A Comparison Study of Classifier Algorithms for Cross-Person Physical Activity Recognition

Yago Saez,[*] Alejandro Baldominos, and Pedro Isasi

Octavian Adrian Postolache, Academic Editor, Alex Casson, Academic Editor, and Subhas Chandra Mukhopadhyay, Academic Editor

Department of Computer Science, Universidad Carlos III de Madrid, 28911 Leganés, Spain;

abaldomi@inf.uc3m.es (A.B.); isasi@ia.uc3m.es (P.I.)

[*]Correspondence: yago.saez@uc3m.es

Abstract

Physical activity is widely known to be one of the key elements of a healthy life. The many benefits of physical activity described in the medical literature include weight loss and reductions in the risk factors for chronic diseases. With the recent advances in wearable devices, such as smartwatches or physical activity wristbands, motion tracking sensors are becoming pervasive, which has led to an impressive growth in the amount of physical activity data available and an increasing interest in recognizing which specific activity a user is performing. Moreover, big data and machine learning are now cross-fertilizing each other in an approach called "deep learning", which consists of massive artificial neural networks able to detect complicated patterns from enormous amounts of input data to learn classification models. This work compares various state-of-the-art classification techniques for automatic cross-person activity recognition under different scenarios that vary widely in how much information is available for analysis. We have incorporated deep learning by using Google's TensorFlow framework. The data used in this study were acquired from PAMAP2 (Physical Activity Monitoring in the Ageing Population), a publicly available dataset containing physical activity data. To perform cross-person prediction, we used the leave-one-subject-out (LOSO) cross-validation technique. When working with large training sets, the best classifiers obtain very high average accuracies (e.g., 96% using extra randomized trees). However, when the data volume is drastically reduced (where available data are only 0.001% of the continuous data), deep neural networks performed the best, achieving 60% in overall prediction accuracy. We found that even when working with only approximately 22.67% of the full dataset, we can statistically obtain the same results as when working with the full dataset. This finding enables the design of more energy-efficient devices and facilitates cold starts and big data processing of physical activity records.

1. Introduction

Physical activity recognition has numerous applications in fields, such as medical monitoring, healthy aging, active living and rehabilitation, and so on. For many years, the medical research literature has extensively explored the positive effects of physical activity on health. These positive effects accrue to humans regardless of age. For instance, in the early stages of physical human development, Janssen and Leblanc [1] explored the benefits of physical activity in school-aged children and youth; Merglen et al. [2] showed the advantages of weekly sports practice for adolescent wellbeing; and Biddle and Asare [3] studied the association between mental illnesses and lack of physical activity. Regarding the latter stages of human development, the work by Chodzko-Zajko [4] focused on the benefits of physical activity in healthy aging and concluded that physical activity reduces the risk factors of chronic diseases.

Activity recognition is a field that has gained increasing interest given the availability of commodity hardware devices (i.e., smartphones, wearable devices, etc.). These devices contain a set of sensors, such as accelerometers, gyroscopes, and GPS, that can provide the basic data required to perform activity recognition, as described by [5], who provided a complete overview of the past, present and future of pervasive sensing technologies and their applications for health monitoring.

Proof of the increasing interest in activity monitoring can be found in the large number of inventions and patents in recent years that aim at monitoring and quantifying people's physical activity [6,7] and (more recently) at recognizing the activity being performed [8]. In some cases, these provide customized personal training features [9,10]. In addition, healthcare-related cell phone apps have also become popular in the last few years. Some of the larger cell phone manufacturers have included specific sensor hardware in their cell phones (e.g., the Bosch (Stuttgart, Germany) BMA280 accelerometer or InvenSense (San Jose, CA, USA)'s six-axis MPU-6700).

Physical activity recognition is primarily a classification problem [11,12]. Physical activity recognition systems are not new; there are previous related works that have been developed since more than two decades [13]. However, the arrival of new technologies to wearable sensors and mobile devices has enabled novel applications in different areas, such as healthcare, sport or security, among others, and at this moment, there are many open problems to be tackled. Recent surveys [14,15,16] have reviewed in detail the most relevant previous works conducted in this field.

According to [14], research in human activity recognition falls into seven different categories: (1) selection of attributes and sensors; (2) obtrusiveness; (3) data collection protocol; (4) recognition performance; (5) energy consumption; (6) processing; and (7) flexibility. This article falls mainly into the last category: flexibility to support new users without the need for collecting additional data of the user and re-training the system. This topic, also known as cross-person activity recognition, has been previously addressed by many other works, such as [17,18] and, more recently, [19].

The contributions we want to make with respect to previous works are: (1) to conduct a complete comparison of the most relevant out-of-the-box scikit-learn classifiers in this field and how they compare with other techniques from previous works; (2) to include a deep learning approach into the comparison and analyze how it compares with common machine learning classifiers; and (3) to include an additional study of how these algorithms perform with limited data, which implies less energy consumption needed and addresses the fifth issue described in [14].

Motivations

As far as we know, this is the first work using the scikit-learn library for physical activity recognition. Most of the existing research works are based on WEKA [11,20,21,22], MATLAB [23], or proprietary developments, such as the work presented by [24], among many others. The motivation for using the scikit-learn library in this paper is two-fold. First, this library was implemented by experts; all algorithms are peer reviewed; and it is widely used and has been gaining in popularity in the last few years (see Figure 1). Second, using this library in conjunction with a public dataset (such as PAMAP2 (Physical Activity Monitoring in the Ageing Population) [25]) ensures that all of the experiments conducted in this work can be easily replicated and can be used as a benchmark.
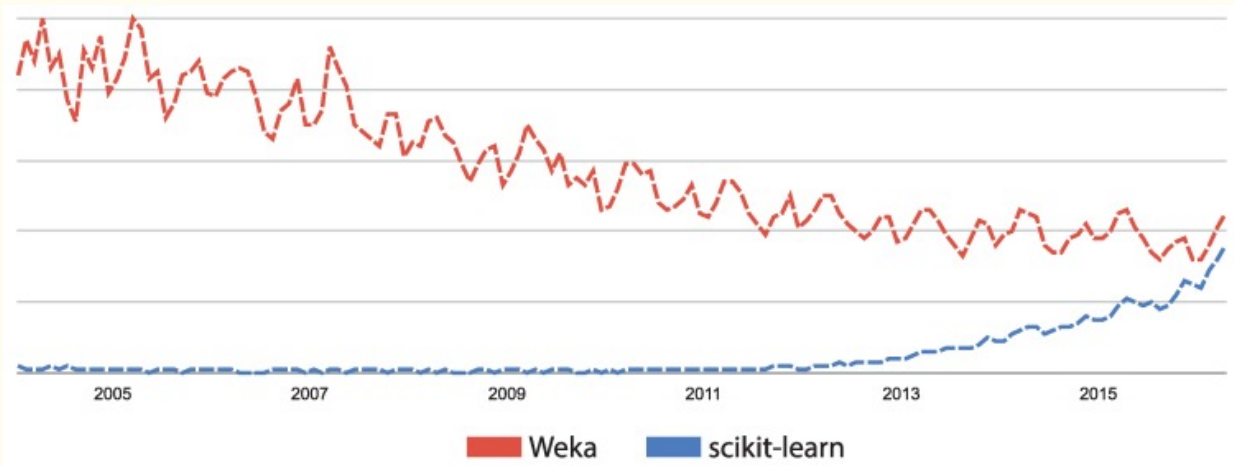
Figure 1

Search interest through time of both WEKA and scikit-learn. Source: Google Trends [37].

