Subjects	Datafile	Label
Subject01	`Subject01/Subject01_Aufnahme000.csv`	*X*
Subject01	`Subject01/Subject01_Aufnahme001.csv`	*X*
Subject01	`Subject01/Subject01_Aufnahme002.csv`	*X*
...	...	...
Subject15	`Subject15/Subject15_Aufnahme438.csv`	*X*
Subject15	`Subject15/Subject15_Aufnahme439.csv`	*X*

Table 2: Tabular visualization of the "***challenge.csv***" dataset

- Change of direction when running to the right or left, left or right foot first ('v-cut-left-left', 'v-cut-left-right', 'v-cut-right-left', 'v-cut' right-Rfirst ')

The entire data are available as CSV files, or Comma-Separated Values, and partitioned as training and testing data, respectively represented by the "***train.csv***" and the "***challenge.csv***" files. Starting with the training dataset file (***train.csv***), it contains *UTF-8*[1] character-encoded, line-wise plain texts, whose first line identifies the feature names followed by the feature values. This file contains a total of 6402 lines, which include both the feature names and the feature values. The feature names are *Subjects*, *Datafile*, *Label*, and the feature values map respectively each feature name. For instance, the first feature values of the file are: *Subject02*, `Subject02/Subject02_Aufnahme002.csv`, *stand-to-sit*. Table 1 illustrates a lightweight version of the data partition of the training dataset file.

Similarly, the testing dataset file (***challenge.csv***) is formatted using the same structure with the exception of the *Label* column, which is unknown and marked with an *X*. The datafile contains a total of 1739 lines counting both the feature names and feature values. Table 2 displays a lightweight version of the data partition of the testing dataset file.

Important: Recalling that the dataset is divided into training and testing data, the subjects "*Subject01, Subject10, Subject14, Subject15*" are the selected ones that are used as testing data to assess the solutions. Note the difference in the starting and ending rows of Tables 1 and 2.

As observed in both Tables 1 and 2, each line corresponds to a recording of a movement. The columns have the following meanings:

- ***Subject***: The ID of the subject
- ***Datafile***: Path of the file containing the sensor data for this recording. For each subject, there is a folder in which individual data files contain the sensor data for individual motion recordings.
- ***Label***: The movement that was recorded

Particularly, the *Label* column of the testing dataset contains repeatedly the letter "*X*" to indicate that this value is not present. That is, at the time of submitting solutions, the submission should

[1]Unicode Transformation Format, extended ASCII, variable-width encoding.

exactly match the testing data, where each X will be replaced by a label. This label corresponds to the classification result of a specific movement. It is important that the spelling (including upper / lower case) of the textual labels matches exactly the spelling of the labels in the training data.

As mentioned above, the datafiles are references to other CSV files. For example, the path file `Subject02/Subject02_Aufnahme000.csv` is a CSV file itself within a folder named *Subject02* located in the root path (i.e., the current directory of the downloaded zip files). The CSV file itself is dataset with a proper format. Basically, the file has a set of comma-separated numbered-values that looks like this: *32688,32224,32991,32609,32790,33048,37168,34610,27374,29068,29264, 28408,31784,28133,29295,29244,33216,37140,34736*.

Each line represents the sensor values measured at one time (sampled at 1000 Hz). The columns represent the individual wearable sensors recording the human activities (see Figure 1):

1. EMG1
2. EMG2
3. EMG3
4. EMG4
5. Airborne
6. ACC upper X
7. ACC upper Y
8. ACC upper Z
9. Goniometer X
10. ACC lower X
11. ACC lower Y
12. ACC lower Z
13. Goniometer Y
14. Gyro upper X
15. Gyro upper Y
16. Gyro upper Z
17. Gyro lower X
18. Gyro lower Y
19. Gyro lower Z

The size of the CSV datafiles vary inappropriately. That is, in most cases, due to inaccurate measurements, random initialization states, mechanical flaws, computational and processing cost, and so on.

3 General comments

3.1 Notation

Let us defined a modeling procedure as a function $\mathcal{P}(\Theta) : \mathbb{R}^m \longrightarrow \mathbb{R}^k$ where m is the number of features, k is the number of classes, and $\Theta : \{Preprocessing_{method}, Feature_{extration}, Statistical_{Tecnique}\}$ is a set representing parameters. Giving this abstraction Θ modifies the structure $\mathcal{P}$ but always takes a *feature vector* $X \in \mathbb{R}^m$, and returns a *Probability vector* $Y \in \mathbb{R}^k$ where each component $\in [0, 1]$.

