

Article

# A Comprehensive Analysis on Wearable Acceleration Sensors in Human Activity Recognition

Majid Janidarmian <sup>\*</sup>, Atena Roshan Fekr, Katarzyna Radecka and Zeljko Zilic

Electrical and Computer Engineering Department, McGill University, Montréal, QC H3A 0E9, Canada; atena.roshanfekr@mail.mcgill.ca (A.R.F.); katarzyna.radecka@mcgill.ca (K.R.); zeljko.zilic@mcgill.ca (Z.Z.)

\* Correspondence: majid.janidarmian@mail.mcgill.ca; Tel.: +1-514-398-3937

Academic Editor: Stephane Evoy

Received: 15 November 2016; Accepted: 28 February 2017; Published: 7 March 2017

**Abstract:** Sensor-based motion recognition integrates the emerging area of wearable sensors with novel machine learning techniques to make sense of low-level sensor data and provide rich contextual information in a real-life application. Although Human Activity Recognition (HAR) problem has been drawing the attention of researchers, it is still a subject of much debate due to the diverse nature of human activities and their tracking methods. Finding the best predictive model in this problem while considering different sources of heterogeneities can be very difficult to analyze theoretically, which stresses the need of an experimental study. Therefore, in this paper, we first create the most complete dataset, focusing on accelerometer sensors, with various sources of heterogeneities. We then conduct an extensive analysis on feature representations and classification techniques (the most comprehensive comparison yet with 293 classifiers) for activity recognition. Principal component analysis is applied to reduce the feature vector dimension while keeping essential information. The average classification accuracy of eight sensor positions is reported to be  $96.44\% \pm 1.62\%$  with 10-fold evaluation, whereas accuracy of  $79.92\% \pm 9.68\%$  is reached in the subject-independent evaluation. This study presents significant evidence that we can build predictive models for HAR problem under more realistic conditions, and still achieve highly accurate results.

**Keywords:** human activity recognition; machine learning; supervised classification; wearable sensors; sensors heterogeneities

---

## 1. Introduction

The maturity of pervasive sensing, wireless technology, and data processing techniques enables us to provide an effective solution for continuous monitoring and promote individual's health. Today, the miniature sensors can be unobtrusively attached to the body or can be part of clothing items to observe people's lifestyle and behavior changes [1]. According to study presented in [2], on-body sensing proves to be the most prevalent monitoring technology for the gait assessment, fall detection and activity recognition/classification. As such, extensive research has been undertaken to select or develop reasoning algorithms to infer activities from the wearable sensor data. Human activity recognition that targets the automatic detection of people activities, is one of the most promising research topics in different areas such as ubiquitous computing and ambient assistive living [3]. Low-cost, yet highly reliable accelerometer is the most broadly used wearable sensor for the sake of activity recognition and could provide high classification accuracy of 92.25% [4], 96% [5], and 99.4% [6]. 3-D accelerations can be represented as:

$$\vec{A} = \frac{d\vec{v}}{dt} = \left( \vec{g} + \vec{l} \right), \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} = \begin{pmatrix} g_x + l_x \\ g_y + l_y \\ g_z + l_z \end{pmatrix}$$

where  $\vec{A}$  (acceleration),  $\vec{g}$  (acceleration due to gravity) and  $\vec{l}$  (applied linear acceleration) are measured in  $\frac{m}{s^2}$ . There is a large amount of work on the use of sensing for activity monitoring and behavior profiling. For example, there are surveys [7–9] that provide an outline of relevant research and applicable techniques. In the real-life scenarios, the performance of a recognition system is often significantly lower than in reported research results. Igual et al. recently applied two different fall detection algorithms to show that the performances of the fall techniques are adversely affected with cross-dataset evaluation [10]. It shows that the performance of a fall detector reduced when it is tested on a dataset different from the one used for training. This is because there exist variations in training and testing device hardware, sensor models, and their operating system characteristics among others [11]. The performance of recognition models mainly depends on the activity set, training data quality, extracted features, and learning algorithms. Since each Machine Learning (ML) model in the literature was trained with a specific dataset and activity set, there is no significant evidence to claim that any predictive model is more precise than the others. In other words, the classification model is built based on the collected samples under specific conditions involving sensor type, position and orientation of sensors on the human body, sampling rate, and activity performance style. Therefore, the trained model may not be directly applied to other related datasets and may fail to understand the pattern, if there is any change in the sensor characteristics, data acquisition scenarios or the users (concerning, e.g., age, weight or physical fitness). For instance, different accelerometer sensors often suffer from various biases and thus differ in precision and resolution. This issue, combined with sampling rate instability of each device introduces major challenges for the HAR system design [12]. The difference in the styles of performance of an activity also poses some challenges for application developers and researchers. Stisen et al. [11] showed that even OS type and CPU load have also drastic negative effects on recognition accuracy. Therefore, we aim to comprehensively evaluate the machine learning algorithms to extend the applicability of the trained model dealing with diverse accelerometer measurements. In this work, we have aggregated 14 well-known benchmark datasets that are publicly available to the research community and, in each dataset, data have been collected with different devices (commercial mass-marketed or research-only devices), acquisition protocols (under naturalistic circumstances or laboratory environments), participants, sensors placements, models and biases, motion artifacts and sampling rate heterogeneities to have a big realistic dataset. Considering these challenges makes it difficult to obtain a robust activity recognizer that is invariant to biases and performs well on unseen datasets. This is the first time to the best of our knowledge that such rigorous activity recognition evaluation at a large-scale on ML techniques is investigated. This study will explain the pros and cons of variety of learning methods and will speed up implementation of robust recognition algorithms using wearable accelerometer sensors. We report the effects of heterogeneities on various classifiers considering two cross-validation techniques. K-fold ( $k = 10$ ) is the most widely accepted methodology to compute the accuracy of a developed model in HAR problem [13]. In this technique, the model is trained using  $k - 1$  of the folds as training data and the obtained model is validated on the remaining part of the data to compute the accuracy or other performance metrics. However, to explore the limitations of finding a personalization approach caused by large variance in per-user accuracy, a subject-independent cross-validation technique, Leave-One-Subject-Out (LOSO), is also considered.

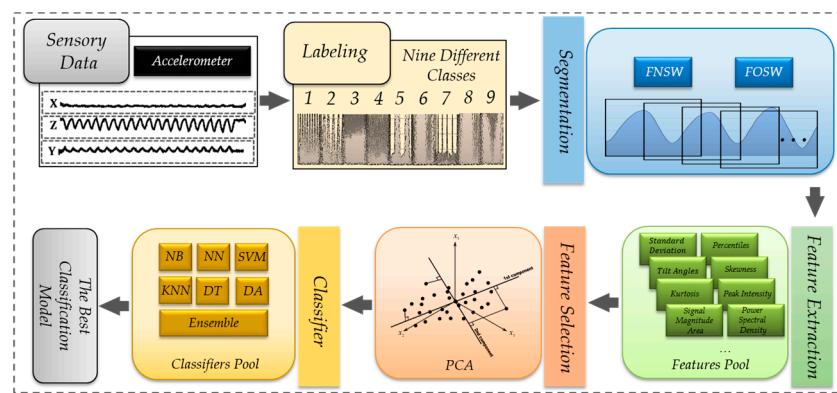
This paper is organized as follows. In Section 2, backgrounds in the field of human activity recognition and the adopted methodologies including feature extraction/selection and classification techniques with the parameters of prediction functions are addressed. The used datasets in this study will be discussed and listed in Section 3. Section 4 presents the experimental results obtained with

different cross-validation techniques. Finally, a conclusion and some research perspectives are given in Section 5.

## 2. Backgrounds and Methodologies

Human Activity Recognition (HAR) starts with collecting data from the motion sensors. The data are partitioned into windows to apply feature extraction thereby filtering relevant information in the raw signals. Afterward, extracted features are used as inputs of each classifier that ultimately yields the HAR model. To evaluate the effect of sensing heterogeneity on classifiers, we do not perform any preprocessing steps. This problem is formulated as follows:

Definition: With  $p$  extracted features from the motion sensors, given a set  $W = \{w_1, w_2, \dots, w_n\}$  of labeled and equal-sized time windows, and a set  $A = \{a_1, a_2, \dots, a_l\}$  of activity labels, the goal is to find the best classifier model  $C$ , such that for any  $w_k$  which contains a feature set  $F_k = \{f_{k,1}, f_{k,2}, \dots, f_{k,p}\}$ , the predicted label  $\hat{a}_k = C(F_k)$  is as identical as possible to the actual activity performed during  $w_k$ .  $p$  is the number of features in vector  $F_k$  extracted from  $w_k$ . Figure 1 depicts the whole system flow of sensor-based activity recognition for nine activities.



**Figure 1.** Sensor-based activity recognition procedure.

### 2.1. Data Segmentation, Feature Extraction and Selection

Stream of sensory data needs to be divided into subsequent segments. Fixed-size Sliding Window (FSW) is the most common method in segmentation step where the data stream is allotted into fixed-length windows with no inter-window gaps. If there is no degree of overlap between adjacent windows, it is called Fixed-size Non-overlapping Sliding Window (FNSW). The second method is Fixed-size Overlapping Sliding Window (FOSW), which is similar to FNSW except that the windows overlap during segmentation [8,9]. The use of overlap between adjacent windows has been shown to be effective in classification problem using wearable sensor data [14,15]. Finding the optimal window size  $t$  is an application-dependent task. The window size should be properly determined in such a way that each window is guaranteed to contain enough samples (at least one cycle of an activity) to differentiate similar movements. In addition, increasing the window size does not necessarily enhance the accuracy but may add computational complexity (causing higher latency). To better address the challenge, we analyze the influence of window sizes (ranging from 1 s to 15 s) on the classification performance.

Feature extraction is to obtain the important characteristics of a data and represent them into a feature vector used as input of a classifier [16]. Table 1 gives details about the most effective time/frequency-domain and heuristic features in the literature in the context of activity recognition. Due to low computational cost and high discriminatory ability of time-domain features, they are the most frequently employed features for real-time applications. We compute all the features listed in Table 1 using each reading of accelerometer sensor consists of 3-D accelerations (x, y, z). However,

to minimize the effects of sensor orientation, we add another dimension to the sensor readouts, which is called the magnitude of the accelerometer vector, i.e.,  $\sqrt{x^2 + y^2 + z^2}$ , because it is less sensitive to the orientation changes [17]. It is worth noting that the correlation features are calculated between each pair of axes, and the tilt angles are estimated by combination of all three axes as shown in Table 1. Each classifier is fed with the feature vectors obtained from fusing data at the feature level. As a result of the above feature extraction process, a total of 176 features are obtained for each segment and then scaled into interval [0, 1] using min-max normalization so as to be used for classification.

As not all features are equally useful in discriminating between activities, Principal Component Analysis (PCA) is applied to map the original features  $F_k = \{f_{k,1}, f_{k,2}, \dots, f_{k,p}\}$  into a lower dimensional subspace (i.e., new mutually uncorrelated features)  $F'_k = \{f'_{k,1}, f'_{k,2}, \dots, f'_{k,m}\}$ , where  $m \leq p$  [18]. It also significantly reduces the computational effort of the classification process. The PCA components can be counted by  $X = YP$ , where  $X$  and  $Y$  are centering and input matrix, respectively and  $P$  is a matrix of eigenvector of the covariance vector matrix  $C_x = P\Lambda P^T$ .  $\Lambda$  is a diagonal matrix whose diagonal elements are the eigenvalues corresponding to each eigenvector [19]. The new feature vectors are so-called principal components and arranged according to their variance (from largest to lowest). To keep the essential information in acceleration data that describe human activity, we take the first principal components that explain 95% of the total variance. The pairwise scatter plots of the first four components (transformed features) of one of test cases are given in Figure 2. As expected, the first components (the first component against the second component) for different classes are better clustered and more distinct.

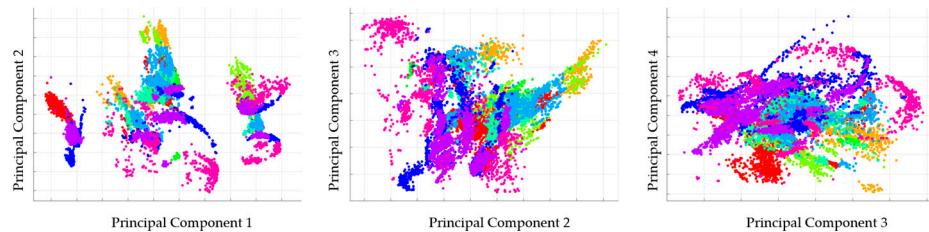
**Table 1.** The features list.

| Feature                          | Description   | Feature                                  | Description   |
|----------------------------------|---|--|---|
| <b>Mean</b>                      | $\mu_s = \frac{1}{n} \sum_{i=1}^n s_i$  | <b>Skewness</b>                          | $\frac{1}{n\sigma_s^3} \sum_{i=1}^n (s_i - \mu_s)^3$  |
| <b>Minimum</b>                   | $\min(s_1, s_2, \dots, s_n)$  | <b>Kurtosis</b>                          | $\frac{1}{n\sigma_s^4} \sum_{i=1}^n (s_i - \mu_s)^4$  |
| <b>Maximum</b>                   | $\max(s_1, s_2, \dots, s_n)$  | <b>Signal Power</b>                      | $\sum_{i=1}^n s_i^2$  |
| <b>Median</b>                    | $\text{median}(s_1, s_2, \dots, s_n)$   | <b>Root Mean Square</b>                  | $\sqrt{\frac{1}{n} \sum_{i=1}^n s_i^2}$   |
| <b>Standard Deviation</b>        | $\sigma_s = \sqrt{\frac{1}{n} \sum_{i=1}^n (s_i - \mu_s)^2}$  | <b>Peak Intensity</b>                    | The number of signal peaks within a certain period of time  |
| <b>Coefficients of Variation</b> | $\frac{\sigma_s}{\mu_s}$  | <b>Pearson's Correlation Coefficient</b> | $\frac{\text{cov}(a,b)}{\sigma_a \sigma_b}$   |
| <b>Peak-to-peak Amplitude</b>    | $\max(s) - \min(s)$   | <b>Inter-axis Cross-Correlation</b>      | $\frac{\sum_{i=1}^n (a_i - \mu_a)(b_i - \mu_b)}{\sqrt{\sum_{i=1}^n (a_i - \mu_a)^2 \sum_{i=1}^n (b_i - \mu_b)^2}}$  |
| <b>Percentiles</b>               | $t = \frac{np_i}{100} + 0.5$ , $p_i = 10, 25, 50, 75, 90$<br>$\text{percentile}(s, p_i) = (1-f)s_k + fs_{k+1}$<br>$k = \text{integer part of } t; f = \text{fractional part of } t$ | <b>Autocorrelation</b>                   | $R(k) = \frac{1}{(n-k)\sigma_s^2} \sum_{i=1}^{n-k} (s_i - \mu)(s_{i+k} - \mu) \quad \forall k < n$ ; the height of the first and second peaks and the position of the second peak of $R(k)$   |
| <b>Interquartile Range</b>       | $\text{percentile}(s, 75) - \text{percentile}(s, 25)$   | <b>Trapezoidal Numerical Integration</b> | $\int_1^n s(x)dx$ using Multiple Segment Trapezoidal Rule   |
| <b>Pitch Angle</b>               | $\arctan(\frac{x_i}{\sqrt{y^2+z^2}})$   | <b>Signal Magnitude Area</b>             | $\frac{1}{n} \sum_{i=1}^n ( x_i  +  y_i  +  z_i )$  |
| <b>Roll Angle</b>                | $\arctan(\frac{y_i}{\sqrt{x^2+z^2}})$   | <b>Signal Vector Magnitude</b>           | $\frac{1}{n} \sum_{i=1}^n \sqrt{x_i^2 + y_i^2 + z_i^2}$   |
| <b>Median Crossings</b>          | $t = s - \text{median}(s)$<br>$MC = \sum_{i=1}^n \text{sgn}(t_i, t_{i+1})$<br>$\text{sgn}(a,b) = \{1 \text{ if } (a,b) < 0; 0 \text{ if } (a,b) > 0\}$                              | <b>Power Spectral Density</b>            | $\frac{1}{n} \sum_{i=1}^{n-1} (s_i \cos \frac{2\pi f_i}{n})^2 + (s_i \sin \frac{2\pi f_i}{n})^2$ $f$ denotes the $f^{\text{th}}$ Fourier coefficient in the frequency domain; the positions and power levels of highest 6 peaks of PSD computed over the sliding window; total power in 5 adjacent and pre-defined frequency bands. |

## 2.2. Machine Learning Techniques

In this study, we are dealing with the supervised machine learning methods where the class labels are used to train the feature vectors extracted from each separate segment of data. We attempt the most complete analysis on performance of classifiers to discriminate among different types of activity. Wide range of machine learning methods have been applied for recognition of human activities such

as decision tree (DT) [20–25], Support Vector Machines (SVM) [4,5,20,23,25–28], K-Nearest Neighbors (KNN) [20,21,23,25,27], Naïve Bayes (NB) [20,23,25,29], artificial Neural Network (NN) [23,24,26,30] and ensemble of classifiers [6,20,25,26].