Nowadays, one of the trending research lines is applying deep learning techniques to extract features and classify in one step, starting from the raw dataset. This approach has been addressed in a few works; a representative set can be found in [26,27,28,29,30,31]. There is an interesting comparison of different deep learning techniques applied to the PAMAP dataset (among others) in [32]. Results found in that work with deep learning techniques are outperformed by the ones achieved in this work with machine learning classifiers, which suggests that more research and comparison still need to be done. Since we have the comparison of [32], the motivation for including deep learning techniques into our comparison is to analyze how these techniques compare in performance with common state-of-the-art machine learning classification techniques, but when working with the classic approach of signals already transformed into the frequency domain.

Last, but not least, one important research topic when working with sensors embedded in devices running on batteries is how to optimize their energy consumption [33,34,35,36]. In physical activity recognition with wearable devices, there is an important tradeoff between accuracy and power consumption. Just collecting and sending all information gathered from all available sensors through the data network have a great impact on energy consumption. However, if we are able to achieve acceptable classification accuracies (higher than 90%) using much less information coming from sensors, then we could use only a sample of what is available in order to save energy and time. In this work, we have tested with nine different sample sizes, starting from 10% to 0.001%.

2. Methodology

In this paper, we compare classic, state-of-the-art machine learning techniques and deep learning techniques to achieve cross-person prediction of the physical activity performed by a subject. When performing cross-person predictions, the distribution of data is heavily affected by the various users; consequently, the prediction performance degrades when a model trained on one person is used on others.

Most works in the field of human activity recognition follow a so-called activity recognition chain (ARC). This sequence of steps was extensively described by [38] as a general-purpose framework for acquiring data samples and building and evaluating activity recognition systems.

Figure 2 shows the different steps involved in the ARC. These include data acquisition from sensors, signal preprocessing, signal segmentation, feature extraction and training a classification model. In this section, we will thoroughly describe each of these steps.
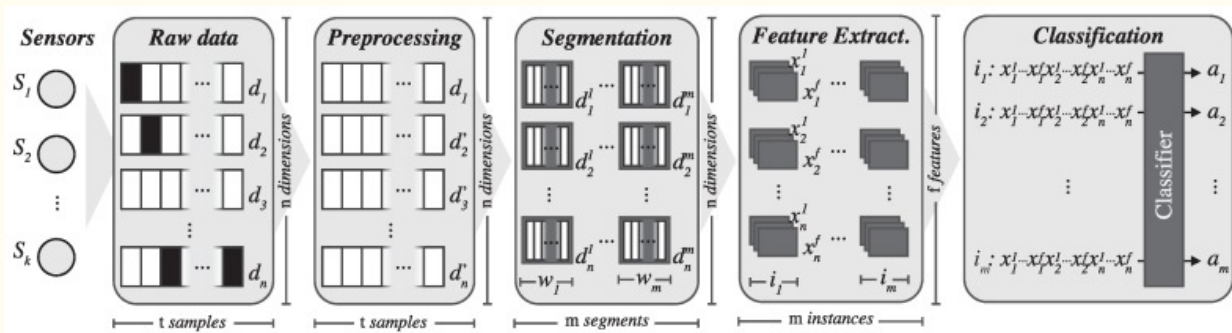

Figure 2

Steps involved in the activity recognition chain (ARC) that range from data collection to classification.

2.1. Data Acquisition

The first stage of the ARC involves data acquisition: the design and development of a process to gather data about the subjects performing physical activities. While these data can be of various types, a frequent approach consists of placing different sensors on subjects' bodies to obtain data about their motions. These devices range from smartphones and wearable devices, such as activity wristbands, to specific sensors attached directly to the body.

Besides the devices themselves, the data acquisition stage must also define a protocol that must be strictly followed to guarantee the validity of the obtained samples and consistency between different subjects.

In this paper, we use the physical activity monitoring dataset PAMAP2, introduced by [24,25,39,40,41,42,43] and publicly available at the University of California, Irvine (UCI) Machine Learning Repository. Selecting this dataset enables future replications of the experiments in this paper by the scientific community.

The PAMAP2 dataset contains labeled information about the physical activity performed by 9 different subjects wearing heart rate monitors and three Colibri wireless inertial measurement units (IMUs). This section provides further detail about the subjects' profiles, the wearable devices and the protocol used to acquire the physical activity data.

2.1.1. Subjects

The PAMAP2 dataset provides physical activity data about 9 different subjects. The subjects who participated in the data acquisition protocol are mainly employees or students at DFKI (German Research Centre for Artificial Intelligence) and consist of 1 female and 8 males, aged from 27.22 ± 3.31 years. Their average body mass index (BMI) ranges from 25.11 ± 2.62 kg·m^{-2}. Further details about each subject are provided in Table 1.

Table 1

Physiological information about the subjects participating in the data acquisition protocol. HR, heart rate.

#	Sex	Age (years)	Height (cm)	Weight (kg)	Resting HR (bpm)	Max HR (bpm)	Dominant Hand
1	Male	27	182	83	75	193	right
2	Female	25	169	78	74	195	right
3	Male	31	187	92	68	189	right
4	Male	24	194	95	58	196	right
5	Male	26	180	73	70	194	right
6	Male	26	183	69	60	194	right
7	Male	23	173	86	60	197	right
8	Male	32	179	87	66	188	left
9	Male	31	168	65	54	189	right

In this work, we omitted Subject 9, as we observed that, due to communication problems, most of the data for that subject were either recorded incorrectly or lost.

2.1.2. Devices

For the data acquisition stage, three Colibri® wireless inertial measurement units (IMUs) were used. These devices are fastened to the wrist on the dominant arm, to the chest and to the ankle on the dominant side, respectively. The IMUs are commercially available through Trivisio Prototyping GmbH (Trier, Germany), an engineering company that specializes in wearable devices. These devices carry 3-axis state-of-the-art sensors that include the following:

- An accelerometer with a scale of ±16 g and a 13-bit resolution
- An accelerometer with a scale of ±6 g and a 13-bit resolution
- A gyroscope with a scale of ±2000°/s and a 16-bit resolution
- A magnetometer with a scale of ±1.3 Ga and a 12-bit resolution

Moreover, the devices also feature a temperature sensor with an accuracy of ±0.5 °C. In addition to temperature, the device provides information about its orientation (pitch, roll and yaw). The size of each device is 52 mm × 42 mm × 17 mm; it weighs 41 g and is able to operate in a self-powered manner within a temperature range from 0 to 55 °C and within a 10-m working distance. The IMUs have a sampling frequency of 100 Hz.

The heart rate monitors worn by the subjects were Model BM-CS5SR from BM Innovations GmbH (BM Wireless Ltd.&Co.KG, Hörgertshausen, Germany). These monitors have a frequency of approximately 9 Hz and an estimated data transmission range of approximately 30 m.

The wearable devices were connected to a Viliv S5 UMPC ultra-portable PC (Dynamism Inc., Chicago, IL, USA), configured with an Intel (Santa Clara, CA, USA) Atom Z520 1.33-GHz CPU and 1 GB of RAM. This PC was used both for storing the data obtained from the devices and for labeling the activities performed at every stage of the protocol.