It is clear that the $\mathcal{P}(\Theta)$ that represents exactly the reality regarding athletics movements is unknown to us, which let us to defined a measure of the amount of veracity that a giving $\mathcal{P}$ has compare to reality.

$$accuracy = \frac{\sum Correct_{classification}}{N} \tag{1}$$

3.2 Cross validation

Guys please explain here what cv is, why is important to use, that is what is the shit with variance and bias, why overfit may occurs. In general terms. why is a good stimator of the test error over unseen data.

3.3 Curse of dimensionality

Guys please explain this also. remember to say the 10 feature rule thumbs and cited jaeger lectures notes.

4 Data Preprocessing

Two *Preprocessing methods* procedures were implemented.

Preprocessing method 1:

1. Take a **subject file**(each file contains a class movement). It can be viewed as $[19 \times N_{fr}]$ matrix composed by 19 columns vectors $S_{data} \in \mathbb{R}^{N_{fr} \times 1}$ where number of records in file is $N_{fr} \in \mathbb{N}^{>0}$.
2. Transpose each S_{data} into a row vector, concatenating them into one single vector $S_{concat} \in \mathbb{R}^{1 \times 19 * N_{fr}}$
3. Repeat step 1 and 2 for every subject file.
4. Create a dataset with the rows of step 3 with it according it corresponding label (extracted from the name of the file). This data set is a matrix D of dimensions $[6401 \times (19 * N_{fr})]$. For $N_{fr} = 56810$ D is $[6401 \times 1079390 + 1]$.

Preprocessing method 2:

1. Take a **subject file**(each file contains a class movement). It can be viewed as $[19 \times N_{fr}]$ matrix composed by 19 columns vectors $S_{data} \in \mathbb{R}^{N_{fr} \times 1}$ where number of records in file is $N_{fr} \in \mathbb{N}^{>0}$.
2. For each S_{data} decomposed it as $s_{data} = \mu + \omega$ where μ is a smoothed version of s_{data} calculated using lowess with tree points average weighted linear regression. A complete derivation of this algorithm can be found in [**1**]. Having s_{data} and μ calculate $\omega = s_{data} - \mu$. Create a vector $\mu_{sta} \in \mathbb{R}^3$ with first tree statistical moments of μ, that is, the average, the variance, and the steepness. calculate the discrete Fourier transformation of ω, extract the first five coefficients and put them in a vector $\omega_{fft} \in \mathbb{R}^5$. A complete derivation of this algorithms can be found in [**2**]. take μ_{sta} and concatenate it with ω_{fft} into a row vector $S \in \mathbb{R}^8$

3. concatenate each S into a single vector $S_{row} \in \mathbb{R}^{8*19}$

4. Repeat steps 1 to 3 for every **subject file** and stack the vectors S with its corresponding label into a matrix D is $[6401 \times 8 * 19]$.

For $Preprocessing_{\ method\ 1}$ the parameter N_{fr} had to be set, since each **subject file** had different number records. By observing a 100 random sample of files, we concluded rather arbitrarily that $N_{fr} = 56810$ was reasonable number of records. In case a particular file did not meet this requirement, the signal per sensor would repeat itself until the desired N_{fr} was reached. The idea with $Preprocessing_{\ method\ 2}$ was to capture the general properties of the movement(using μ_{sta}) in terms of its statistical moments. On the other hand, we also attempted to capture information about the periodicity of the movement using ω_{fft}.

5 Data Exploitation

Depending on the $Preprocessing_{method}$ some $Feature_{extration}$ and $Statistical_{Tecnique}$ combination would more adequate to implement than other. Here we only present highest *accuracy* combinations.

5.1 Data Exploitation with $Preprocessing_{\ method\ 1}$

One problem that is immediately observed with $Preprocessing_{\ method\ 1}$ is the immense dimensionality of the sample space, therefore trying to find a $Statistical_{Tecnique}$ capable of learning an adequate decision function is basically impossible. To bypass this problem implemented a $Feature_{extration}$ and $Statistical_{Tecnique}$ shown in the Figure 2.