**Figure 2.** The pairwise scatter plots of the first four components.

We explore 293 different classifiers including Decision Tree, Discriminant Analysis, Support Vector Machines, K-Nearest Neighbors, Ensemble Methods, Naïve Bayes and Neural Network with their different parameters. The methods and their parameters setting are described and given IDs in Appendix A. The main objective of implementing different classification techniques is to review, compare and evaluate their performance considering the most well-known heterogeneous datasets publicly open to the research community. We are going to intertwine different issues and suggest solutions if we expect reasonable results in the practical applications.

### 3. Datasets

To design a robust learning model working in more realistic conditions, we combined 14 datasets, focusing on accelerometer sensors that contain several sources of heterogeneities such as measurement units, sampling rates and acquisition protocols that are present in most real-world applications. Table 2 listed the datasets and brought the details of the collected data in each project. In total, the aggregated dataset has about 35 million acceleration samples from 228 subjects (with age ranging from 19 to 83) of more than 70 different activities. This is the most complete, realistic, and transparent dataset in this context.

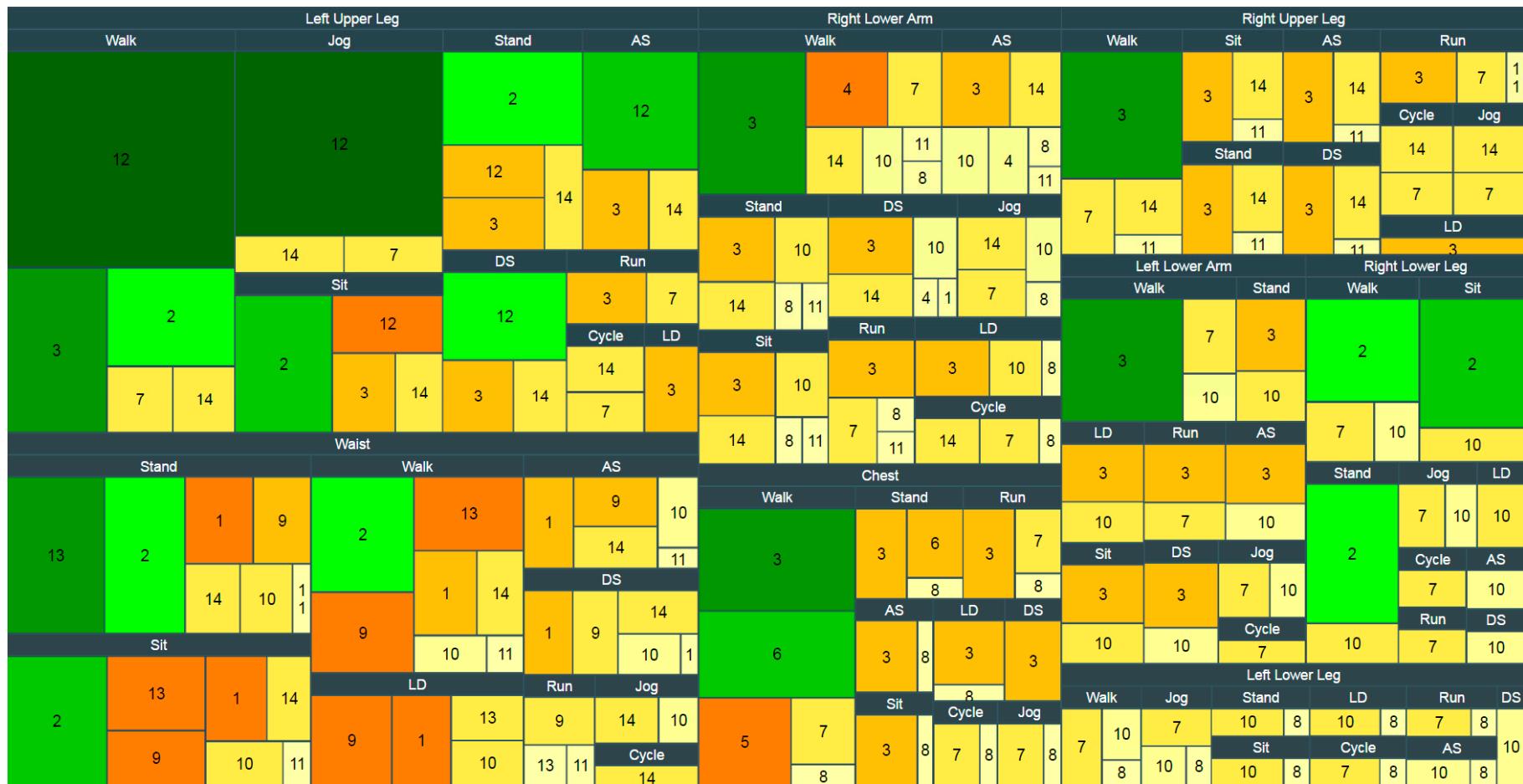
We considered 10 major positions on the body i.e., Waist (W), Right Lower Arm (RLA), Left Lower Arm (LLA), Right Upper Arm (RUA), Left Upper Arm (LUA), Right Lower Leg (RLL), Left Lower Leg (LLL), Right Upper Leg (RUL), Left Upper Leg (LUL), and Chest (C). All sensors positions described in each dataset have been mapped into the major positions. For example, if a subject puts the cellphone in the left front pants pocket, we consider it as Left Upper Leg (LUL) position, or wrist, which is a great place for many commercial wearables, is considered as RLA/LLA in this paper. In this field, numerous studies [31,32] have shown that the performance of HAR systems strongly depends on sensor placement since the number and the placement of inertial sensors have direct effects on the measurement of bodily motions. Each placement turns out to be more suitable in terms of performance for particular activities. Besides, having fewer sensors attached to the body is more preferable since wearing multiple ones can become burdensome and is not well-accepted. Therefore, we limited our modeling and analysis for single-accelerometer data while we still expect a sufficiently high recognition rate for the picked activities. According to the datasets, the most examined activities (top activities) are walking, running, jogging, cycling, standing, sitting, lying down, ascending and descending stairs which also represent the majority of everyday living activities. Another observation we find is that in eight major positions we have data for all top activities. Therefore, we choose them as target activities for eight separate positions (W, RLA, LLA, RUL, LUL, RLL, LLL and C). We created rectangular tree map that presents dense volumes of data in a space filling layout allowing for the visual comparison of datasets contributions in each target position (see Figure 3). For example, as depicted in Figure 3, datasets 3, 5, 6, 7 and 8 contribute data for constructing the chest dataset with nine activities.

**Table 2.** The datasets used in this study.

| Dataset  | Number of Subjects             | Sensor Type  | Frequency | Sensor Placement   | Activity Type  | Description  |
|----------|--------------------------------|--|-----------|--|--|--|
| (1) [5]  | 30 (19–48 year)                | accelerometer gyroscope (Samsung Galaxy S II smartphone)           | 50 Hz     | waist (1)  | walking, ascending stairs, descending stairs, sitting, standing, laying (6)  | In the first trial, each subject placed the smartphone in a predetermined position i.e., the left side of the belt. However, in the second attempt, they could fix the phone in a desired position on the waist. |
| (2) [6]  | 4 (28–75 year)<br>(45 ± 21.49) | ADXL335 accelerometer (connected to an ATmega328V microcontroller) | ~8 Hz     | waist, left thigh, right ankle, right arm (4)  | walking, sitting, sitting down, standing, standing up (5)  | The data have been collected during 8 h of five different activities for all subjects.   |
| (3) [27] | 8 (20–30 year)                 | accelerometer gyroscope magnetometer (Xsens MTx unit)              | 25 Hz     | chest, right and left wrists, right side of the right knee, left side of the left knee (5)                   | walking in a parking lot, sitting, standing, lying, ascending/descending stairs, walking on a treadmill with a speed of 4 km/h (in flat and 15° inclined positions), etc. (19) | The subjects performed nineteen activities by their own style and were not controlled during data collection sessions.   |
| (4) [33] | 16 (19–83 year)                | accelerometer (6-bit resolution)                                   | 32 Hz     | right wrist (1)  | walking, climbing stairs, descending stairs, laying down on bed, sitting down on chair, brushing teeth, eating meat, etc. (14)   | There are postural transitions, reiterated and complex activities in the dataset.  |
| (5) [34] | 22 (25–35 year)                | Accelerometer (Google Nexus One)                                   | ~30 Hz    | jacket pocket on the chest (1)   | walking (1)  | The walking data of several subjects were collected in indoor and outdoor under real-life circumstances.   |
| (6) [34] | 15 (27–35 year)                | accelerometer (Shimmer)  | 52 Hz     | chest (1)  | walking, walking and talking, standing, standing up, talking while standing, going up/down stairs, etc. (7)  | They used a low-power, low-cost BeagleBoard with a Linux embedded operating system to transmit data over Bluetooth.  |
| (7) [21] | 17 (22–37 year)                | accelerometer gyroscope magnetometer (Xsens MTx unit)              | 50 Hz     | right and left calves, right and left thighs, back, right and left lower arms and right, left upper arms (9) | walking, jogging, running, jump up, rowing, cycling, etc. (33)   | The dataset includes a wide range of physical activities (warm up, cool down and fitness exercises).   |
| (8) [22] | 10                             | accelerometer gyroscope magnetometer (Shimmer)                     | 50 Hz     | chest, right wrist, left ankle (3)   | walking, sitting and relaxing, standing still, lying down, climbing stairs, running, cycling, etc. (12)  | This dataset covers common activities of the daily living, given the diversity of body parts involved in each one, the intensity of the actions and their execution speed or dynamicity.                         |

**Table 2.** *Cont.*

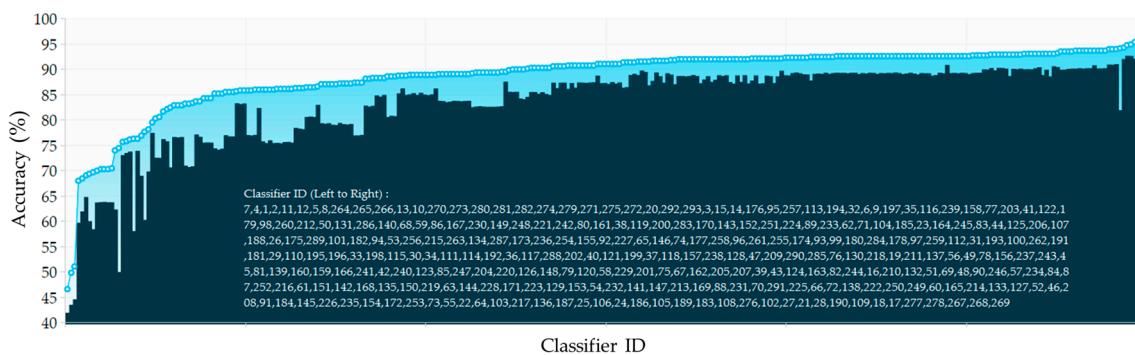
| Dataset   | Number of Subjects                    | Sensor Type   | Frequency | Sensor Placement   | Activity Type   | Description  |
|-----------|---------------------------------------|---|-----------|--|---|--|
| (9) [35]  | 14 (21–49 year)<br>( $30.1 \pm 7.2$ ) | accelerometer gyroscope<br>(MotionNode)                       | 100 Hz    | front right hip (1)  | walking forward, left and right, sitting and fidgeting, standing, going upstairs and downstairs, running forward, jumping up and down, etc. (12)                | There were 5 trials for each activity and each subject performed the experiments on different days at indoor and outdoor places.   |
| (10) [36] | 20 (19–75 year)                       | accelerometer 2-axis gyroscope<br>(attached to Tmote Sky)     | 30 Hz     | waist, right and left wrists, right and left ankle (5)   | walking forward, right-circle and left-circle, sitting, lying down, standing, going upstairs and downstairs, jogging, jumping, turning right and left etc. (13) | The design of the wearable sensor network was based on platform named DexterNet that implemented a 3-level architecture for controlling heterogeneous body sensors.        |
| (11) [23] | 4 (25–30 year)                        | accelerometer gyroscope<br>(Samsung Galaxy S II)              | 50 Hz     | belt, right arm, right wrist and right jeans pocket (4)  | walking, sitting, standing, walking upstairs and downstairs, running (6)  | Every participant performed each activity between 3 and 5 min. The smartphone was horizontally kept for belt and vertically for the arm, wrist, and pocket.                |
| (12) [24] | 36                                    | accelerometer<br>(Android-based smartphone)                   | 20 Hz     | front pants leg pocket (1)   | walking, sitting, standing, upstairs, downstairs, jogging (6)   | The android app, through a simple graphical user interface, permits to record the user's name, start and stop the data collection, and label the activity being performed. |
| (13) [37] | 19 (23–52 year)                       | accelerometer gyroscope<br>magnetometer (Xsens MTx unit)      | 100 Hz    | belt either on the right or the left part of the body, at the subject's choice (1)                 | walking, sitting, standing, lying, running, falling, jumping (9)  | Data were logged in indoor and outdoor settings under semi-naturalistic conditions.  |
| (14) [25] | 10 (25–30 year)                       | accelerometer gyroscope<br>magnetometer (Samsung Galaxy S II) | 50 Hz     | right and left jeans pocket, belt position towards the right leg, right upper arm, right wrist (5) | walking, sitting, standing, walking upstairs and downstairs, jogging, biking (8)  | All test protocols were carried inside a building, except biking.  |



**Figure 3.** The rectangular tree map that presents dense volumes of data in a space filling layout to see datasets contributions in each target position. Laying Down (LD), Ascending Stairs (AS), Descending Stairs (DS). The number inside each rectangle indicates the dataset number (see first column of Table 2).