2.1.3. Protocol

All subjects adhered to a protocol describing a sequence of 12 different activities that had to be performed in a specific order, spending a predetermined amount of time on each activity.

The description of the activities to be performed is as follows:

- lying quietly while doing nothing (small movements are allowed).
- sitting in a chair (changing postures is allowed).
- standing still (talking or gesticulating is allowed).
- ironing one or two shirts.
- vacuuming one or two rooms, moving objects if required.
- ascending stairs for a distance of five floors.
- descending stairs for a distance of five floors.
- walking outside with a speed of 4 to 6 km/h.
- Nordic walking on asphaltic terrain using asphalt pads on the walking poles.
- cycling with slow to moderate pace.
- running or jogging outside.
- jumping rope either with both feet at the same time or alternating feet.

Table 2 describes the protocol defined for data collection. The third and fourth columns specify the activity code and the estimated effort by means of the metabolic equivalent (MET), respectively, as provided by Ainsworth et al. [44].

Table 2

Protocol for data collection, including the activity, the estimated effort (through the metabolic equivalent) and the time each activity must be performed. MET, metabolic equivalent.

#	Activity	Code	Effort (MET)	Time (min)
1	lying	07011	1.0	3
2	sitting	09040	1.8	3
3	standing	09050	1.8	3
4	ironing	05070	2.3	3
5	break			1
6	vacuuming	05043	3.5	3
7	break			1
8	ascending stairs	17130	8.0	3
9	break			2
10	descending stairs	17070	3.0	3
11	break			1
12	ascending stairs	17130	8.0	3
13	descending stairs	17070	3.0	3
14	break			2
15	walking	17190, 17200	3.3–3.8	3
16	break			1
17	Nordic walking	?	5.0–6.0	3
18	break			1
19	cycling	01010	4.0	3
20	break			1
21	running	12020, 12030	7.0–8.0	3
22	break			2
23	jumping rope	15551, 15552	8.0–10.0	3

While this protocol should ideally be followed by all subjects, in a real setup, some data are lost or some subjects refuse to perform certain activities, resulting in invalid data in the dataset. Table 3 shows the total time spent by each subject in each activity. It should be noted that the actual collected data differ from the expected collection results according to the protocol. More importantly, this information clearly justifies the exclusion of Subject 9 from this work.

Table 3

Actual time (in seconds) spent by the subjects in each activity (this may differ from the protocol). Sub., subject.

	Sub. 1	Sub. 2	Sub. 3	Sub. 4	Sub. 5	Sub. 6	Sub. 7	Sub. 8	Sub 9
Lying	271.86	234.29	220.43	230.46	236.98	233.39	256.10	241.64	0.00
Sitting	234.79	223.44	287.60	254.91	268.63	230.40	122.81	229.22	0.00
Standing	217.16	255.75	205.32	247.05	221.31	243.55	257.50	251.59	0.00
Walking	222.52	325.32	290.35	319.31	320.32	257.20	337.19	315.32	0.00
Running	212.64	92.37	0.00	0.00	246.45	228.24	36.91	165.31	0.00
cycling	235.74	251.07	0.00	226.98	245.76	204.85	226.79	254.74	0.00
Nordic walk	202.64	297.38	0.00	275.32	262.70	266.85	287.24	288.87	0.00
Ascending stairs	158.88	173.40	103.87	166.92	142.79	132.89	176.44	116.81	0.00
Descending stairs	148.97	152.11	152.72	142.83	127.25	112.70	116.16	96.53	0.00
Vacuuming	229.40	206.82	203.24	200.36	244.44	210.77	215.51	242.91	0.00
Ironing	235.72	288.79	279.74	249.94	330.33	377.43	294.98	329.89	0.00
Jumping rope	129.11	132.61	0.00	0.00	77.32	2.55	0.00	88.05	64.9

After the data collection stage was complete, data were stored in tabular format using one row for each sample with 54 columns for the attributes:

- 1   timestamp (in s).
- 2   activity.
- 3   heart rate.
- 4 to 20   IMU on wrist.
- 21 to 37   IMU in chest.
- 38 to 54   IMU on ankle.

For each IMU, the data are structured in columns as follows:

- 1   temperature (in °C).
- 2 to 4   3D acceleration data (ms^{-2}), scale ±16 g.
- 5 to 7   3D acceleration data (ms^{-2}), scale ±6 g.
- 8 to 10   3D gyroscope data (rad/s), scale ±2000 °/s.
- 11 to 13   3D magnetometer data (µT), scale ±1.3 Ga.
- 14 to 17   orientation.

2.2. Signal Preprocessing

Because the data collection stage was performed in a real-world scenario, the raw data contain noise, invalid data or missing samples. For this reason, preprocessing is an essential phase before performing classification. This section describes the different actions that take place in the preprocessing stage, which receives as input the original raw data and returns a new, transformed dataset.

2.2.1. Remove the Timestamp

The timestamp is not a useful feature for classification, as it is not part of the physiological signals extracted during the data collection stage. Thus, the timestamp should be removed; otherwise, the classifier might learn the protocol sequencing and achieve a high classification accuracy due to this feature.

2.2.2. Remove the Orientation

The authors of the PAMAP2 dataset state that the orientation information it contains is either invalid or not relevant for this data collection effort [18]. For this reason, we also removed the orientation data, consisting of a total of 12 features in the dataset, four per each IMU.

2.2.3. Remove Transitions

After one activity ends and before the next one starts, the dataset contains samples labeled as transition. These samples do not correspond with any activity; instead, they are noise and should be removed.

2.2.4. Estimate Missing Values

In the PAMAP2 dataset, missing values are specified as not-a-number (NaN). For most attributes, the number of missing values is negligible, and these occur due to communication failures between the IMUs and the PC. For heart rate measurements, most missing values are because of the frequency of the heart monitor (9 Hz), compared to the 100 Hz of the IMUs. Therefore, about 8 out of 9 samples have missing values for the subjects' heart rates.

In all cases, we estimated missing values as the average of the previous and the next available values. We consider this naive approach to be valid because the missing values would not be expected to change significantly within one hundredth of a second (or one ninth of a second in the case of the heart rate).

2.3. Signal Segmentation

Initially, the raw data are in the time domain because each sample captures values from the different sensors at a certain point in time. Of course, these tabular data are suitable for training machine learning models, but we expect to achieve higher classification accuracy when the raw data are transformed to capture temporal information as in previous works, such as those by [22,45]. Bearing this consideration in mind, we designed a segmentation stage in which we transform the input data into the frequency domain.