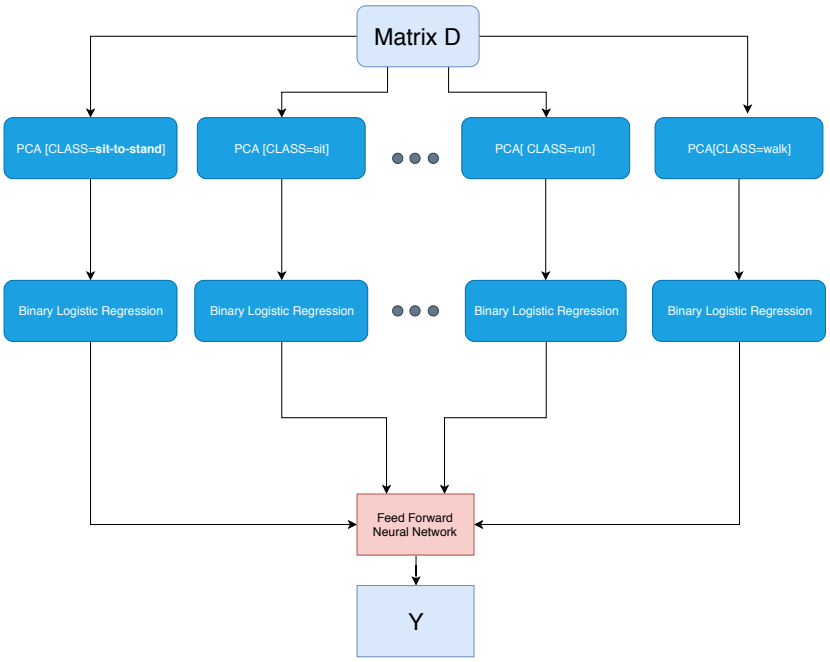

Figure 2: $\mathcal{P}(\Theta_1)$ Scheme

We theorized that the importance of features in matrix D measured by its variance will strongly depend on the particular movement involved, acknowledging this we filtered matrix D by class, and applied principal component analysis $PCA_{|class} : \mathbb{R}^{1079390} \longrightarrow \mathbb{R}^{275}$ to extract the first 275 principal components(this transformation will be the beginning a branch in Figure 2) that accounts to 90 %

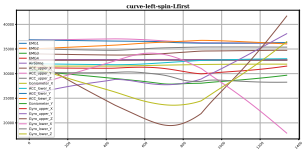

Figure 3: curve-left-spin-Lfirst

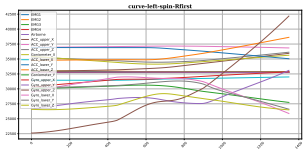

Figure 4: curve-left-spin-Rfirst

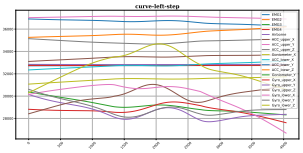

Figure 5: curve-left-step

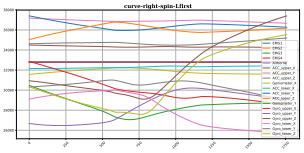

Figure 6: curRight-spin-Lfirst

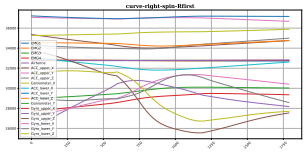

Figure 7: curRight-spin-Rfirst

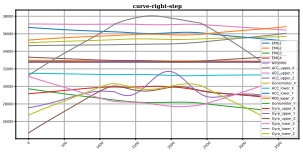

Figure 8: curRight-step

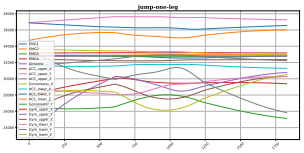

Figure 9: jump-one-leg

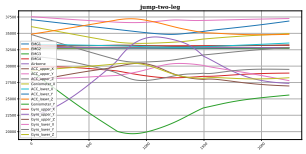

Figure 10: jump-two-leg

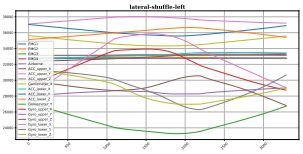

Figure 11: lateral-shuffle-left

of the variance. Note that the first level in 2 has 22 $PCA_{|class}$ extractors. For training, matrix D is passed without filtering per label to each $PCA_{|class}$ and then feed to a *Binary logistic regression* $LogR_{|class} : \mathbb{R}^{275} \longrightarrow [0, 1]$ where the classes are encoded in *one vs all* manner. Each of the $LogR_{|class}$ will output a degree of believe that a particular X belows to a class k. Finally concatenating the degree of believe predicted by each $LogR_{|class}$ we construct a $X' \in \mathbb{R}^{22}$. Finally X' is passed to a single feed forward neuronal network $NN : \mathbb{R}^{22} \longrightarrow \mathbb{R}^{22}$ that predicts our final probability vector Y in a *hot encoded* structure, we take the maximum component and assign its corresponding label to that register.