#### 4. Experimental Results and Discussions

In this section, we report the effects of the heterogeneities, from sensors characteristics, data collection scenarios and subjects, on various feature representation techniques and 293 classifiers considering two cross-validation techniques. First, the 10-fold cross-validation strategy is applied as one of the most accurate approaches for model selection. Figure 4 shows the minimum and maximum obtained accuracy of each classifier over different window sizes with the waist accelerometer data. The algorithms are sorted according to the best obtained accuracy. Considering the best acquired accuracy for each classification category in this position, the ensemble methods KNN (Subspace) and Tree (Bagging) achieved the highest activity recognition rate whereas DT performed the worst. Further DA, DA (Subspace), Tree (AdaBoost), Tree (RUSBoost) and NB performed almost equal but worse than SVM, NN and KNN. As can be seen, some classification learning algorithms are more sensitive to parameters settings and window size and may thus be more likely to exhibit significant differences. To have a better and deeper investigation, we extracted the classifiers with top 5% accuracies and call them “*topClassifiers*” for each position. Figure 5 depicts the range of *topClassifiers* accuracies for each position. A red dashed line annotation shows the 95th percentile of obtained accuracies.



**Figure 4.** The minimum and maximum accuracy of each classifier over different window sizes, ranging from 1 s to 15 s, with the waist accelerometer data.

As can be seen in this figure, most of the recognition methods remain consistent in their relative performance across different accelerometer data obtained from different positions. As explained in Section 2, finding the optimal length of window size is an application-dependent task. The window size should be properly determined in such a way that each window is guaranteed to contain enough samples to differentiate similar activities or movements. Thus, we consider different window sizes ranging from 1 s to 15 s in steps of 1 s to ensure the statistical significance of the calculated features. It comprises most of the values used in the previous activity recognition systems [8]. Figure 6 describes the ranking of different window sizes in providing the best accuracy (among all classifiers) in each position. For example, window of length 7 s provides the best classification accuracy when the sensor is attached on the RLA. The second best accuracy value for RLA is achieved with window size 8 s. The first line in this figure shows the window sizes of the best accuracies for each position. In contrast to LLL where the top four accuracy values have been observed in small window sizes ranging from 2 s to 5 s, chest provides the top-rank accuracy values in large window sizes from 10 s to 15 s. This observation is more highlighted in orange line ( $w = 1$  s) where all positions obtain the worst case accuracy values except for the LLL. However, in some cases we can change the window size (increase/decrease) at the expense of a subtle performance drop. For example, in chest position, the window size can be reduced from 15 s to 7 s by only tolerating 0.16% in recognition performance.

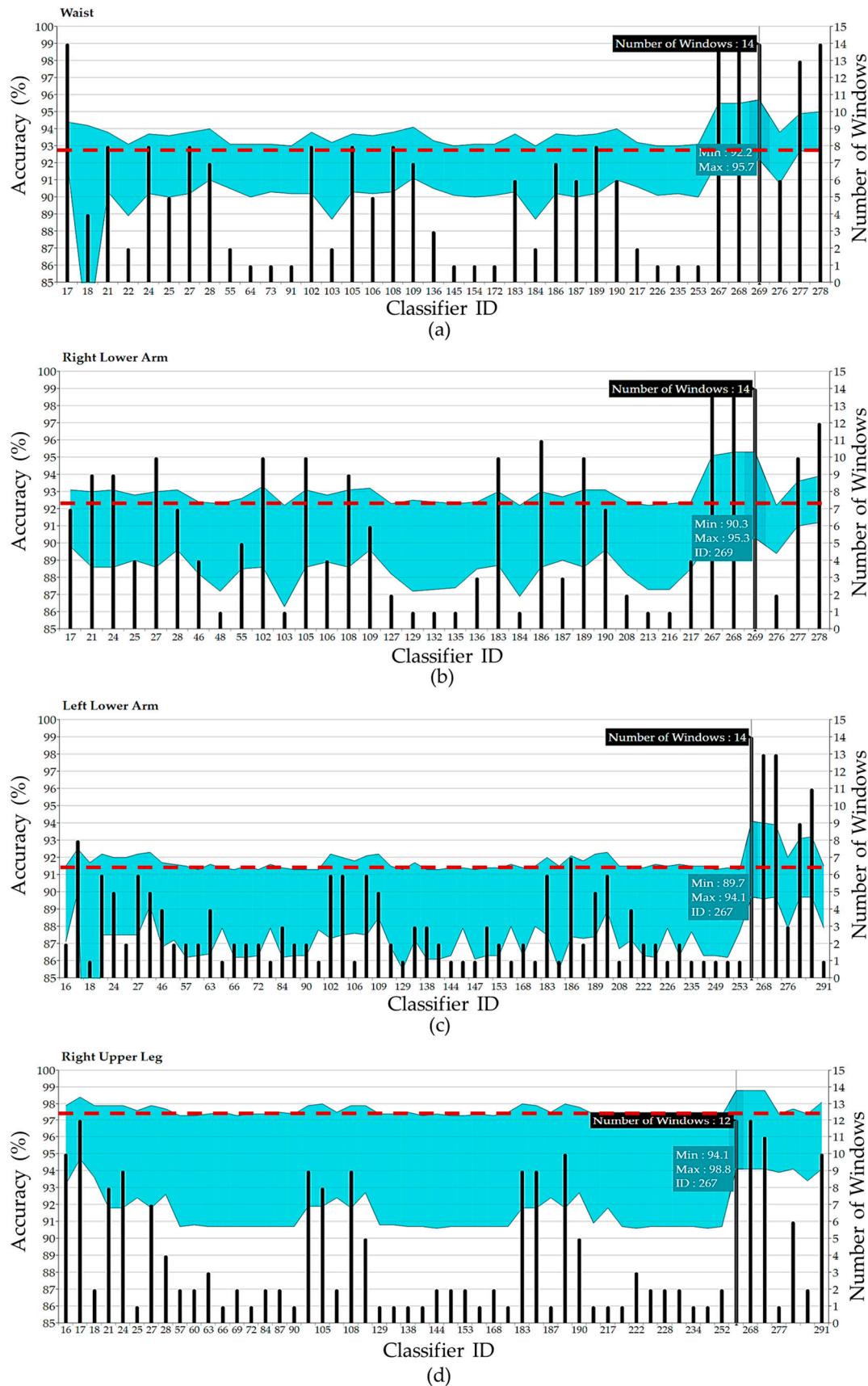
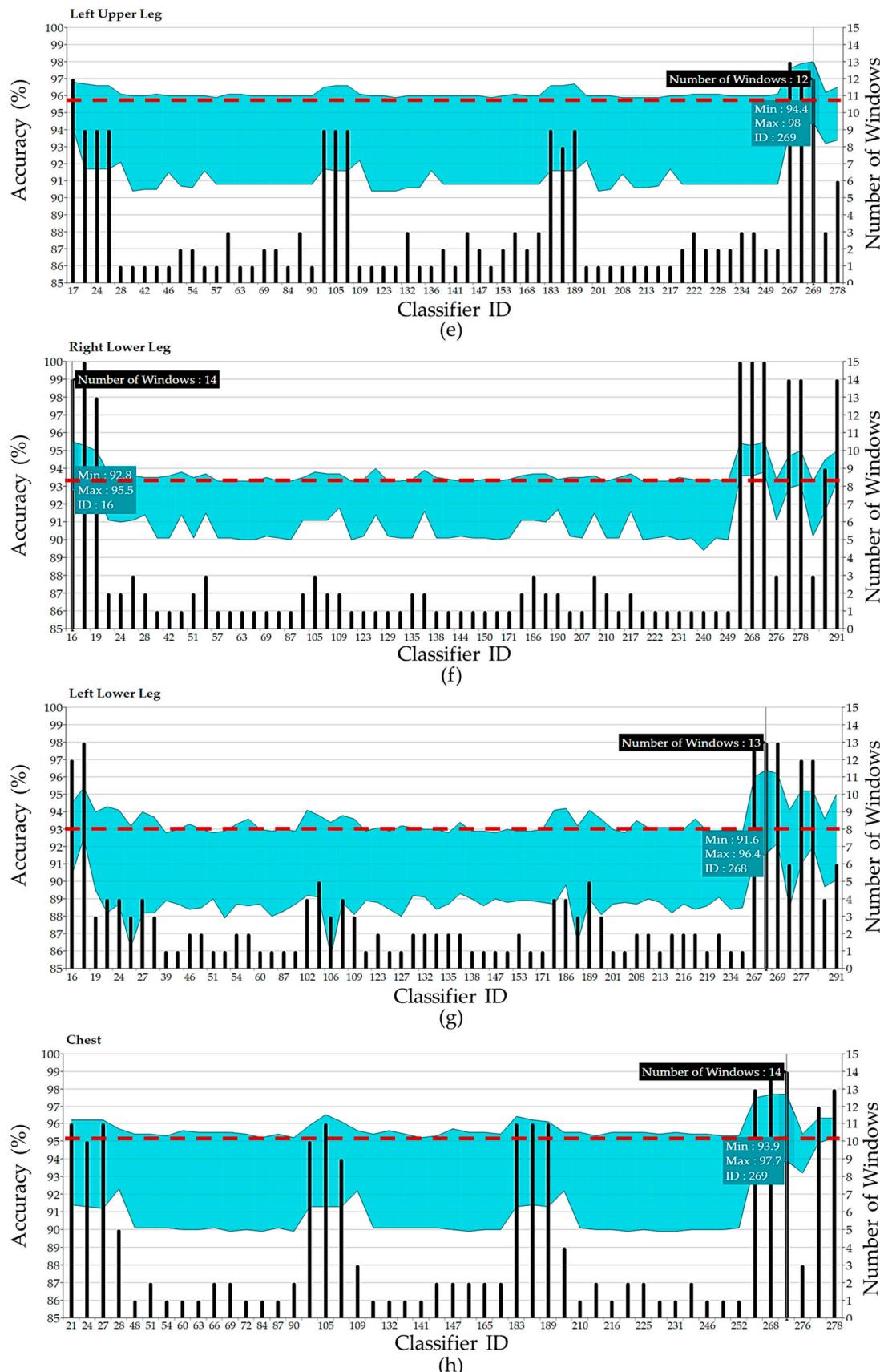
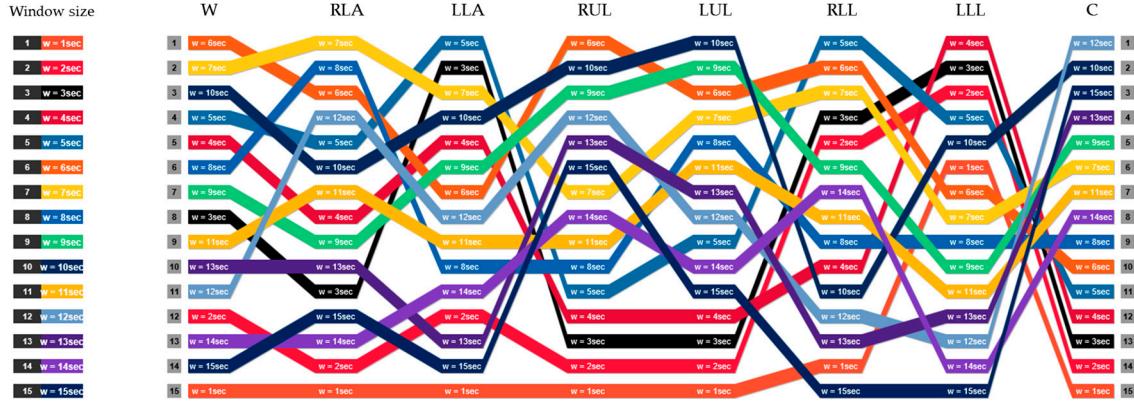


Figure 5. Cont.

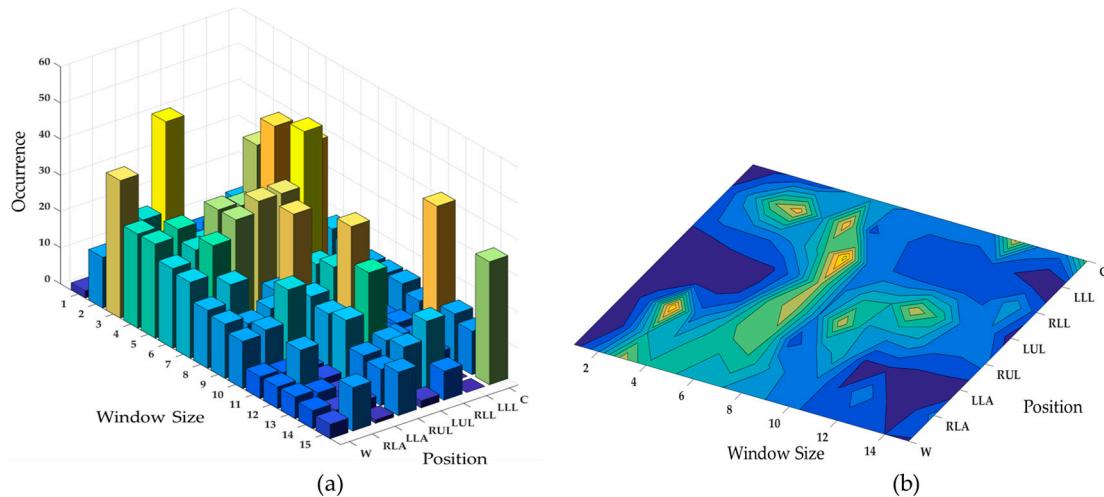


**Figure 5.** The range of *topClassifiers* accuracies for (a) Waist; (b) RLA; (c) LLA; (d) RUL; (e) LUL; (f) RLL; (g) LLL; and (h) Chest.



**Figure 6.** The rank of window sizes in providing the best accuracy in each position.

The bar charts in Figure 5 indicate the number of window sizes (1 s to 15 s) in which the *topClassifiers* provided good results (top 5%). An interesting observation from the bars is that some classifiers such as KNN (Subspace), Tree (Bagging) and SVM work well with most windows sizes. This means they could mitigate the effect of window size to gain meaningful information for the activity classification process. To have a better understanding of window size effect on accuracy, Figure 7 shows the *topClassifiers* across all window sizes in each position. As can be observed, the interval 3–10 s proves to provide the best accuracies in most cases considering the target activities. This range can be reduced if fusion of multiple sensors is used for feature extraction. Another point worth mentioning that is for the underlying periodic activities, large window sizes do not necessarily translate into a better recognition performance.