Figure 3 shows how patterns in movement are different between walking and running and how they have common patterns among different subjects for the same activities.
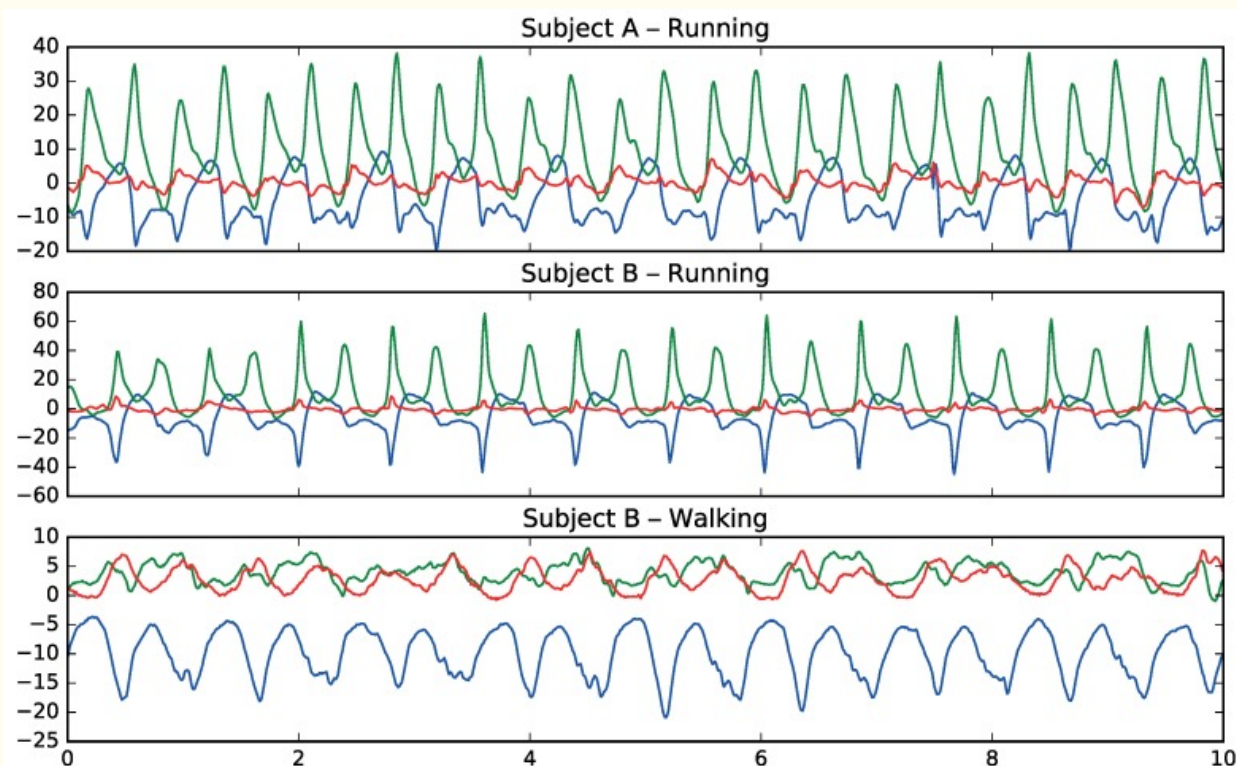
Figure 3

Information from the accelerometer sensor for 2 different activities (running and walking). The unit of the vertical axis is m/s^2. The colors are blue (x), green (y) and red (z).

To do this, for each original feature in the dataset, we take a sliding window of a size of 512, corresponding to 5.12 s of physical activity data. We guarantee that these sliding windows contain samples from only one activity. Then, we compute a discrete Fourier transform (DFT) for this signal window using the FFT algorithm. As a result, the signal is transformed into the frequency domain, and the output comprises 512 values.

2.4. Feature Extraction

After transforming the input data into the frequency domain by computing the DFT for each sliding window of each attribute, we process the resulting signal to extract features that compose the new transformed dataset.

In Pirttikangas et al. [12], the authors found the most important feature to be the mean acceleration; they did not include mean heart rate or heart rate correlations in their subset of best features. However, other works, such as those by [17,21], concluded that mean heart rate is a significant feature depending on the type of physical activity involved.

In the feature extraction stage, each sliding window generates a new instance. To compose the features of this instance, we compute a statistical summary of the 512 values resulting from applying the DFT; this summary consists of the following values: average, median, variance, maximum, minimum, 25% percentile and the 75% percentile. As a result, each physiological feature in the preprocessed dataset is substituted by these 7 features in the resulting dataset.

In conclusion, we end up with a dataset containing 280 features and one class (the physical activity), where each feature is a value from the statistical summary of the DFT of another feature in the preprocessed dataset.

Regarding feature selection, a wide variety of feature selection methods that can identify which attributes are most important for classifiers can be found in [46]. The selection of significant features is important to reduce the dimensionality of the classification problem and because in several domains (including the medical domain), the identification of discriminative features can contribute significantly to the domain knowledge.

In this study, we decided not to perform feature selection for two reasons. First, the main objective of this work is to evaluate how well state-of-the-art classifiers deal with very small dataset sizes for physical activity recognition. We did not want to reduce the features and the size of the dataset at the same time. Second, as we learned from previous works by [45], feature selection affects different classifiers in different ways; therefore, depending on the feature selection method used, we would either positively or negatively affect some classifiers. To avoid this bias in the results, we would have had to execute many combinations of different feature selection methods with the algorithms and parameters, which would detract from the main goals of this work and affect the generality and understanding of this paper.

2.5. Classifier Model Training

The last stage of the activity recognition chain involves the training of a classification model that is able to discriminate the physical activity performed by a certain subject. Our objective is to eventually obtain a classifier that maximizes accuracy (i.e., the hit rate between the predicted and the actual class in a test set).

The resulting classifier is intended to be used in a real-time cross-person prediction system that must be able to accurately predict the activity that a new user is performing. When facing this problem, two different approaches are commonly considered regarding the data feeding the classification model; either (1) the model is subject-independent, and data from a set of different users are used to train the classifier; or (2) the model is subject-group specific, and only historic data from similar users (for example, clustered by age, gender, etc.) are used for training and prediction. While the second approach could potentially provide higher accuracy, it has an important flaw: the system is not able to provide accurate predictions during cold start, when there is not enough information about the users. With cross-person prediction, subject-specific training cannot be performed because we have no data to use for new users. Another interesting and novel approach proposes using template matching to classify sports activities. This approach showed robust classification accuracies when tested on unseen data and when limited training examples were available [47].

For this reason, the classification system we propose features a hybrid approach that is subject-independent during cold start, but eventually turns into a subject-specific model after enough data have been recorded for a particular user. To test this approach, in this paper, we use leave-one-subject-out (LOSO) cross-validation. This evaluation methodology splits the dataset into N folds, where N is the number of subjects (in this case $N = 8$), using $N - 1$ folds as the training set and the remaining fold as the test set. Through this procedure, we guarantee that the classifier learns nothing about the subject to be predicted, thus decreasing learning bias and achieving accurate results for cross-person prediction. Note that an evaluation methodology that does not use LOSO would provide better results in terms of accuracy [18] because the model would have already encountered instances from the target subject, but its results would not be representative of the actual prediction capabilities during cross-person prediction or during a cold start.

To choose the most adequate classification model, we first completed a preliminary evaluation that compared different supervised learning techniques using increasingly small percentages (e.g., 5%, 1% and 0.1%) of the complete input data. Some techniques provide higher classification accuracies when data are normalized; thus, in some cases, the samples were normalized to the

range [0, 1] by applying Equation (1), where $\hat{v}^{(f)}$ is the transformed value, $v^{(f)}$ is the original value and both values, the maximum and minimum required to calculate the normalization, refer to feature f, as shown below:

$$\hat{v}^{(f)} = \frac{v^{(f)} - min^{(f)}}{max^{(f)} - min^{(f)}} \tag{1}$$

To prototype a fast evaluation system, we used Python's machine learning library, scikit-learn [48] (Version 0.17.1). To the best of our knowledge, no detailed comparison study has yet been performed on activity recognition that covers the different categories of classifier algorithms available in scikit-learn.