$$\Theta_1 := \{Preprocessing\ _{method\ 1}, PCA_{|class}, LogR_{|class}, NN\} \tag{2}$$

5.2 Data Exploitation with $Preprocessing\ _{method\ 2}$

The dimension of X using $Preprocessing\ _{method\ 2}$ is R^{152} so no further dimension reduction is required. We implemented a variety of feed forward neuron networks using tensor flow, where we varied the Network architecture composed by the number of hidden units $H_{units} \in \mathbb{N}^{>0}$ per hidden layer $H_L \in \mathbb{N}^{>0}$ the regularization parameter $\alpha \in \mathbb{R}^{>0}$, and the activation functions, namely $\{Logistic, tanh, relu\}$. From Figures 3 to 25 we can observe μ signal examples per label.

$$\Theta_2 := \{Preprocessing\ _{method\ 2}, NN\} \tag{3}$$

6 Data Analysis

Ralph i think it would be good if you explain here what you do when you implemented the "fully connected neuron network, like explain the parameters optimization and so on". that implementation will account for the neuron network at the end of the diagram in figure 2, report here also the score you

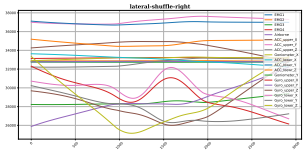

Figure 12: lateral-shuffle-right

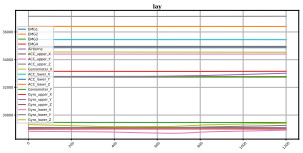

Figure 13: lay

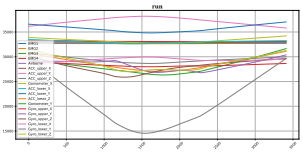

Figure 14: run

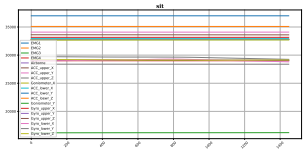

Figure 15: sit

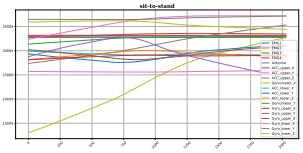

Figure 16: sit-to-stand

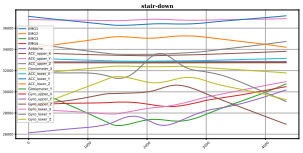

Figure 17: stair-down

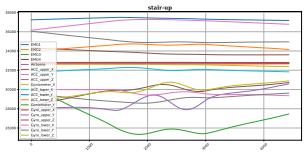

Figure 18: stair-up

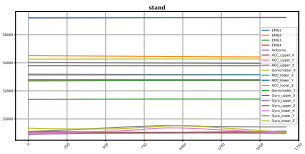

Figure 19: stand

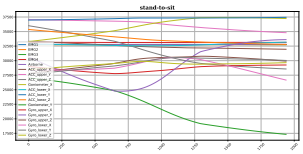

Figure 20: stand-to-sit

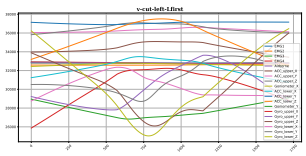

Figure 21: v-cut-left-Lfirst

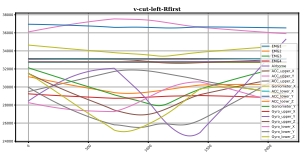

Figure 22: v-cut-left-Rfirst

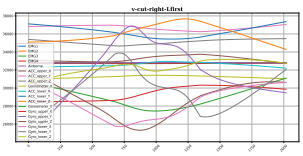

Figure 23: v-cut-right-Lfirst

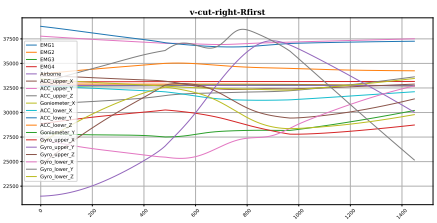

Figure 24: v-cut-right-Rfirst

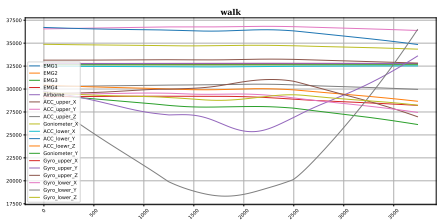

Figure 25: walk

got. By submitting both approaches to the challenge we observed that $\mathcal{P}(\Theta_2)$ outperformed $\mathcal{P}(\Theta_1)$ so for this and the following sections we will focus our attention on $\mathcal{P}(\Theta_1)$