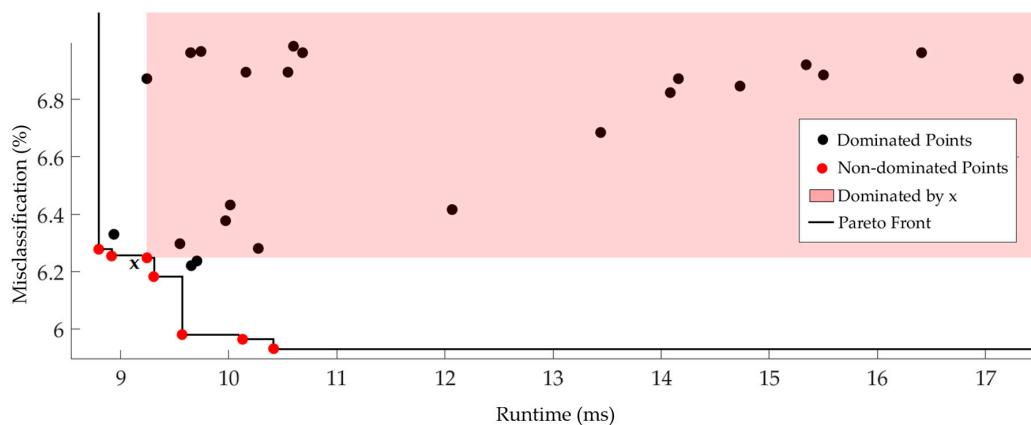


**Figure 7.** Effect of window size to gain meaningful information for the activity classification in (a) 3D and (b) 2D representations.

Although accuracy is necessary for all recognition algorithms, it is not the only parameter to consider in designing a recognition model. The runtime complexity (classification step) is another important challenge as the model should be working fast and responsive regardless of where it is deployed i.e., on/off-device. Thus, we make use of the concept of Pareto optimality to extract superior solutions from *topClassifiers* to tradeoff classifier accuracy and runtime. We consider two objective functions, i.e., misclassification and runtime, to be minimized. A feasible solution  $x$  dominates a feasible solution  $y$  when

$$\forall i, f_i(x) \leq f_i(y)$$

where  $f_i$  is the  $i$ th objective function. However, in many problems, there is usually no single solution that is superior to all others, so the non-dominated solutions compose the Pareto front. For example, in Figure 8, we populate the runtime-accuracy plane with some *topClassifiers* for waist position and depict the Pareto front. The shaded area represents the region in  $f_1 \times f_2$  space that is dominated by the point  $x$  which is non-dominated and hence belong to the Pareto front [38]. All points in this region are inferior to  $x$  in both objectives. In addition, if we want to minimize an objective to a constraint, e.g.,  $f_1(x) < c$ , the Pareto front provides the solution for all possible values of the cap  $c$  [38]. Therefore, the Pareto front contains significantly richer information than one obtains from single-objective formulations. Table 3 summarizes the non-dominated classifiers in each position. The results show a clear tradeoff between classifiers runtime and accuracy. There is no strong relation between the sensor position and classification performance. In overall, the highest classification accuracy was achieved by KNN (Subspace), and KNN stayed in the second place, which is followed by SVM and NN. Figure 9a depicts the overall view of the non-dominated classifiers and their power in providing high recognition accuracy. The size of each classifier ID in this figure represents the number of times that the corresponding classifier has been reported in Table 3.

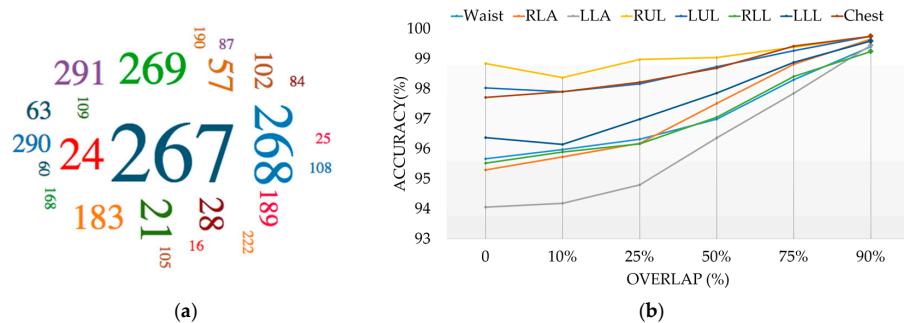


**Figure 8.** Illustration of some Pareto fronts when minimizing two objectives (misclassification and classification runtime) according to the obtained results in the waist.

The KNN has the best classification runtime ( $7 \pm 1.78$  ms) fed with a feature vector among all of them. While for classification accuracy, it is always after its ensemble method KNN (Subspace). In all cases, KNN (Subspace) with average accuracy ( $96.42\% \pm 1.63\%$ ) provided better results than all other non-dominated classifiers, with the exception of data from the RLL, where the SVM (95.52%) provided superior accuracy. However, SVM's prominence is negligible while considering its runtime (113.05 ms) and no significant accuracy improvement (0.1%). Given the accuracy results stated in Table 3, although NN classifiers provide promising results in most cases, they are dominated by other techniques and could be only among the selected methods in three positions RUL, RLL and LLL. With a closer look at the classifications results in Figure 5 and tabulated results, we find out ensemble method Tree (Bagging) is a very strong method and is among *topClassifiers* in all cases, but is always outperformed by other methods in terms of both accuracy and runtime. According to the selected KNN classifiers, the distance method affects most in the performance. City block and Euclidean have been the best choices and too large value for  $k$  does not improve the performance as it destroys locality. We also can draw another conclusion that there was no significant difference in KNN (Subspace) performance with different number of learners (i.e., 10, 20 and 40). Therefore, applying fewer learners is preferentially utilized due to its much lower runtime.

Regarding the position analysis, the best performance (98.85% and 98.03%) is achieved with the aggregated data on RUL and LUL. Chest is the next best performing placement (97.72%) compared to the RLL (95.52%), LLL (96.38%), RLA (95.30%), LLA (94.06%) and Waist (95.67%).

Different studies [15,26,39] show that the use of overlap between successive sliding windows help the classifiers to be trained with more feature vectors and consequently improve the recognition performance. However, it adds more computational work needed to process the overlapped data multiple times. To evaluate the effectiveness of the degree of overlap, the overlaps of 10%, 25%, 50%, 75% and 90% are used where the percentage is the amount the window slides over the previous window. For instance, a sliding window with 25% overlap will start the next window while the previous window is 75% complete. The value can range from 0% to 99%, since a 100% sliding window is erroneous. Figure 9b illustrates the recognition system capabilities for diverse overlap values while keeping the best window size in each position (see Figure 6). The results demonstrate that the performance tendency is increased in most cases by overlapping more data segmentations. An average increase of 3.28% in the best accuracy was found between the 0% and 90% overlap scenarios and all obtained with ensemble of KNN. Figure 10 also illustrates the number of classifiers that provide good results (90%–99%) by considering different overlap sizes. The larger the overlap, the more improvement is expected in performance. As described, this is because more features can be trained and consequently the predictive model almost certainly works better in testing phase; however, it suffers from more training time.

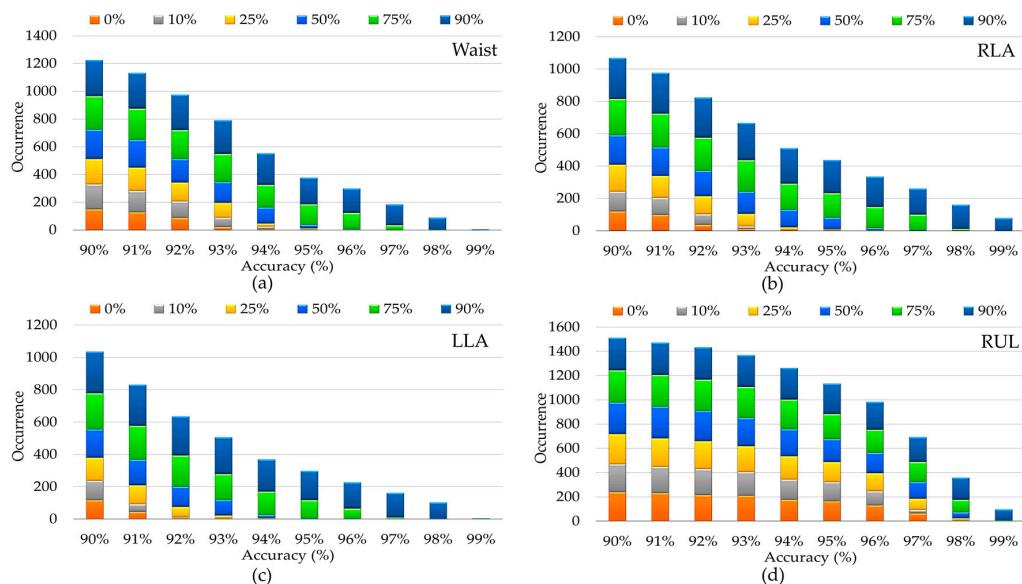


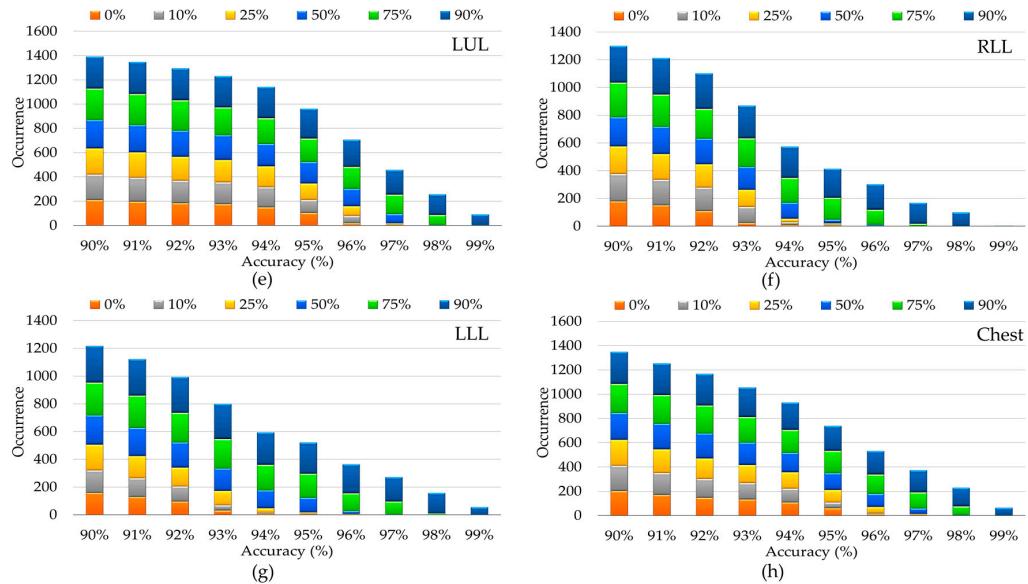
**Figure 9.** (a) Overall view of the non-dominated classifiers (classifier ID) and their power in providing high recognition accuracy (b) recognition system capabilities for diverse overlap values.

In activity recognition problem, K-fold cross-validation is an accurate approach for model selection; however, Leave-One-Subject-Out (LOSO), which is also called subject-independent cross-validation, can be used to avoid possible overfitting and is one of the best approaches in estimating realistic performance. Because LOSO reflects inter-subject variability and tests on yet-unseen data, it consequently leads to a decrease in accuracy. According to combined datasets in each position (see Figure 3), we conducted LOSO evaluation where the KNN (Subspace) is trained on activity data for all subjects except one. Then, the classifier is tested on the data for only the subject left out of the training data set. The procedure is then repeated for all subjects in each position and the mean accuracy is reported. For each position, the window size and overlap are set based on the best acquired results from 10-fold evaluation. Based on the aggregated data for each position, the models trained in LUL (92.35%) and LLL (90.03%) positions were not affected much by interpersonal differences in the body movement of subjects and could obtain relatively good results of accuracy. They are followed by Chest (86.31%) and RLL (83.51%), which are better performing placements compared to the lower arm positions. The RLA (64.62%) was influenced the most and LLA (77.12%) accounted for a substantial decrease in accuracy, as well. Result of Waist (72.53%) was very similar to one reached by RUL (72.91%), but both suffer performance degradation by more than 25%. As expected the accuracy of recognition in all positions was reduced since the trained model deals with a set of data that might have new measurement characteristics. As the best classifier of each position in average goes down 17.13% in accuracy, there is a need for further studies investigating new relevant features and novel machine learning model to be sufficiently flexible in dealing with inter-person differences in the activities' performances in a position-aware scenario.

**Table 3.** The accuracy and runtime of non-dominated classifiers.

| Classifier ID          | Accuracy (%) | Misclassification (%) | Runtime (ms) | Classifier ID          | Accuracy (%) | Misclassification (%) | Runtime (ms) |
|------------------------|--------------|-----------------------|--------------|------------------------|--------------|-----------------------|--------------|
| <b>Waist</b>           |              |                       |              | <b>Left Upper Leg</b>  |              |                       |              |
| 21                     | 93.82        | 6.18                  | 9.31         | 21                     | 96.69        | 3.31                  | 3.52         |
| 28                     | 94.02        | 5.98                  | 9.57         | 24                     | 96.62        | 3.38                  | 3.39         |
| 108                    | 93.75        | 6.25                  | 9.24         | 57                     | 95.93        | 4.07                  | <b>2.92</b>  |
| 109                    | 94.07        | 5.93                  | 10.41        | 60                     | 96.11        | 3.89                  | 3.11         |
| 183                    | 93.75        | 6.25                  | 8.92         | 222                    | 96.15        | 3.85                  | 3.16         |
| 189                    | 93.72        | 6.28                  | <b>8.80</b>  | 267                    | 97.63        | 2.37                  | 34.73        |
| 190                    | 94.04        | 5.96                  | 10.13        | 268                    | 97.86        | 2.14                  | 102.90       |
| 267                    | 95.48        | 4.52                  | 45.95        | 269                    | <b>98.03</b> | <b>1.97</b>           | 151.83       |
| 268                    | 95.51        | 4.49                  | 121.58       | <b>Right Lower Leg</b> |              |                       |              |
| 269                    | <b>95.67</b> | <b>4.33</b>           | 196.10       | 16                     | <b>95.52</b> | <b>4.48</b>           | 113.05       |
| <b>Right Lower Arm</b> |              |                       |              | 24                     | 93.82        | 6.18                  | 8.03         |
| 24                     | 93.11        | 6.89                  | <b>7.61</b>  | 28                     | 93.45        | 6.55                  | <b>8.02</b>  |
| 102                    | 93.29        | 6.71                  | 8.01         | 267                    | 95.36        | 4.64                  | 30.73        |
| 267                    | 95.14        | 4.86                  | 33.90        | 290                    | 94.52        | 5.48                  | 8.25         |
| 268                    | 95.29        | 4.71                  | 87.80        | 291                    | 94.97        | 5.03                  | 9.14         |
| 269                    | <b>95.30</b> | <b>4.70</b>           | 149.78       | <b>Left Lower Leg</b>  |              |                       |              |
| <b>Left Lower Arm</b>  |              |                       |              | 21                     | 94.26        | 5.74                  | 7.91         |
| 28                     | 92.29        | 7.71                  | 7.59         | 25                     | 93.16        | 6.84                  | <b>7.79</b>  |
| 57                     | 91.50        | 8.50                  | <b>7.31</b>  | 267                    | 95.99        | 4.01                  | 32.36        |
| 63                     | 91.61        | 8.39                  | 7.35         | 268                    | <b>96.38</b> | <b>3.62</b>           | 83.45        |
| 102                    | 92.24        | 7.76                  | 7.50         | 290                    | 93.64        | 6.36                  | 7.89         |
| 267                    | <b>94.06</b> | <b>5.94</b>           | 32.50        | 291                    | 95.02        | 4.98                  | 8.85         |
| <b>Right Upper Leg</b> |              |                       |              | <b>Chest</b>           |              |                       |              |
| 24                     | 97.93        | 2.07                  | 7.04         | 21                     | 96.17        | 3.83                  | 6.97         |
| 57                     | 97.33        | 2.67                  | <b>6.94</b>  | 84                     | 95.25        | 4.75                  | <b>6.58</b>  |
| 63                     | 97.43        | 2.57                  | <b>6.94</b>  | 87                     | 95.43        | 4.57                  | 6.95         |
| 183                    | 98.05        | 1.95                  | 7.49         | 105                    | 96.48        | 3.52                  | 7.63         |
| 189                    | 97.97        | 2.03                  | 7.21         | 168                    | 95.39        | 4.61                  | 6.91         |
| 267                    | <b>98.85</b> | <b>1.15</b>           | 32.12        | 183                    | 96.37        | 3.63                  | 7.02         |
| 291                    | 98.14        | 1.86                  | 8.08         | 267                    | 97.52        | 2.48                  | 29.64        |
|                        |              |                       |              | 268                    | 97.67        | 2.33                  | 76.49        |
|                        |              |                       |              | 269                    | <b>97.72</b> | <b>2.28</b>           | 125.10       |

**Figure 10. Cont.**



**Figure 10.** Analysis of number of classifiers, which provide good results (90%–99%) by taking different overlap sizes into account for different positions.