The same data are input to different classifiers, and the results are stored for analysis. When selecting the classifiers, we used a wide set that includes classical [49] and state-of-the-art [50] techniques. We tested all of the classifiers with various parameters. The classifiers with lower performances were discarded. The following classifiers were finally selected to be studied in more detail.

2.5.1. Distance-Based Methods

These algorithms assume that the data have some type of similarity and relations based on geometric properties and can be grouped according to these patterns.

- The k-nearest neighbors algorithm (k-NN) is one of the simplest and most effective nonparametric machine learning algorithms for classification and regression [51]. The k-NN classifier obtained better results for online recognition when compared to C4.5 decision tree. [52]. After testing several parameters, we eventually used the k-NN algorithm with its default scikit-learn parameters (i.e., $k = 5$ nearest neighbors, uniform weights and Minkowski distance with $p = 2$, which is equivalent to the standard Euclidean metric). The Euclidean distance is the most commonly-used distance metric for continuous variables; however, when working with high-dimensional data (e.g., more than 10 dimensions), it is highly recommended that dimension reduction by means of preprocessing techniques, such as principal component analysis (PCA) or linear discriminant analysis (LDA), be performed. However, for this work, as discussed earlier, we are not testing different feature selection methods or dimension reduction by means of decomposition algorithms for any of the proposed techniques.

2.5.2. Statistical Methods

These techniques assume that data follow a probabilistic function that needs to be inferred.

- When using the Gaussian naive Bayes (Gaussian NB) to work with continuous data, the usual assumption is that the continuous values related to each class are distributed according to a Gaussian distribution [53]. We decided to include Gaussian NB both because it is a well-known technique in machine learning and because it was the best performer without signal segmentation, obtaining an average accuracy of 0.715. However, with the signal segmentation explained in Section 2.3, some of the variables are highly correlated within a class. This negatively affects Gaussian NB, which relies on a model that assumes zero off-diagonal covariance (i.e., no correlations between variables within a class). This technique implements the default version in scikit-learn, where the likelihood of the features is assumed to be Gaussian. In addition to Gaussian NB, we also tested the

Bernoulli naive Bayes and multinomial methods. In both cases, these methods achieved worse results compared to Gaussian NB. For exact details on the algorithm used to update feature means and variance online, see the work by [54].

- The LDA technique is a generalization of Fisher's linear discriminant [55] that is commonly used in statistics, pattern recognition and machine learning to find a linear combination of features that discriminate between two or more classes [56]. LDA can be used as a feature selection algorithm for dimensionality reduction or as a classifier. It is easily computed, has few parameters to tune, is inherently multiclass and has proven to perform very effectively in practice. LDA is closely related to naive Bayes in the sense that both classifiers assume within-class Gaussian distributions. However, LDA relies on a more flexible model that works better with correlations within a class. Therefore, one would expect the LDA results to be better than Gaussian NB under these settings. We also tested the quadratic discriminant analysis, but obtained poor results compared to LDA. We used the default scikit-learn parameters for the comparisons.

2.5.3. Kernel Methods

These models perform pattern analysis based on a kernel function, which is a similarity function over pairs of data points.

- Stochastic gradient descent (SGD) is a simple, but efficient approach to discriminative learning of linear classifiers under convex loss functions, such as logistic regression or support vector machines (linear). Although this technique has long been available in the machine learning community, it has recently received more attention due to its performance with large-scale problems [57]. SGD's main drawback is that it requires a number of parameters that must be tuned for the method to perform well. For this work, after testing some parameter combinations, we finally used the scikit-learn implementation, which is based on a robust implementation of the averaged stochastic gradient descent algorithm [58]. For this mode, we used the default parameters, setting "hinge" as the loss function, which means that the SGD fits a linear support vector machine and "l2" as the penalty, which is the standard regularization for linear support vector machine models (squared Euclidean norm).
- Support vector machines (SVC linear, SVC RBF) provide state-of-the-art performance for classification, regression and outlier detection, scaling well even with a large-dimensional feature vector [59,60]. The choice of the parameter C is critical to achieve a properly trained support vector machine. Our experiments on this problem concluded the best settings are $C = 0.025$ and $C = 1$ for SVC linear and SVC RBF, respectively. The SVC was implemented using libsvm and tested with various kernel functions (i.e., linear, polynomial, radial basis function (RBF) and sigmoid). Finally, based on the test results, the linear and RBF kernel functions were selected for comparison purposes.

2.5.4. Decision Tree-Based Methods

These techniques build a model that predicts a target variable by learning interpretable decision rules inferred from the training data [61]. Using a decision tree as a predictive model is widely used as a decision support tool because they are easily interpreted (i.e., they provide a sequence of decisions to obtain the final classification result).

- Among the different decision tree implementations, we selected one that is widely used: an optimized version of the classification and regression trees (CARTs) algorithm using Gini impurity as a node splitting measure. CART is very similar to C4.5, but differs in its support for numerical target variables. Moreover, it does not compute rule sets.

We conducted several experiments with decision trees, but the results were not competitive when compared to other techniques, such as random forest or extreme trees. The results obtained with ensembles clearly point out that when executing decision trees, selecting only the relevant features would improve the results significantly. However, because the goal of this work is to compare the performance of algorithms out of the box, we decided not to execute decision trees with a prior feature selection step.

2.5.5. Ensemble Learners

These models combine the predictions of several base estimators built with a given learning algorithm to improve the results and robustness over a single estimator. In [11], the authors evaluated a range of base- and meta-level classifiers and found that combining classifiers using plurality voting provided the best accuracy. When working with these meta-algorithms, it is possible to test a large number of combinations of techniques and parameters. In our work, we ran boosting, bagging and voting with various estimator combinations (k-NN, logistic, random forest and Gaussian); however, we obtained worse results compared to the extra randomized tree ensembles.

- The random forest method consists of an ensemble of decision trees [62]. It combines the predictions made by multiple decision trees, each one generated using a different randomly selected subset of the features [63]. Because random forests are made of several weighted decision trees, they are not easy to interpret (sometimes decisions can be made through voting on contradicting rules). However, they do not require any domain knowledge or complicated parameter settings and perform very well with high-dimensional data. The main parameters to adjust when using these methods are *n_estimators* (i.e., the number of trees in the forest) and *max_features* (the size of the random subsets of features to consider when splitting a node). After testing some values, we set the number of trees to 30 with a maximum depth of 10. Empirically, a good default value is to use $max\_features = \sqrt{n\_features}$ for classification tasks (where *n_features* is the number of features in the data, leading to a *max_features* value for this work of $\sqrt{280} \sim 17$).
- The extra randomized trees (extra trees) method is similar to the random forest method, but the randomness goes one step further in the way splits are computed [64]. The two main differences are that the extra trees method splits nodes by choosing fully random cut points and grows the trees using the entire learning sample. Similar to the random forest method, the main parameters to adjust are *n_estimators* and *max_features*. In this work, we set the number of trees to 30 and expanded the nodes of the tree until either all pf the leaves were pure or until all of the leaves contained less than 2 samples. As with random forest, *max_features* is set to 17.
- Adaptive boosting (AdaBoost) is a machine-learning meta-algorithm that can be used in conjunction with many other types of learning algorithms to improve their performance [65]. The methods for voting classification algorithms such as bagging and AdaBoost have been very successful for improving results with many different classifiers on both artificial and real-world datasets [66]. Particularly, AdaBoost has been called the best out-of-the-box classifier in the world [67]. The AdaBoost algorithm used is AdaBoost-SAMME.R, a variant of SAMME (Stagewise Additive Modeling using a Multi-class Exponential loss function) which is recommended for continuous multi-class domains [68]. We used AdaBoost to test the performance of random forest (AdaBoost random forest) and extra randomized trees (AdaBoost extra trees) algorithms. The parameters for the tree-based ensembles were the same as described earlier.