7 Results and Discussions

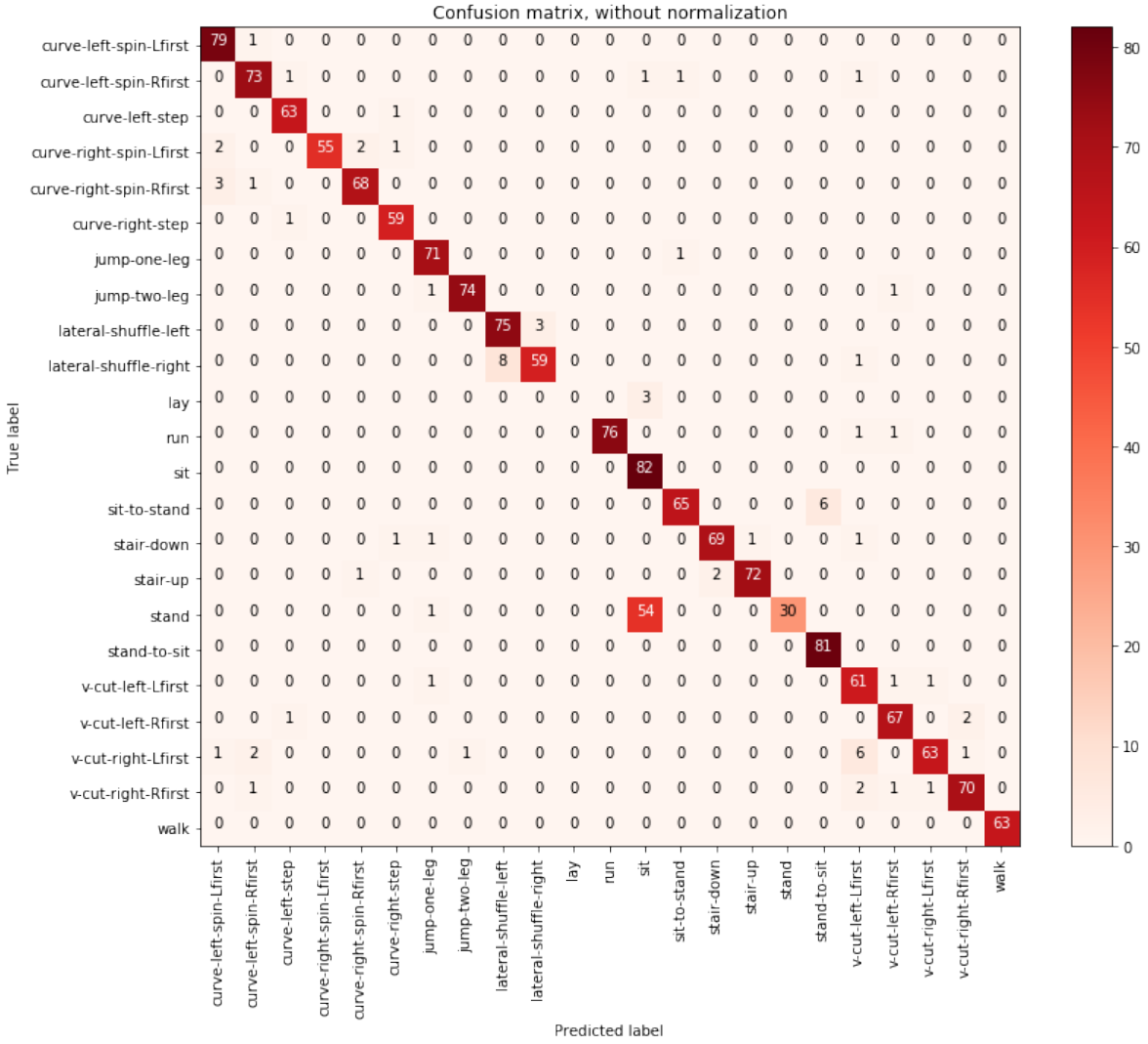

Figure 26: $\mathcal{P}(\Theta_2)$ Confusion matrix on test data

8 Conclusion

References

[1] Cleveland, W.S. (1979) "Robust Locally Weighted Regression and Smoothing Scatterplots". Journal of the American Statistical Association 74 (368): 829-836.

[2] Oraintara, S., Chen, Y. J., & Nguyen, T. Q. (2002). Integer fast Fourier transform. IEEE Transactions on Signal Processing, 50(3), 607-618.