## 5. Conclusions

In this study, different machine learning techniques were deeply explored when heterogeneity of devices and their usage scenarios are intrinsic. In addition, each position was analyzed based on the aggregated tri-axial accelerometer data from different datasets. Hence, the quantitative comparison of the classifiers was hindered by the fact that each position is explored with a different aggregated dataset. In each position investigation, in addition to different sources of heterogeneities in data, there are also different factors such as body shape, clothing, straps, belt and accidental misplacements/disorientations (in the form of rotations or translations) that make the analysis harder to have a solid model. The averaged results showed  $96.44\% \pm 1.62\%$  activity recognition accuracy when using K-fold and  $79.92\% \pm 9.68\%$  accuracy when using a subject-independent cross-validation. According to the obtained results, it is clear that new data with different sources of heterogeneities could significantly reduce the accuracy with more than 32% (e.g., in RLA) based on LOSO evaluation. An overall look into the results, KNN and its ensemble methods showed stable results over different positions and window sizes, indicating its ability in designing a robust and responsive machine learning model in the wearables, and they are followed by NN and SVM. However, as we showed in this paper, the choice of parameter values in each classifier can have a significant impact on recognition accuracy and should be taken into account (see Appendix A). Considering the promising results in this pilot study, we intend to work on novel features extraction methods and classifiers that outperform classical classification methods while better dealing with inter-person differences and data diversities. Another point that deserves to be further assessed is optimizing the runtime performance since it has a great role in efficiency of the cloud-based machine learning deployments.

**Author Contributions:** Majid Janidarmian designed the study, implemented the methodologies and drafted the manuscript. Atena Roshan Fekr has contributed to the revision of this paper and provided insightful comments and suggestions. Katarzyna Radcka and Zeljko Zilic directed the study and contributed to the overall study design as well as the analysis of results.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Abbreviations

The followings are the list of abbreviations: SC: Split Criterion; MS: Maximum number of Splits; T: Type; KF: Kernel Function; C: Coding; KS: Kernel Scale; P: Polynomial; GA: GAussian; OO: One-vs-One; PO: Polynomial Order; BT: Break Ties; D: Distance; DW: Distance Weight; EX: EXponent; N: number of Neighbors; SM: SMallest; NE: NEarest; RA: RAndom; CB: City Block; CC: ChebyChev; CO: CORrelation; CS: CoSine; EU: EUclidean; SE: Standardized Euclidean; MA: MAhalanobis; MI: MIkowsky; SP: SPearman; EQ: EQual; IN: INverse; SI: SquaredInverse; EL: Ensemble Learner; NL: Number of Learners; DI: DIstribution; TF: Training Function; HL: number of Hidden Layers; SCG:: Scaled Conjugate Gradient; RP: Resilient backPropagation; LM: Levenberg–Marquardt backpropagation.

## Appendix A.

In this section, the utilized approaches for activity classification problem are summarized while considering different parameters settings.

### Appendix A.1. Decision Tree

In this method, the discriminatory ability of the features is examined one at a time to create a set of rules. Decision Tree (DT) has been used in many studies [15,39], and the results show that it performs well with time and frequency domain features. In the top-down tree structure, each leaf represents a classification label and each branch denotes conjunctions of attributes that lead to the leaves. In other words, decision trees classify instances by starting at the root of the tree and moving through it (with decision being made at each node) until a leaf node. Generally, the utilized growing and pruning algorithms in decision tree induction are greedy and follow a recursive manner. The construction of a tree involves determining split criterion, stopping criterion and class assignment rule [40]. The most common techniques to measure the node impurity (for splitting nodes) are explained in Table A1 [41]. They define the node splits, where each split maximizes the decrease in impurity. In this study, we split branch nodes layer by layer until there are 4, 20 or 100 branch nodes. Therefore, we define nine different decision trees considering the mentioned split and stopping criteria. When a node is determined to be a leaf, it has to be given a class label. A commonly used class assignment function is the majority rule meaning that class  $k$  is assigned to node  $t$  as follows [42]:

$$l_k = \max_i p(i|t), t \in \tilde{T}$$

The set of terminal nodes (leafs) denoted by  $\tilde{T}$ .

### Appendix A.2. Discriminant Analysis

Discriminant Analysis (DA) is widely used in classification problems [43,44]. In this algorithm, there are three main elements: prior probability, posterior probability and cost [45]. A prior probability  $P(k)$  is the probability that an observation will fall into class  $k$  before you collect the data. There are two main choices, i.e., uniform and empirical. If the prior probability of class  $k$  is 1 over the total number of classes, it is called uniform. Besides, the number of training feature vectors of class  $k$  divided by all training features set defines the empirical prior probability.

**Table A1.** The common techniques for splitting nodes in DT.

| Split Criterion              | Description   | Split Criterion | Description   | Split Criterion | Description   |
|------------------------------|---|-----------------|---|-----------------|---|
| Gini's Diversity Index (GDI) | $\sum_i p(i)(1 - p(i)) = 1 - \sum_i p^2(i)$<br><i>p(i) is the probability that an arbitrary sample belongs to class li.</i> | Deviance        | <i>based the concept of entropy from information theory</i> | Towing rule     | $p(L)p(R)\left(\sum_i  L(i) - R(i) \right)^2$<br><i>Let L(i)/R(i) denote the fraction of members of class i in the left/right child node after a split and p(L)/p(R) are the fractions of observations that split to the left/right</i> |

A posterior probability is the probability of assigning observations to classes given the data. The product of the prior probability and the multivariate normal density explains the posterior probability that a point  $x$  belongs to label  $k$ . With mean  $\mu_k$  and covariance  $\Sigma_k$  at a point  $x$ , the density function of the multivariate normal can be described as below [45]:

$$(x|k) = \frac{1}{(2\pi|\Sigma_k|)^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)\right)$$

The observed data  $x$  (the feature vector of one analysis segment) is classified to label  $k$  with the largest posterior probability. The posterior probability that an observation  $x$  belongs to label  $k$  is:

$$\hat{P}(k|x) = \frac{P(x|k)P(k)}{P(x)}$$

where  $P(x)$  indicates the probability of the feature vector  $x$  and is the sum over  $k$  of  $P(x|k)P(k)$ . Therefore, the predicted classification  $\hat{y}$  with  $m$  classes is:

$$\hat{y} = \arg \min_{y=1 \dots k} \sum_{k=1}^m \hat{P}(k|x)C(y|k)$$

$C(y|k)$  is the cost of classifying an observation  $y$  when its correct label is  $k$  [45]. In this paper, we consider five types of discriminant analysis classifiers: linear; and diagonal and pseudo variants of linear and quadratic types.

### Appendix A.3. Support Vector Machine

A Support Vector Machine (SVM) is defined based on one or a set of separating hyperplanes in a high dimensional space and was first proposed for binary-classification problems. SVM generates linear functions by considering a set of labels obtained from training dataset. The linear separator is created considering the maximum margin from the hyperplane to the support vectors [46,47]. By  $n$  training samples,  $(x_i, y_i); i = 1, 2, \dots, n$  we have:

$$\{x_i, y_i\}, i = 1, \dots, n, y_i \in \{-1, +1\}, x_i \in R^d$$

$y_i$  shows the binary nature of the classifier with either 1 or -1, representing the class of  $x_i$ . The  $R^d$  is a  $d$ -dimensional vector space over the real numbers. The decision boundary of a linear SVM classifier is as follows:

$$w^T x + b = 0$$

where  $w$  and  $b$  indicate a weight vector and bias, respectively. There are different linear separators, though SVM targets the one with maximum-margin hyperplane from any data point. The linear classifier is as follows:

$$f(x) = \text{sign}(w^T x + b)$$

The main goal is to find the best  $w$  and  $b$  which can maximize the geometric margin ( $\frac{2}{\|w\|}$ ), with linear constraints  $y_i(w^T x_i + b) \geq 1$  for all  $(x_i, y_i)$ . This optimization problem can be defined as a minimization problem as follows:

$$\min_{w,b} \left( \frac{1}{2} \|w\|^2 \right)$$

$$\text{s.t. } y_i(w^T x_i + b) \geq 1, i = 1, \dots, n$$

To solve this problem, the optimization function is transformed into the Lagrangian dual with the Karush–Kuhn–Tucker (KKT) conditions so that the Lagrange multiplier vector  $\alpha_i$  is linked with each inequality of the constraints as:

$$\max_{\alpha \geq 0} \min_{w,b} \left\{ \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \alpha_i [y_i(w^T x_i + b) - 1] \right\},$$

$$\max_{\alpha \geq 0} \left\{ \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j \right\}, \text{ s.t. } \sum_{i=1}^n \alpha_i y_i = 0$$

Thus, the optimal linear classification function is obtained as below, where  $n_{sv}$  denotes the number of support vectors. Different studies show that SVM could provide an efficient non-linear classification and provide very promising results [48,49].  $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$  is the kernel function which provides the inner product value of  $x_i$  and  $x_j$  in the feature space.

$$f(x) = \text{sign}(w^T x + b) = \text{sign}\left(\sum_{i,j=1}^{n_{sv}} \alpha_i y_i x_i^T x_j + b\right)$$

$$f(x) = \text{sign}\left(\sum_{i,j=1}^{n_{sv}} \alpha_i y_i K(x_i, x_j) + b\right)$$

The most often used kernels in SVM are as described in Table A2. The type of the kernel function that transforms data from input space to a higher dimensional feature space has direct impact on the performance of the SVM classifier. Although there exist no well-defined rules for selecting the kernel type [46], here three well-known kernel functions are applied in SVM with Error Correcting Output Codes (ECOC) with One-Versus-One (OVO) technique to evaluate our multi-classification problem. ECOC breaks the multiclass task into a number of binary classification tasks which are then combined to output the final result [50]. The OVO coding design exhausts all combinations of class pair assignments. Therefore, if we have  $K$  distinct classes, the number of learners is  $\frac{k(k-1)}{2}$ . For each binary learner, one class is positive, another is negative, and the rest are ignored.

**Table A2.** The applied kernels in SVM.

| Kernel        | Formula     | Kernel            | Formula             | Kernel                             | Formula   |
|---------------|-------------|-------------------|---------------------|------------------------------------|---|
| <b>Linear</b> | $x_i^T x_j$ | <b>Polynomial</b> | $(x_i^T x_j + 1)^d$ | <b>Radial basis function (RBF)</b> | $\exp\left(-\frac{\ x_i - x_j\ ^2}{2\sigma^2}\right)$ |

**Table A3.** The distance metrics in KNN.

| Distance Metric   | Description                               | Distance Metric               | Description   | Distance Metric    | Description   |
|-------------------|---|-------------------------------|---|--------------------|---|
| <b>Euclidean</b>  | $\sqrt{\sum_{i=1}^n (x_{si} - y_{ti})^2}$ | <b>Standardized Euclidean</b> | $\sqrt{\sum_{i=1}^n \frac{(x_{si} - \bar{x}_s)^2}{s_i^2}}$ $s_i$ is the standard deviation of the $x_{si}$ and $y_{ti}$ over the sample set | <b>Correlation</b> | $1 - \frac{(x_s - \bar{x}_s)(y_t - \bar{y}_t)'}{\sqrt{(x_s - \bar{x}_s)(x_s - \bar{x}_s)'} \sqrt{(y_t - \bar{y}_t)(y_t - \bar{y}_t)'}}$<br>$\bar{x}_s = \frac{1}{n} \sum_i x_{si}$ and $\bar{y}_s = \frac{1}{n} \sum_i y_{ti}$                      |
| <b>City Block</b> | $\sum_{i=1}^n  x_{si} - y_{ti} $          | <b>Minkowski</b>              | $\sqrt[p]{\sum_{i=1}^n  x_{si} - y_{ti} ^p}$ In this work, $p = 3$  | <b>Mahalanobis</b> | $\sqrt{(x_{si} - y_{ti}) C^{-1} (x_{si} - y_{ti})'}$ $C$ is the covariance matrix   |
| <b>Chebychev</b>  | $\max_i \{ x_{si} - y_{ti}  \}$           | <b>Cosine</b>                 | $1 - \frac{x_s y_t'}{\sqrt{(x_s x_t')(y_t y_t')}}$  | <b>Spearman</b>    | $1 - \frac{(r_s - \bar{r}_s)(r_t - \bar{r}_t)'}{\sqrt{(r_s - \bar{r}_s)(r_s - \bar{r}_s)'} \sqrt{(r_t - \bar{r}_t)(r_t - \bar{r}_t)'}} r_{si}$<br>is the rank of $x_{si}$ over $x_s$ . If any $x_s$ values are tied, their average rank is computed |

#### Appendix A.4. K-Nearest Neighbors

K-Nearest Neighbor (KNN) is based on a neighborhood majority voting scheme and assigns the new instance to the most common class amongst its  $K$  nearest. Simplicity and runtime are the main advantages of this method which used in several research works [15,26,51]. There are different metrics to determine the distance  $d(x_s, y_t)$  between two vectors  $x_s$  and  $y_t$ . Table A3 describes the methods used in this study. The three applied distance weights are equal (no weighting), inverse ( $1/d$ ) and squared inverse ( $1/d^2$ ). If multiple classes have the same smallest cost, the smallest index, the class with the nearest neighbor or a random tiebreaker among tied groups is used. Regarding selection of  $k$ , larger value may improve performance and reduce the effect of noise on the classification, but makes boundaries between classes are less distinct [52]. In addition, setting  $k$  to a too large value may destroy locality, and as a result, KNN looks at samples that are not neighbors. They are different techniques to select and find a good  $k$  [53,54]. Here, we consider three values 1, 10 and 100 for  $k$ ; therefore, in total we run 243 KNNs with different settings.

#### Appendix A.5. Ensemble Methods

Generally, ensemble classifier refers to a combination of different classifiers that are cooperatively trained on data set and then classify new data by taking a weighted vote of their predictions to obtain better predictive performance. Indeed, within the sensor-based recognition domain, different studies [6,15,20,25,26] report where an ensemble method outperformed a range of other classification models. Bagging, as named from the phrase “bootstrap aggregating”, is used to improve results of classification algorithms and help to avoid overfitting [55]. This ensemble method constructs bootstrap samples by repetitively resampling training instances with replacement. A sequence of classifiers  $c_{1:b}$  ( $b = 10, 30, 50$ ) in respect to variation of the training set is created by the bagging method. The prediction of a compound classifier, derived from the combinations of  $c_{1:b}$ , is given as:

$$c(d_i) = \text{sign} \left( \sum_{m=1}^b \alpha_m c_m(d_i) \right)$$

The above formula can be interpreted as to classify an example  $d_i$  to the class with majority voting, and  $\alpha$  should be chosen so that more accurate classifiers have stronger impact on the final prediction than less accurate classifiers [56]. More details about the theory of classifier voting can be found in [57].