2.5.6. Deep Learning

These techniques are an extension or a re-branding of neural networks [69]. They are based on the idea of modeling problems with high-level abstractions in data by using multiple processing layers that basically perform multiple non-linear transformations. These multiple transformations are organized hierarchically, with each layer processing the outputs of the previous layer, exploiting the idea of incremental learning, in which more abstract concepts are learned from the lower level ones. The processing layers can have various architectures, such as deep neural networks, convolutional deep neural networks, deep belief networks or recurrent neural networks. The idea of using deep learning layers is very interesting to researchers in many fields because these techniques can automatically build high-level representations of the raw input, offering a more generic solution because the feature construction process can be fully automated. Deep learning techniques have been successfully applied to many different fields, including computer vision, natural language processing, speech recognition, and so on, and have achieved state-of-the-art results on various tasks [70,71,72]. Regarding activity recognition, deep learning techniques have been applied in a few works; a representative set can be found in [26,27,28,29,30,31].

- For a deep neural network (DNN), we chose the TensorFlow framework, recently introduced by Google (Mountain View, CA, USA) [73]. Additionally, to achieve a better integration with scikit-learn, we used the skflow package, which provides a simplified interface for accessing and mimicking TensorFlow scikit. As mentioned before, deep learning techniques have the advantage of being able to build high-level representations of the raw input data without the need for data processing and segmentation. While this is an interesting approach, we have identified a very high impact of the network architecture on the results, a behavior consistent with what we have found in the literature [32], including the fact that some architectures do not work at all. Therefore, in this work, to create an objective comparison, we used the same preprocessed data to test all of the classifiers, including the DNNs. The main motivation for testing these techniques with preprocessed data is to investigate whether their ability to extract and/or transform features among the 280 attributes translates into competitive classification performances. Given the importance of the architecture design, for this work, we have tested numerous topologies (combinations of tensors and dense and convolutional layers). However, after running a large set of experiments, we observed through the evolution of the logistic regression loss function that simple architectures were sufficient to converge within a few iterations obtaining competitive results, as shown in Figure 4. Adding more complexity to the topology required considerably more computation time, all to achieve only very small improvements in accuracy (or even no improvement at all due to overfitting). The results of our tests with different architectures pointed out two things: (1) incorrect topology designs dramatically affect the results; and (2) the results obtained with simple architectures were probably so close to the global optimum that adding more complexity in the layers did not lead to any significant improvement.
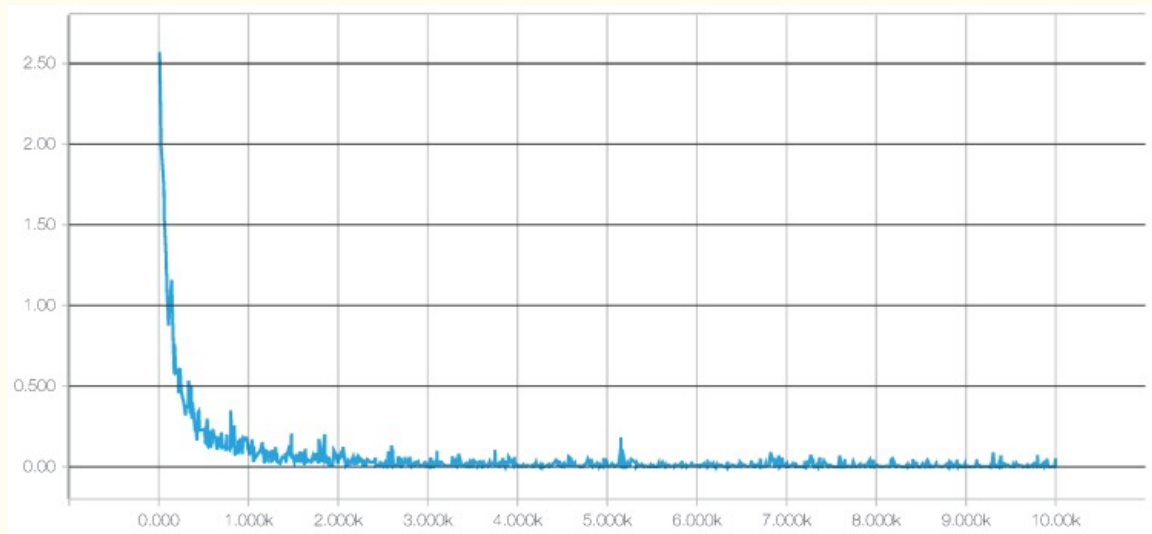
Figure 4

Evolution of the logistic regression loss function along iterations of the deep neural network training process, generated with TensorBoard.

Consequently, for comparison purposes, we finally selected two configurations: DNN 1L, which has 1 hidden dense layer with 280 nodes, and DNN 2L, which has 2 hidden dense layers with 560 and 280 nodes, respectively. For both configurations, we used a logistic regression as the cost function, 10,000 iterations with the Adam optimizer, a dropout probability of 0.5 and a learning rate of 0.001. The computational graph for training such a deep neural network, as generated by TensorBoard, is shown in Figure 5.
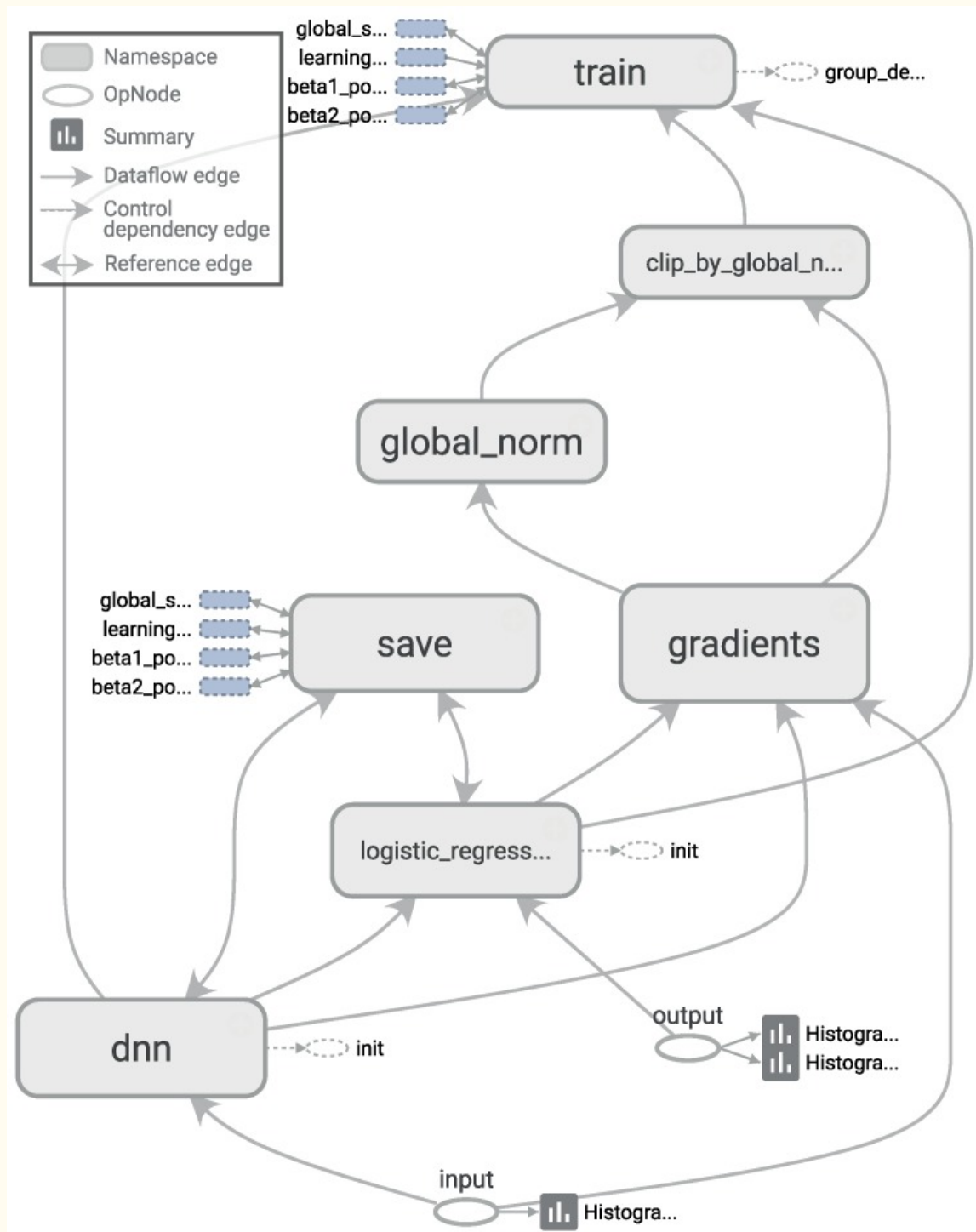
Figure 5

Architecture of a TensorFlow graph for training a deep neural network, generated with TensorBoard.

2.6. Parameter Setup

Table 4 shows the parameters used when running each of the classifiers. These parameter values were determined after a previous phase of manual exploration. The table also indicates whether the input data were normalized to the range [0, 1] after training the classifier. Note that while both normalized and non-normalized data were tested with all algorithms, only those

configurations that led to higher accuracies are detailed in this paper. Although they are not shown in the table, random seeds are chosen randomly; thus, they are different for each run. Parameters that are not shown in the table were set to their default values.

Table 4

Classifiers used in scikit-learn along with the values set for their parameters and whether data were previously normalized in the range [0, 1].

Classifier	Normalized
sklearn.neighbors.KNeighborsClassifier (`k = 5`)	Yes
sklearn.naive_bayes.GaussianNB	No
sklearn.discriminant_analysis.LinearDiscriminantAnalysis	No
sklearn.linear_model.SGDClassifier (`loss = hinge, penalty = 12`)	Yes
sklearn.svm.SVC (`kernel = linear, max_iter = - 1, C = 0.025`)	Yes
sklearn.svm.SVC (`kernel = rbf, max_iter = -1, C = 1`)	Yes
sklearn.ensemble.ExtraTreesClassifier (`n_estimators = 30`)	No
sklearn.ensemble.RandomForestClassifier (`n_estimators = 30, max_depth = 10`)	No
sklearn.ensemble.AdaBoostClassifier (`ExtraTreesClassifier, n_estimators = 30`)	No
sklearn.ensemble.AdaBoostClassifier (`RandomForestClassifier, n_estimators = 30, max_depth = 10`)	No
skflow.ops.dnn (`[280], logistic_regression, keep_prob = 0.5, steps = 10000, optimizer = Adam, learning_rate = 0.001`)	Yes
skflow.ops.dnn (`[560, 280], logistic_regression, keep_prob = 0.5, steps = 10000, optimizer = Adam, learning_rate = 0.001`)	Yes

2.7. Quality Metrics

For each individual run of all of the experiments conducted, we collected accuracy, precision, recall and F_β scores, with $\beta = \{1, 2\}$. The accuracy metric is simply the percentage of correctly-classified instances. Although the dataset used for this work is not perfectly balanced, because we are working with 12 classes, we consider that average accuracy metric to be both a simple and appropriate metric for representing the classifiers' overall performances. For this reason, and to aid readability, we summarized only the accuracy metric plus variance for the preliminary evaluation. Of course, we also analyzed the confusion matrix, precision, recall, F_1 and F_2, but without finding any significant results to discuss.

For the selected classifiers, we have included more detail about the confusion matrix, precision (Equation (2)), recall (Equation (3)) and F_β (Equation (4)) scores, with $\beta = 1$ and $\beta = 2$. Because this is a multiclass problem, these metrics are obtained by weighted averaging. The following equations describe how each of the previous metrics is computed:

$$(2)$$

$$P(y, \hat{y}) = k \sum_{l \in L} |\hat{y}_l| \frac{|y_l \cap \hat{y}_l|}{|y_l|}$$

$$R(y, \hat{y}) = k \sum_{l \in L} |\hat{y}_l| \frac{|y_l \cap \hat{y}_l|}{|\hat{y}_l|} \tag{3}$$

$$F_\beta(y, \hat{y}) = k \sum_{l \in L} |\hat{y}_l| (1 + \beta^2) \frac{P(y_l, \hat{y}_l) \times R(y_l, \hat{y}_l)}{\beta^2 P(y_l, \hat{y}_l) + R(y_l, \hat{y}_l)} \tag{4}$$

These equations conform to the following conventions:

- y is the set of predicted (sample, label) pairs.
- $\hat{y}$ is the set of true (sample, label) pairs.
- L is the set of labels.
- y_l and $\hat{y}_l$ are, respectively, the subsets of y and $\hat{y}$ with label l.
- $k = (\sum_{l \in L} |\hat{y}_l|)^{-1}$

Another commonly-used quality metric is the receiver operating characteristic (ROC) metric; however, for this work, we did not analyze the ROC curves, because this is a multiclass problem. First, ROC curves, which generally work well for visualization, would be difficult to visualize for 12 classes, requiring the volume under the ROC surfaces metric (VUS) or Cobweb diagrams. Second, for each class, a pairwise comparison of the performance of that class versus all of the other classes would be required [74,75].

3. Results

This section describes the results obtained from the evaluation of the different classifiers.

3.1. Preliminary Evaluation

The purpose of the preliminary evaluation is to quickly evaluate the performance of the very different techniques included. To do this, we used small percentages of the training data, thus decreasing the required training time. In particular, we first tried running the experiments with 5% of the training data. Additionally, to check whether the subset size affects the accuracy, we also compared the performance of the various techniques using sampled subsets of 10%, 1%, 0.5%, 0.1%, 0.05%, 0.01%, 0.005% and 0.001% of the training set, respectively.

The reduced training subset is built by random sampling without replacement. Given that the training subset is random while most algorithms are stochastic, the performance for each technique and subject is computed by taking the average results from 50 runs with different random seeds (two random seeds are used, one for sampling the training set and the other for the classifier initialization).