Another approach is boosting which attempts to turn a weak learner into a strong learner by gradually adapting how models are made. Each new model added to the ensemble is biased to take more notice to training instances that earlier models misclassified [45]. AdaBoost.M2 is a very prevalent boosting algorithm for multi-class classification. The algorithm trains learners sequentially and requires the weak learner to output an array of confidences associated with each possible labeling of an example. For every learner with index  $t$ , AdaBoost.M2 computes the weighted pseudo-loss for  $N$  observations and  $k$  classes as below [45]:

$$\varepsilon_t = \frac{1}{2} \sum_{n=1}^N \sum_{k \neq y_n} d_{n,k}^{(t)} (1 - h_t(x_n, y_n) + h_t(x_n, k))$$

where  $h_t(x_n, k)$  and  $d_{n,k}^{(t)}$  are the confidence of prediction by learner and observation weights at step  $t$  for class  $k$ , respectively. The second sum is over all classes other than  $y_n$  that is the true class. For more details, the reader is referred to [58].

RUSBoost is designed to improve the performance of models trained on skewed data. It combines data sampling and boosting, providing an effective method at classifying imbalanced data. It applies Random Under Sampling (RUS), a method that randomly takes out examples from the majority class for each weak learner in the ensemble until a preferred class distribution is found. If the smallest class has  $N$  training instances, classes with more instances are under sampled by taking only  $N$  observations. For reweighting and constructing the ensemble, it follows the procedure in AdaBoost.M2 [59].

Random subspace ensembles (Subspace) is similar to bagging except that the features are randomly sampled. Thus, subspace ensembles have the advantage of less memory and computations than ensembles with all features resulting in considerably shorter model training times. To train a weak learner, this technique selects a random set of  $m$  predictors (in this study,  $m = 12$ ) from the  $d$  possible values without replacement. In our study, it repeats this procedure until there are 10, 30 or 50 weak learners. Finally, it takes an average of the score prediction of the weak learners, and classifies the observation with the maximum mean score [60].

In this manuscript, the boosting and bagging algorithms are based on tree learners, and the subspace has been applied to discriminant analysis and k-nearest neighbor learners.

#### Appendix A.6. Naïve Bayes

Naïve Bayes (NB) is a powerful probabilistic classifier employing a simplified version of Bayes formula to decide on a class of a new instance [61]. In activity recognition, NB proved to perform well in the previous studies [20,62]. The following equation shows the Naïve Bayes under assumption of feature independence, though the assumption is usually violated in practice.

$$P(l|f_1 \dots f_n) = \frac{p(l) \prod_{i=1}^n p(f_i|l)}{p(f_1 \dots f_n)}$$

where  $l$  represents labels/classes ( $l = 1, 2 \dots L$ ) and  $f_i$  is a feature vector. The denominator of the right side of the equation is a constant, and  $p(l)$  is a prior. The posterior probability  $P(l|f_1 \dots f_n)$  is determined by the likelihood  $\prod_{i=1}^n p(f_i|l)$  and  $p(f_1 \dots f_n)$  is the joint density of the predictors, so,  $p(f_1 \dots f_n) = \sum_{l=1}^L p(l) \prod_{i=1}^n p(f_i|l)$ .

**Table A4.** The applied kernel smoother types in NB.

| Kernel Type              | Formula                              | Kernel Type         | Formula                          |
|--------------------------|--------------------------------------|---------------------|----------------------------------|
| <b>Uniform</b>           | $0.5 I\{ x  \leq 1\}$                | <b>Epanechnikov</b> | $0.75 (1 - x^2) I\{ x  \leq 1\}$ |
| <b>Normal (Gaussian)</b> | $\frac{1}{\sqrt{2\pi}} e^{-0.5 x^2}$ | <b>Triangular</b>   | $(1 -  x ) I\{ x  \leq 1\}$      |

The Naïve Bayes classifier combines the independent feature mode with a decision rule. The common rule is known as the maximum a posteriori or MAP decision rule.

$$\arg \max_l P(l|f_1 \dots f_n) = \arg \max_l \prod_{i=1}^n p(f_i|l)$$

A typical assumption when dealing with data stream is that continuous values associated with each class are distributed according to a Normal (Gaussian) distribution. However, to alleviate this assumption, NB classifier computes a separate kernel density estimate for each class according to its training data [45]. There exists a large range of kernels that can be exploited for the kernel density estimate. Table A4 shows the kernel smoother types we applied in this study.

#### Appendix A.7. Neural Network

Artificial Neural Network (NN) is generally presented as a system of interconnected neurons that are capable of machine learning. The basic processing unit of an NN is called perceptron and is a decision making unit with several inputs and a single output. The input neuron  $p_i$  is weighted with an appropriate  $w_i$ . The perceptron sums the dot product of weights and inputs vectors, and adds a bias  $b$ . The obtained total signal will be transformed by a function which not only can be linear, but is most often a nonlinear transformation (e.g., log-sigmoid and tan-sigmoid) [63]. This process is summarized as:

$$a = f \left( b + \sum_{j=1}^i w_j p_j \right)$$

Feedforward neural networks are one of the most broadly used models in many real-world scientific problems. The network is divided into layers; therefore, it can learn nonlinear relationships between input and output vectors with nonlinear transfer functions. In the input layer, the nodes pass the values to the neurons or hidden units placed in the subsequent layer, which is called hidden layer. In this paper, we considered 10, 20, and 40 hidden neurons. The final layer is the output layer that depends upon the number of class labels in the classification problem [64].

In training the network, its parameters are adjusted incrementally until difference between the output units and the target values is minimized. Resilient backpropagation [65], scaled conjugate gradient [66] and Levenberg–Marquardt backpropagation are the most well-known network training algorithms. For example, Levenberg–Marquardt optimization uses the Hessian matrix approximation,  $J^T J$ , in the following Newton-like update [45]:

$$w_i(t+1) = w_i(t) - [J^T J + \mu I]^{-1} J^T e$$

where Jacobian matrix  $J$  holds the first derivatives of the network errors in respect of the weights and biases.  $\mu$  stands for an adjustment factor,  $e$  for a vector of network errors and  $I$  for the identity matrix [63,67]. Resilient backpropagation network training algorithm updates weight and bias values according to the algorithm explained in [65]. In this method, the magnitude of the derivative has no influence on the weight update and only the sign of the derivative can define the direction of the weight update.

Therefore, in total, there are 293 different classifiers with different settings, as listed in Figure A1.

|    |                              |     |                                 |     |                                 |     |  |
|----|------------------------------|-----|---------------------------------|-----|---------------------------------|-----|--|
| 1  | DT_SC(gdi)MS(4)              | 75  | KNN_BT(SM)D(M)EX(3)DW(EQ)N(1)   | 149 | KNN_BT(NE)D(MA)DW(EQ)N(100)     | 223 | KNN_BT(RA)D(EU)DW(IN)N(10)             |
| 2  | DT_SC(gdi)MS(20)             | 76  | KNN_BT(SM)D(M)EX(3)DW(EQ)N(10)  | 150 | KNN_BT(NE)D(MA)DW(IN)N(1)       | 224 | KNN_BT(RA)D(EU)DW(IN)N(100)            |
| 3  | DT_SC(gdi)MS(100)            | 77  | KNN_BT(SM)D(M)EX(3)DW(EQ)N(100) | 151 | KNN_BT(NE)D(MA)DW(IN)N(10)      | 225 | KNN_BT(RA)D(EU)DW(SJ)N(1)              |
| 4  | DT_SC(towing)MS(4)           | 78  | KNN_BT(SM)D(M)EX(3)DW(IN)N(1)   | 152 | KNN_BT(NE)D(MA)DW(IN)N(100)     | 226 | KNN_BT(RA)D(EU)DW(SJ)N(10)             |
| 5  | DT_SC(towing)MS(20)          | 79  | KNN_BT(SM)D(M)EX(3)DW(IN)N(10)  | 153 | KNN_BT(NE)D(MA)DW(SJ)N(1)       | 227 | KNN_BT(RA)D(EU)DW(SJ)N(100)            |
| 6  | DT_SC(towing)MS(100)         | 80  | KNN_BT(SM)D(M)EX(3)DW(IN)N(100) | 154 | KNN_BT(NE)D(MA)DW(SJ)N(10)      | 228 | KNN_BT(RA)D(MA)DW(EQ)N(1)              |
| 7  | DT_SC(deviance)MS(4)         | 81  | KNN_BT(SM)D(M)EX(3)DW(SJ)N(1)   | 155 | KNN_BT(NE)D(MA)DW(SJ)N(100)     | 229 | KNN_BT(RA)D(MA)DW(EQ)N(10)             |
| 8  | DT_SC(deviance)MS(20)        | 82  | KNN_BT(SM)D(M)EX(3)DW(SJ)N(10)  | 156 | KNN_BT(NE)D(M)EX(3)DW(EQ)N(1)   | 230 | KNN_BT(RA)D(MA)DW(EQ)N(100)            |
| 9  | DT_SC(deviance)MS(100)       | 83  | KNN_BT(SM)D(M)EX(3)DW(SJ)N(100) | 157 | KNN_BT(NE)D(M)EX(3)DW(EQ)N(10)  | 231 | KNN_BT(RA)D(MA)DW(IN)N(1)              |
| 10 | DA_T(linear)                 | 84  | KNN_BT(SM)D(M)EX(3)DW(EQ)N(1)   | 158 | KNN_BT(NE)D(M)EX(3)DW(EQ)N(100) | 232 | KNN_BT(RA)D(MA)DW(IN)N(10)             |
| 11 | DA_T(diagLinear)             | 85  | KNN_BT(SM)D(SE)DW(EQ)N(10)      | 159 | KNN_BT(NE)D(M)EX(3)DW(IN)N(1)   | 233 | KNN_BT(RA)D(MA)DW(IN)N(100)            |
| 12 | DA_T(diagQuadratic)          | 86  | KNN_BT(SM)D(SE)DW(EQ)N(100)     | 160 | KNN_BT(NE)D(M)EX(3)DW(IN)N(10)  | 234 | KNN_BT(RA)D(MA)DW(SJ)N(1)              |
| 13 | DA_T(pseudoLinear)           | 87  | KNN_BT(SM)D(SE)DW(IN)N(1)       | 161 | KNN_BT(NE)D(M)EX(3)DW(IN)N(100) | 235 | KNN_BT(RA)D(MA)DW(SJ)N(10)             |
| 14 | DA_T(pseudoQuadratic)        | 88  | KNN_BT(SM)D(SE)DW(IN)N(10)      | 162 | KNN_BT(NE)D(M)EX(3)DW(SD)N(1)   | 236 | KNN_BT(RA)D(MA)DW(SJ)N(100)            |
| 15 | SVM_KF(linear)(C)OO KS(auto) | 89  | KNN_BT(SM)D(SE)DW(IN)N(100)     | 163 | KNN_BT(NE)D(M)EX(3)DW(SD)N(10)  | 237 | KNN_BT(RA)D(M)EX(3)DW(EQ)N(1)          |
| 16 | SVM_KF(P)(C)OO PO(2)KS(auto) | 90  | KNN_BT(SM)D(SE)DW(SJ)N(1)       | 164 | KNN_BT(NE)D(M)EX(3)DW(SJ)N(100) | 238 | KNN_BT(RA)D(M)EX(3)DW(EQ)N(10)         |
| 17 | SVM_KF(P)(C)OO PO(3)KS(auto) | 91  | KNN_BT(SM)D(SE)DW(SJ)N(10)      | 165 | KNN_BT(NE)D(SE)DW(EQ)N(1)       | 239 | KNN_BT(RA)D(M)EX(3)DW(EQ)N(100)        |
| 18 | SVM_KF(GA)(C)OO KS(1,2)      | 92  | KNN_BT(SM)D(SE)DW(SJ)N(100)     | 166 | KNN_BT(NE)D(SE)DW(EQ)N(10)      | 240 | KNN_BT(RA)D(M)EX(3)DW(IN)N(1)          |
| 19 | SVM_KF(GA)(C)OO KS(4,9)      | 93  | KNN_BT(SM)D(SP)DW(EQ)N(1)       | 167 | KNN_BT(NE)D(SP)DW(EQ)N(100)     | 241 | KNN_BT(RA)D(M)EX(3)DW(IN)N(10)         |
| 20 | SVM_KF(GA)(C)OO KS(20)       | 94  | KNN_BT(SM)D(SP)DW(EQ)N(10)      | 168 | KNN_BT(NE)D(SP)DW(IN)N(10)      | 242 | KNN_BT(RA)D(M)EX(3)DW(IN)N(100)        |
| 21 | KNN_BT(SM)D(CB)DW(EQ)N(1)    | 95  | KNN_BT(SM)D(SP)DW(EQ)N(100)     | 169 | KNN_BT(NE)D(SP)DW(EQ)N(10)      | 243 | KNN_BT(RA)D(M)EX(3)DW(SJ)N(1)          |
| 22 | KNN_BT(SM)D(CB)DW(EQ)N(10)   | 96  | KNN_BT(SM)D(SP)DW(IN)N(1)       | 170 | KNN_BT(NE)D(SE)DW(IN)N(100)     | 244 | KNN_BT(RA)D(M)EX(3)DW(SJ)N(10)         |
| 23 | KNN_BT(SM)D(CB)DW(EQ)N(100)  | 97  | KNN_BT(SM)D(SP)DW(IN)N(10)      | 171 | KNN_BT(NE)D(SE)DW(SJ)N(1)       | 245 | KNN_BT(RA)D(M)EX(3)DW(SJ)N(100)        |
| 24 | KNN_BT(SM)D(CB)DW(IN)N(1)    | 98  | KNN_BT(SM)D(SP)DW(IN)N(100)     | 172 | KNN_BT(NE)D(SE)DW(SJ)N(10)      | 246 | KNN_BT(RA)D(SE)DW(EQ)N(1)              |
| 25 | KNN_BT(SM)D(CB)DW(IN)N(10)   | 99  | KNN_BT(SM)D(SP)DW(SJ)N(1)       | 173 | KNN_BT(NE)D(SE)DW(SJ)N(100)     | 247 | KNN_BT(RA)D(SE)DW(EQ)N(10)             |
| 26 | KNN_BT(SM)D(CB)DW(IN)N(100)  | 100 | KNN_BT(SM)D(SP)DW(SJ)N(10)      | 174 | KNN_BT(NE)D(SP)DW(EQ)N(1)       | 248 | KNN_BT(RA)D(SE)DW(EQ)N(100)            |
| 27 | KNN_BT(SM)D(CB)DW(SJ)N(1)    | 101 | KNN_BT(SM)D(SP)DW(SJ)N(100)     | 175 | KNN_BT(NE)D(SP)DW(EQ)N(10)      | 249 | KNN_BT(RA)D(SE)DW(IN)N(1)              |
| 28 | KNN_BT(SM)D(CB)DW(EQ)N(10)   | 102 | KNN_BT(NE)D(CB)DW(EQ)N(10)      | 176 | KNN_BT(NE)D(SP)DW(EQ)N(100)     | 250 | KNN_BT(RA)D(SE)DW(IN)N(10)             |
| 29 | KNN_BT(SM)D(CB)DW(SJ)N(100)  | 103 | KNN_BT(NE)D(CB)DW(EQ)N(10)      | 177 | KNN_BT(NE)D(SP)DW(IN)N(1)       | 251 | KNN_BT(RA)D(SE)DW(IN)N(100)            |
| 30 | KNN_BT(SM)D(CC)DW(EQ)N(1)    | 104 | KNN_BT(NE)D(CB)DW(EQ)N(100)     | 178 | KNN_BT(NE)D(SP)DW(IN)N(10)      | 252 | KNN_BT(RA)D(SE)DW(SJ)N(1)              |
| 31 | KNN_BT(SM)D(CC)DW(EQ)N(10)   | 105 | KNN_BT(NE)D(CB)DW(IN)N(1)       | 179 | KNN_BT(NE)D(SP)DW(IN)N(10)      | 253 | KNN_BT(RA)D(SE)DW(SJ)N(10)             |
| 32 | KNN_BT(SM)D(CC)DW(EQ)N(100)  | 106 | KNN_BT(NE)D(CB)DW(IN)N(10)      | 180 | KNN_BT(NE)D(SP)DW(SJ)N(1)       | 254 | KNN_BT(RA)D(SE)DW(SJ)N(100)            |
| 33 | KNN_BT(SM)D(CC)DW(IN)N(1)    | 107 | KNN_BT(NE)D(CB)DW(IN)N(100)     | 181 | KNN_BT(NE)D(SP)DW(SJ)N(10)      | 255 | KNN_BT(RA)D(SP)DW(EQ)N(1)              |
| 34 | KNN_BT(SM)D(CC)DW(IN)N(10)   | 108 | KNN_BT(NE)D(CB)DW(SJ)N(1)       | 182 | KNN_BT(NE)D(SP)DW(SJ)N(100)     | 256 | KNN_BT(RA)D(SP)DW(EQ)N(10)             |
| 35 | KNN_BT(SM)D(CC)DW(IN)N(100)  | 109 | KNN_BT(NE)D(CB)DW(SJ)N(10)      | 183 | KNN_BT(RA)D(CB)DW(EQ)N(1)       | 257 | KNN_BT(RA)D(SP)DW(EQ)N(100)            |
| 36 | KNN_BT(SM)D(CC)DW(SJ)N(1)    | 110 | KNN_BT(NE)D(CB)DW(SJ)N(100)     | 184 | KNN_BT(RA)D(CB)DW(EQ)N(10)      | 258 | KNN_BT(RA)D(SP)DW(IN)N(1)              |
| 37 | KNN_BT(SM)D(CC)DW(SJ)N(10)   | 111 | KNN_BT(NE)D(CC)DW(EQ)N(1)       | 185 | KNN_BT(RA)D(CB)DW(EQ)N(100)     | 259 | KNN_BT(RA)D(SP)DW(IN)N(10)             |
| 38 | KNN_BT(SM)D(CC)DW(SJ)N(100)  | 112 | KNN_BT(NE)D(CC)DW(EQ)N(10)      | 186 | KNN_BT(RA)D(CB)DW(IN)N(1)       | 260 | KNN_BT(RA)D(SP)DW(IN)N(100)            |
| 39 | KNN_BT(SM)D(CP)DW(EQ)N(1)    | 113 | KNN_BT(NE)D(CC)DW(EQ)N(100)     | 187 | KNN_BT(RA)D(CB)DW(IN)N(10)      | 261 | KNN_BT(RA)D(SP)DW(SJ)N(1)              |
| 40 | KNN_BT(SM)D(CP)DW(EQ)N(10)   | 114 | KNN_BT(NE)D(CC)DW(IN)N(1)       | 188 | KNN_BT(RA)D(CB)DW(IN)N(100)     | 262 | KNN_BT(RA)D(SP)DW(SJ)N(10)             |
| 41 | KNN_BT(SM)D(CP)DW(EQ)N(100)  | 115 | KNN_BT(NE)D(CC)DW(IN)N(10)      | 189 | KNN_BT(RA)D(CB)DW(SJ)N(1)       | 263 | KNN_BT(RA)D(SP)DW(SJ)N(100)            |
| 42 | KNN_BT(SM)D(CP)DW(IN)N(1)    | 116 | KNN_BT(NE)D(CC)DW(IN)N(100)     | 190 | KNN_BT(RA)D(CB)DW(SJ)N(10)      | 264 | EL(Discriminant)Method(Subspace)NL(10) |
| 43 | KNN_BT(SM)D(CP)DW(IN)N(10)   | 117 | KNN_BT(NE)D(CC)DW(SJ)N(1)       | 191 | KNN_BT(RA)D(CB)DW(SJ)N(100)     | 265 | EL(Discriminant)Method(Subspace)NL(30) |
| 44 | KNN_BT(SM)D(CP)DW(IN)N(100)  | 118 | KNN_BT(NE)D(CC)DW(SJ)N(10)      | 192 | KNN_BT(RA)D(CC)DW(EQ)N(1)       | 266 | EL(Discriminant)Method(Subspace)NL(50) |
| 45 | KNN_BT(SM)D(CP)DW(SJ)N(1)    | 119 | KNN_BT(NE)D(CC)DW(SJ)N(100)     | 193 | KNN_BT(RA)D(CC)DW(EQ)N(10)      | 267 | EL(KNN)Method(Subspace)NL(10)          |
| 46 | KNN_BT(SM)D(CP)DW(SJ)N(10)   | 120 | KNN_BT(NE)D(CP)DW(EQ)N(1)       | 194 | KNN_BT(RA)D(CC)DW(EQ)N(100)     | 268 | EL(KNN)Method(Subspace)NL(30)          |
| 47 | KNN_BT(SM)D(CO)DW(SJ)N(100)  | 121 | KNN_BT(NE)D(CO)DW(EQ)N(10)      | 195 | KNN_BT(RA)D(CC)DW(IN)N(1)       | 269 | EL(KNN)Method(Subspace)NL(50)          |
| 48 | KNN_BT(SM)D(CS)DW(EQ)N(1)    | 122 | KNN_BT(NE)D(CS)DW(EQ)N(100)     | 196 | KNN_BT(RA)D(CC)DW(IN)N(10)      | 270 | EL(Tree)Method(AdaBoostM2)NL(10)       |
| 49 | KNN_BT(SM)D(CS)DW(EQ)N(10)   | 123 | KNN_BT(NE)D(CS)DW(IN)N(1)       | 197 | KNN_BT(RA)D(CC)DW(IN)N(100)     | 271 | EL(Tree)Method(AdaBoostM2)NL(30)       |
| 50 | KNN_BT(SM)D(CS)DW(EQ)N(100)  | 124 | KNN_BT(NE)D(CS)DW(IN)N(10)      | 198 | KNN_BT(RA)D(CC)DW(SJ)N(1)       | 272 | EL(Tree)Method(AdaBoostM2)NL(50)       |
| 51 | KNN_BT(SM)D(CS)DW(IN)N(1)    | 125 | KNN_BT(NE)D(CS)DW(IN)N(100)     | 199 | KNN_BT(RA)D(CC)DW(SJ)N(10)      | 273 | EL(Tree)Method(RUSBoost)NL(10)         |
| 52 | KNN_BT(SM)D(CS)DW(IN)N(10)   | 126 | KNN_BT(NE)D(CS)DW(SJ)N(1)       | 200 | KNN_BT(RA)D(CC)DW(SJ)N(100)     | 274 | EL(Tree)Method(RUSBoost)NL(30)         |
| 53 | KNN_BT(SM)D(CS)DW(IN)N(100)  | 127 | KNN_BT(NE)D(CS)DW(SJ)N(10)      | 201 | KNN_BT(RA)D(CC)DW(EQ)N(1)       | 275 | EL(Tree)Method(RUSBoost)NL(50)         |
| 54 | KNN_BT(SM)D(CS)DW(SJ)N(1)    | 128 | KNN_BT(NE)D(CS)DW(SJ)N(100)     | 202 | KNN_BT(RA)D(CD)DW(EQ)N(10)      | 276 | EL(Tree)Method(Bag)NL(10)              |
| 55 | KNN_BT(SM)D(CS)DW(SJ)N(10)   | 129 | KNN_BT(NE)D(CS)DW(EQ)N(1)       | 203 | KNN_BT(RA)D(CD)DW(EQ)N(100)     | 277 | EL(Tree)Method(Bag)NL(30)              |
| 56 | KNN_BT(SM)D(CS)DW(SJ)N(100)  | 130 | KNN_BT(NE)D(CS)DW(EQ)N(10)      | 204 | KNN_BT(RA)D(CD)DW(IN)N(1)       | 278 | EL(Tree)Method(Bag)NL(50)              |
| 57 | KNN_BT(SM)D(EU)DW(EQ)N(1)    | 131 | KNN_BT(NE)D(CS)DW(EQ)N(100)     | 205 | KNN_BT(RA)D(CD)DW(IN)N(10)      | 279 | NB_DI(kernel)Kernel(normal)            |
| 58 | KNN_BT(SM)D(EU)DW(EQ)N(10)   | 132 | KNN_BT(NE)D(CS)DW(IN)N(1)       | 206 | KNN_BT(RA)D(CD)DW(IN)N(100)     | 280 | NB_DI(kernel)Kernel(box)               |
| 59 | KNN_BT(SM)D(EU)DW(EQ)N(100)  | 133 | KNN_BT(NE)D(CS)DW(IN)N(10)      | 207 | KNN_BT(RA)D(CD)DW(SJ)N(1)       | 281 | NB_DI(kernel)Kernel(epanechnikov)      |
| 60 | KNN_BT(SM)D(EU)DW(IN)N(1)    | 134 | KNN_BT(NE)D(CS)DW(IN)N(100)     | 208 | KNN_BT(RA)D(CD)DW(SJ)N(10)      | 282 | NB_DI(kernel)Kernel(triangle)          |
| 61 | KNN_BT(SM)D(EU)DW(IN)N(10)   | 135 | KNN_BT(NE)D(CS)DW(SJ)N(1)       | 209 | KNN_BT(RA)D(CD)DW(SJ)N(100)     | 283 | NN_pattNet_TF(SCG)HL(10)               |
| 62 | KNN_BT(SM)D(EU)DW(IN)N(100)  | 136 | KNN_BT(NE)D(CS)DW(SJ)N(10)      | 210 | KNN_BT(RA)D(CS)DW(EQ)N(1)       | 284 | NN_pattNet_TF(SCG)HL(20)               |
| 63 | KNN_BT(SM)D(EU)DW(SJ)N(1)    | 137 | KNN_BT(NE)D(CS)DW(SJ)N(100)     | 211 | KNN_BT(RA)D(CS)DW(EQ)N(10)      | 285 | NN_pattNet_TF(SCG)HL(40)               |
| 64 | KNN_BT(SM)D(EU)DW(SJ)N(10)   | 138 | KNN_BT(NE)D(EU)DW(EQ)N(1)       | 212 | KNN_BT(RA)D(CS)DW(EQ)N(100)     | 286 | NN_pattNet_TF(RP)HL(10)                |
| 65 | KNN_BT(SM)D(EU)DW(SJ)N(100)  | 139 | KNN_BT(NE)D(EU)DW(EQ)N(10)      | 213 | KNN_BT(RA)D(CS)DW(IN)N(1)       | 287 | NN_pattNet_TF(RP)HL(20)                |
| 66 | KNN_BT(SM)D(MA)DW(EQ)N(1)    | 140 | KNN_BT(NE)D(EU)DW(EQ)N(100)     | 214 | KNN_BT(RA)D(CS)DW(IN)N(10)      | 288 | NN_pattNet_TF(RP)HL(40)                |
| 67 | KNN_BT(SM)D(MA)DW(EQ)N(10)   | 141 | KNN_BT(NE)D(EU)DW(IN)N(1)       | 215 | KNN_BT(RA)D(CS)DW(IN)N(100)     | 289 | NN_pattNet_TF(LML)HL(10)               |
| 68 | KNN_BT(SM)D(MA)DW(EQ)N(100)  | 142 | KNN_BT(NE)D(EU)DW(IN)N(10)      | 216 | KNN_BT(RA)D(CS)DW(SJ)N(10)      | 290 | NN_pattNet_TF(LML)HL(20)               |
| 69 | KNN_BT(SM)D(MA)DW(IN)N(1)    | 143 | KNN_BT(NE)D(EU)DW(IN)N(100)     | 217 | KNN_BT(RA)D(CS)DW(SJ)N(10)      | 291 | NN_pattNet_TF(LML)HL(40)               |
| 70 | KNN_BT(SM)D(MA)DW(IN)N(10)   | 144 | KNN_BT(NE)D(EU)DW(SJ)N(1)       | 218 | KNN_BT(RA)D(CS)DW(SJ)N(100)     | 292 | NN_softmax_TF(SCG)                     |
| 71 | KNN_BT(SM)D(MA)DW(IN)N(100)  | 145 | KNN_BT(NE)D(EU)DW(SJ)N(10)      | 219 | KNN_BT(RA)D(EU)DW(EQ)N(1)       | 293 | NN_softmax_TF(LML)                     |
| 72 | KNN_BT(SM)D(MA)DW(SJ)N(1)    | 146 | KNN_BT(NE)D(EU)DW(SJ)N(100)     | 220 | KNN_BT(RA)D(EU)DW(EQ)N(10)      |     |  |
| 73 | KNN_BT(SM)D(MA)DW(SJ)N(10)   | 147 | KNN_BT(NE)D(MA)DW(EQ)N(1)       | 221 | KNN_BT(RA)D(EU)DW(EQ)N(100)     |     |  |
| 74 | KNN_BT(SM)D(MA)DW(SJ)N(100)  | 148 | KNN_BT(NE)D(MA)DW(EQ)N(10)      | 222 | KNN_BT(RA)D(EU)DW(IN)N(1)       |     |  |

**Figure A1.** The list of the classifiers we explored in this paper. The numbers indicate the classifiers IDs.

## References

- Kumar, S.; Nilsen, W.; Pavel, M.; Srivastava, M. Mobile health: Revolutionizing healthcare through transdisciplinary research. *IEEE Comput.* **2013**, *46*, 28–35. [[CrossRef](#)]
- Khusainov, R.; Azzi, D.; Achumba, I.; Bersch, S. Real-Time Human Ambulation, Activity, and Physiological Monitoring: Taxonomy of Issues, Techniques, Applications, Challenges and Limitations. *Sensors* **2013**, *13*, 12852–12902. [[CrossRef](#)] [[PubMed](#)]
- Chen, L.; Hoey, J.; Nugent, C.D.; Cook, D.J.; Yu, Z. Sensor-Based Activity Recognition. *IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev.* **2012**, *42*, 790–808. [[CrossRef](#)]
- He, Z.; Jin, L. Activity recognition from acceleration data using AR model representation and SVM. In Proceedings of the 2008 International Conference on Machine Learning and Cybernetics, Kunming, China, 12–15 July 2008; Volume 4, pp. 2245–2250.

5. Anguita, D.; Ghio, A.; Oneto, L.; Parra, X.; Reyes-Ortiz, J.L. A Public Domain Dataset for Human Activity Recognition Using Smartphones. In Proceedings of the 21th European Symposium on Artificial Neural Network. Computational Intelligence and Machine Learning (ESANN 2013), Bruges, Belgium, 24–26 April 2013.
6. Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. In *Advances in Artificial Intelligence—SBIA 2012, Proceedings of the 21st Brazilian Symposium on Artificial Intelligence, Curitiba, Brazil, 20–25 October 2012*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 52–61.
7. Lara, O.D.; Labrador, M.A. A Survey on Human Activity Recognition using Wearable Sensors. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 1192–1209. [[CrossRef](#)]
8. Banos, O.; Galvez, J.-M.; Damas, M.; Pomares, H.; Rojas, I. Window Size Impact in Human Activity Recognition. *Sensors* **2014**, *14*, 6474–6499. [[CrossRef](#)] [[PubMed](#)]
9. Preece, S.J.; Goulermas, J.Y.; Kenney, L.P.; Howard, D.; Meijer, K.; Crompton, R. Activity identification using body-mounted sensors—A review of classification techniques. *Physiol. Meas.* **2009**, *30*, 4. [[CrossRef](#)] [[PubMed](#)]
10. Igual, R.; Medrano, C.; Plaza, I. A comparison of public datasets for acceleration-based fall detection. *Med. Eng. Phys.* **2015**, *37*, 870–878. [[CrossRef](#)] [[PubMed](#)]
11. Stisen, A.; Blunck, H.; Bhattacharya, S.; Prentow, T.; Kjørgaard, M.; Dey, A.; Sonne, T.; Jensen, M. Smart devices are different: Assessing and mitigating mobile sensing heterogeneities for activity recognition. In Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, Seoul, Korea, 1–4 November 2015; pp. 127–140.
12. Blunck, H.; Bhattacharya, S.; Stisen, A.; Prentow, T.S.; Kjærgaard, M.B.; Dey, A.; Jensen, M.M.; Sonne, T. Activity recognition on smart devices: Dealing with diversity in the wild. *GetMobile* **2016**, *20*, 34–38.
13. Labrador, M.A.; Lara, O.D.; Human, Y. *Activity Recognition: Using Wearable Sensors and Smartphones*; Chapman & Hall/CRC Computer and Information Science Series; CRC Press Book: Boca Raton, FL, USA, 2013.
14. Keogh, E.; Chu, S.; Hart, D.; Pazzani, M. An Online Algorithm for Segmenting Time Series. In Proceedings of the International Conference on Data Mining, San Jose, CA, USA, 29 November–2 December 2001; pp. 289–296.
15. Janidarmian, M.; Radecka, K.; Zilic, Z. Automated diagnosis of knee pathology using sensory data. In Proceedings of the IEEE/EAI International Conference on Wireless Mobile Communication and Healthcare (Mobihealth), Athens, Greece, 3–5 November 2014; pp. 95–98.
16. Krishnan, N.C.; Juillard, C.; Colbry, D.; Panchanathan, S. Recognition of hand movements using wearable accelerometers. *J. Ambient Intell. Smart Environ.* **2009**, *2*, 143–155.
17. Sun, L.; Zhang, D.; Li, B.; Guo, B.; Li, S. Activity Recognition on an Accelerometer Embedded Mobile Phone with Varying Positions and Orientations. In *Ubiquitous Intelligence and Computing; Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6406, pp. 548–562.
18. He, Z.; Jin, L. Activity recognition from acceleration data based on discrete cosine transform and SVM. In Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 2009 (SMC 2009), San Antonio, TX, USA, 11–14 October 2009; pp. 5041–5044.
19. Peterek, T.; Penhaker, M.; Gajdoš, P.; Dohnálek, P. Comparison of classification algorithms for physical activity recognition. In *Innovations in Bio-Inspired Computing and Applications*; Springer: Ostrava, Czech Republic, 2014; pp. 123–131.
20. Ravi, N.; Dandekar, N.; Mysore, P.; Littman, M.L. Activity recognition from accelerometer data. In Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence (IAAI), Pittsburgh, PA, USA, 9–13 July 2005; pp. 1541–1546.
21. Banos, O.; Toth, M.A.; Damas, M.; Pomares, H.; Rojas, I.; Amft, O. A benchmark dataset to evaluate sensor displacement in activity recognition. In Proceedings of the 14th International Conference on Ubiquitous Computing (Ubicomp 2012), Pittsburgh, PA, USA, 5–8 September 2012.
22. Banos, O.; Villalonga, C.; Garcia, R.; Saez, A.; Damas, M.; Holgado, J.A.; Lee, S.; Pomares, H.; Rojas, I. Design, implementation and validation of a novel open framework for agile development of mobile health applications. *BioMed. Eng. OnLine* **2015**, *14*, S6. [[CrossRef](#)] [[PubMed](#)]

23. Shoaib, M.; Scholten, J.; Havinga, P.J.M. Towards physical activity recognition using smartphone sensors. In Proceedings of the 10th IEEE International Conference on Ubiquitous Intelligence and Computing, Vietri sul Mare, Italy, 18–20 December 2013; pp. 80–87.
24. Kwapisz, J.R.; Weiss, G.M.; Moore, S.A. Activity Recognition using Cell Phone Accelerometers. In Proceedings of the Fourth International Workshop on Knowledge Discovery from Sensor Data, Washington, DC, USA, 25 July 2010; pp. 74–82.
25. Shoaib, M.; Bosch, S.; Incel, O.D.; Scholten, H.; Havinga, P.J.M. Fusion of Smartphone Motion Sensors for Physical Activity Recognition. *Sensors* **2014**, *14*, 10146–10176. [CrossRef] [PubMed]
26. Janidarmian, M.; Radecka, K.; Zilic, Z. Analysis of Motion Patterns for Recognition of Human Activities. In Proceedings of the 2015 ACM 5th International Conference on Wireless Mobile Communication and Healthcare (Mobihealth), London, UK, 14–16 October 2015; pp. 68–72.
27. Altun, K.; Barshan, B.; Tunçel, O. Comparative study on classifying human activities with miniature inertial and magnetic sensors. *Pattern Recognit.* **2010**, *43*, 3605–3620. [CrossRef]
28. Liu, S.; Gao, R.X.; John, D.; Staudenmayer, J.; Freedson, P.S. SVM-based multi-sensor fusion for free-living physical activity assessment. In Proceedings of the 33rd Annual International IEEE EMBS Conference, Boston, MA, USA, 30 August–3 September 2011; pp. 3188–3191.
29. Long, X.; Yin, B.; Aarts, R.M. Single-accelerometer-based daily physical activity classification. In Proceedings of the 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Minneapolis, MN, USA, 3–6 September 2009; pp. 6107–6110.
30. Győrbíró, N.; Fábián, Á.; Hományi, G. An activity recognition system for mobile phones. *Mob. Netw. Appl.* **2009**, *14*, 82–91. [CrossRef]
31. Vahdatpour, A.; Amini, N.; Sarrafzadeh, M. On-body device localization for health and medical monitoring applications. In Proceedings of the 2011 IEEE International Conference on Pervasive Computing and Communications (PerCom), Seattle, WA, USA, 21–25 March 2011; pp. 37–44.
32. Saeedi, R.; Purath, J.; Venkatasubramanian, K.; Ghasemzadeh, H. Toward seamless wearable sensing: Automatic on-body sensor localization for physical activity monitoring. In Proceedings of the 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Chicago, IL, USA, 26–30 August 2014; pp. 5385–5388.
33. Bruno, B.; Mastrogiovanni, F.; Sgorbissa, A.; Vernazza, T.; Zaccaria, R. Analysis of human behavior recognition algorithms based on acceleration data. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, 6–10 May 2013; pp. 1602–1607.
34. Casale, P.; Pujol, O.; Radeva, P. Personalization and user verification in wearable systems using biometric walking patterns. *Pers. Ubiquitous Comput.* **2012**, *16*, 563–580. [CrossRef]
35. Zhang, M.; Sawchuk, A.A. USC-HAD: A Daily Activity Dataset for Ubiquitous Activity Recognition Using Wearable Sensors. In Proceedings of the ACM International Conference on Ubiquitous Computing (UbiComp) Workshop on Situation, Activity and Goal Awareness (SAGAware), Pittsburgh, PA, USA, 5–8 September 2012.
36. Yang, A.; Jafari, R.; Sastry, S.; Bajcsy, R. Distributed Recognition of Human Actions Using Wearable Motion Sensor Networks. *J. Ambient Intell. Smart Environ.* **2009**, *1*, 103–115.
37. Frank, K.; Röckl, M.; Vera Nadales, M.J.; Robertson, P.; Pfeifer, T. Comparison of exact static and dynamic Bayesian context inference methods for activity recognition. In Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), Mannheim, Germany, 29 March–2 April 2010; pp. 189–195.
38. Balaprakash, P.; Tiwari, A.; Wild, S.; Jarvis, S.A.; Wright, S.A.; Hammond, S.D. Multi objective optimization of HPC kernels for performance power and energy. In *International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems*; Springer Lecture Notes in Computer Science: Denver, CO, USA, 2014; pp. 239–260.
39. Fekr, A.R.; Janidarmian, M.; Radecka, K.; Zilic, Z. Respiration Disorders Classification with Informative Features for m-Health Applications. *IEEE J. Biomed. Health Inform.* **2016**, *20*, 733–747. [CrossRef] [PubMed]
40. Rokach, L.; Maimon, O. *Data Mining with Decision Trees: Theory and Applications*; World Scientific Publishing Co., Inc.: River Edge, NJ, USA, 2008.
41. Zambon, M.; Lawrence, R.; Bunn, A.; Powell, S. Effect of alternative splitting rules on image processing using classification tree analysis. *Photogramm. Eng. Remote Sens.* **2006**, *72*, 25–30. [CrossRef]

42. Theodoridis, S.; Koutroumbas, K. *Pattern Recognition*, 4th ed.; Elsevier / Academic Press: Cambridge, MA, USA, 2008.
43. Zheng, E.; Chen, B.; Wang, X.; Huang, Y.; Wang, Q. On the Design of a Wearable Multi-sensor System for Recognizing Motion Modes and Sit-to-stand Transition. *Int. J. Adv. Robot. Syst.* **2014**, *11*, 30. [CrossRef]
44. Chen, T.; Mazomenos, E.; Maharatna, K.; Dasmahapatra, S. On the trade-off of accuracy and computational complexity for classifying normal and abnormal ECG in remote CVD monitoring systems. In Proceedings on the IEEE Workshop Signal Processing Systems (SIPS), Quebec City, QC, Canada, 17–19 October 2012; pp. 37–42.
45. Supervised Learning, Statistics Toolbox, MATLAB R2105b documentation. Available online: <https://cn.mathworks.com/login?uri=https%3A%2F%2Fcn.mathworks.com%2Fhelp%2Freleases%2FR2015b%2Findex.html> (accessed on 6 March 2017).
46. Chamasemani, F.F.; Singh, Y.P. Multi-class Support Vector Machine (SVM) Classifiers—An Application in Hypothyroid Detection and Classification. In Proceedings of the 2011 Sixth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), Penang, Malaysia, 27–29 September 2011; pp. 351–356.
47. Fekr, A.R.; Janidarmian, M.; Radecka, K.; Zilic, Z. A Medical Cloud-Based Platform for Respiration Rate Measurement and Hierarchical Classification of Breath Disorders. *Sensors* **2014**, *14*, 11204–11224. [CrossRef] [PubMed]
48. Parera, J.; Angulo, C.; Rodriguez-Molinero, A.; Cabestany, J. User daily activity classification from accelerometry using feature selection and SVM. In *Bio-Inspired Systems: Computational and Ambient Intelligence*; LNCS: Salamanca, Spain, 2009; pp. 1137–1144.
49. He, H.; Garcia, E.A. Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 1263–1284.
50. Dietterich, T.G.; Bakiri, G. Solving multiclass learning problem via error-correcting output codes. *J. Artif. Intell. Res.* **1995**, *2*, 263–286.
51. Foerster, K.; Monteleone, S.; Calatroni, A.; Roggen, D.; Trooster, G. Incremental KNN classifier exploiting correct—Error teacher for activity recognition. In Proceedings of the 9th International Conference on Machine Learning and Applications, Washington, DC, USA, 12–14 December 2010; pp. 445–450.
52. Everitt, B.S.; Landau, S.; Leese, M.; Stahl, D. *Miscellaneous Clustering Methods*, in *Cluster Analysis*, 5th ed.; John Wiley & Sons, Ltd.: Chichester, UK, 2011.
53. Hall, P.; Park, B.U.; Samworth, R.J. Choice of neighbour order in nearest-neighbour classification. *Ann. Stat.* **2008**, *36*, 2135–2152. [CrossRef]
54. Anil, K.G. On optimum choice of k in nearest neighbor classification. *Comput. Stat. Data Anal.* **2006**, *50*, 3113–3123.
55. Breiman, L. *Bagging Predictors*; Technical Report 421; Department of Statistics, University of California at Berkeley: Berkeley, CA, USA, 1994.
56. Machova, K.; Barcak, F.; Bednar, P. A bagging method using decision trees in the role of base classifiers. *Acta Polytech. Hung.* **2006**, *3*, 2.
57. Schapire, R.; Freund, Y. Boosting the margin: A new explanation for the effectiveness of voting methods. *Ann. Stat.* **1998**, *26*, 1651–1686. [CrossRef]
58. Freund, Y.; Schapire, R.E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [CrossRef]
59. Seiffert, C.; Khoshgoftaar, T.; Hulse, J.; Napolitano, A. RUSBoost: Improving classification performance when training data is skewed. In Proceedings of the 19th International Conference on Pattern Recognition, Tampa, FL, USA, 8–11 December 2008; pp. 1–4.
60. Ho, T.K. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.* **1998**, *20*, 832–844.
61. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: Berlin, Germany, 2006.
62. Lester, J.; Choudhury, T.; Kern, N.; Borriello, G.; Hannaford, B. A hybrid discriminative/generative approach for modeling human activities. In Proceedings of the 19th International Joint Conference on Artificial Intelligence, Edinburgh, UK, 30 July–5 August 2005; pp. 1541–1546.
63. Cortina-Puig, M.; Istamboulie, G.; Noguer, T.; Marty, J. Analysis of Pesticide Mixtures Using Intelligent Biosensors. In *Intelligent and Biosensors*; Vernon, S.S., Ed.; InTech: Rijeka, Croatia, 2010; Chapter 10.

64. Sharma, A.; Lee, Y.-D.; Chung, W.-Y. High Accuracy Human Activity Monitoring Using Neural Network. In Proceedings of the Third International Conference on Convergence and Hybrid Information Technology, 2008 (ICCIT '08), Washington, DC, USA, 11–13 November 2008; Volume 1, pp. 430–435.
65. Riedmiller, M.; Braun, H. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In Proceedings of the IEEE International Conference on Neural Network, San Francisco, CA, USA, 28 March–1 April 1993; pp. 586–591.
66. Molier, M. A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. *Neural Netw.* **1993**, *6*, 525–533. [[CrossRef](#)]
67. Hagan, M.T.; Menhaj, M. Training feed-forward networks with the Marquardt algorithm. *IEEE Trans. Neural Netw.* **1994**, *5*, 989–993. [[CrossRef](#)] [[PubMed](#)]



© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).