3.2. Classifiers' Performances

The average performances for each classifier were studied by computing the accuracy on a per-classifier basis. During the experiments, a different random 5% of the training data were sampled for each run, and 50 runs were executed for each subject, totaling 400 runs for each classifier. Because we used LOSO cross-validation, the training set for each subject contains only data from other subjects.

Figure 6 displays the aggregated accuracy performance of each classifier. A more detailed analysis of the average accuracy, as well as the standard deviation are provided on a per-subject basis in Table 5, where the best classifier for each subject is highlighted in bold text. The figure shows that the extra trees technique is the best for Subjects 6 and 8; LDA provides the highest accuracies for Subjects 3, 4 and 7; and the deep neural network is the best choice for Subjects 1 and 2. Surprisingly, while SVC with the RBF kernel provides low accuracy on average, it outperforms the other classifiers when predicting the activity for Subject 5.
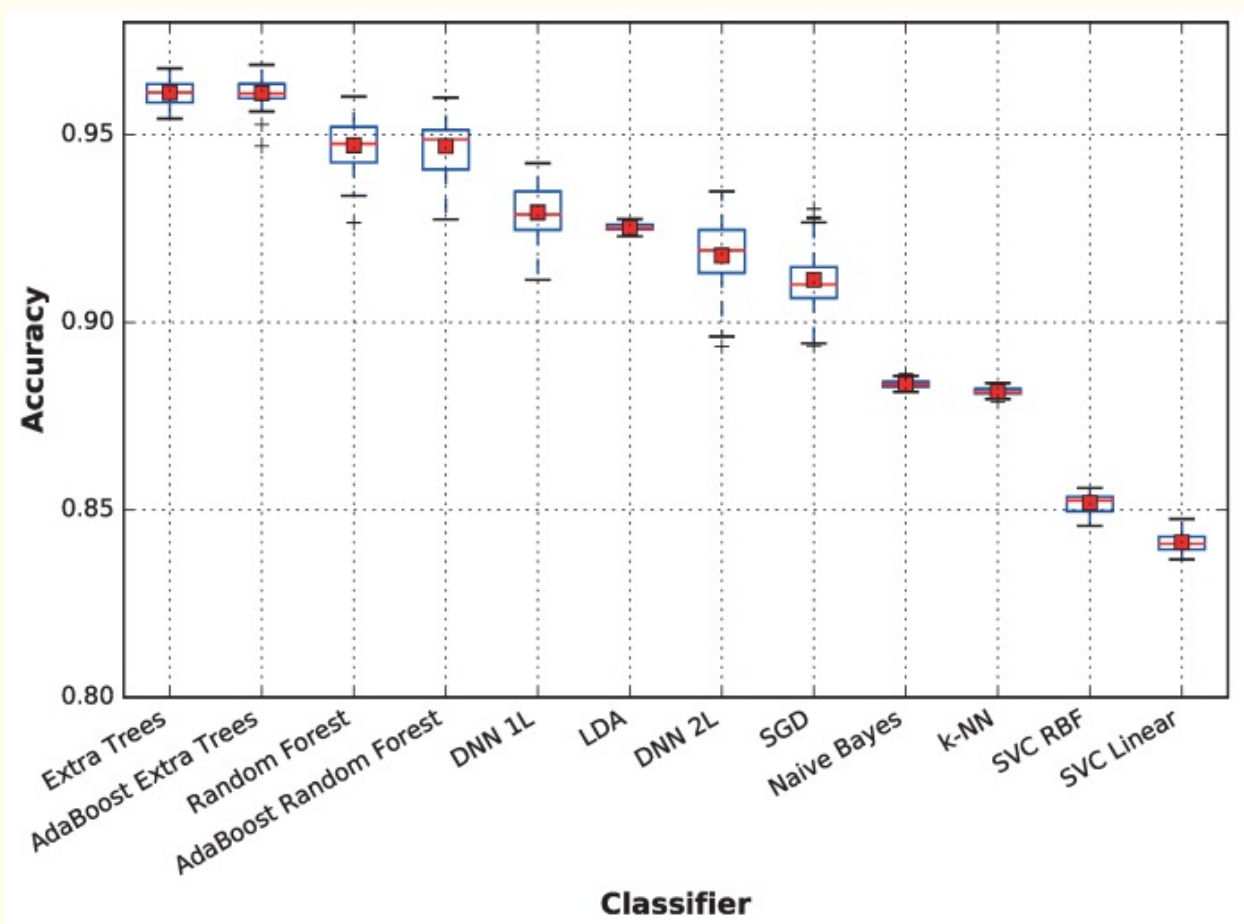

Figure 6

The distribution of accuracies along 50 runs for each classifier. The data represent the average accuracies for all subjects. The results were obtained by training each model 50 times per subject with 5% of the training data, using LOSO cross-validation.

Table 5

Average accuracy (and standard deviation) for each classifier and subject when using 5% of the training data. The bold text indicates the classifiers that outperformed others for each subject. ET, extra trees; RF, random forest; SGD, stochastic gradient descent.
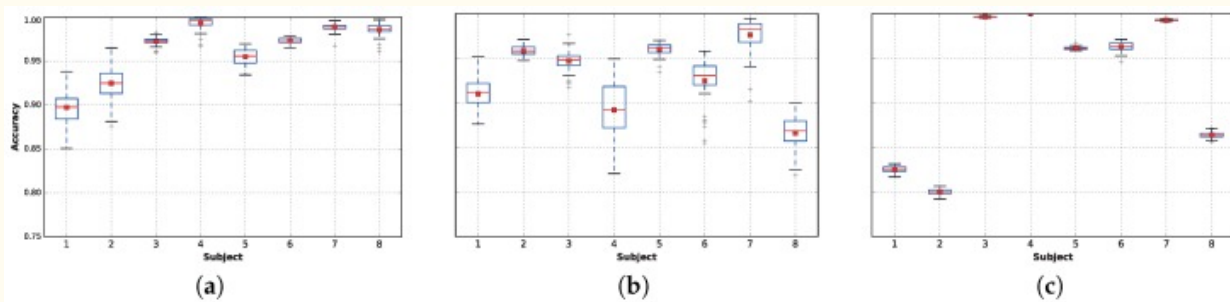
| Classifier | Accuracy | | | | | | | | Av |
	Sub. 1	Sub. 2	Sub. 3	Sub. 4	Sub. 5	Sub. 6	Sub. 7	Sub. 8	
Extra Trees	0.897 (0.019)	0.925 (0.018)	0.973 (0.004)	0.993 (0.008)	0.955 (0.009)	**0.973** (0.003)	0.989 (0.004)	**0.986** (0.007)	C (C
AdaBoost (ET)	0.900 (0.018)	0.922 (0.022)	0.972 (0.004)	0.993 (0.009)	0.954 (0.008)	**0.973** (0.003)	0.989 (0.004)	0.985 (0.008)	C (C
Random Forest	0.874 (0.012)	0.895 (0.045)	0.955 (0.010)	0.982 (0.020)	0.953 (0.014)	0.966 (0.008)	0.975 (0.012)	0.978 (0.005)	C (C
AdaBoost (RF)	0.873 (0.011)	0.897 (0.042)	0.955 (0.010)	0.983 (0.021)	0.951 (0.017)	0.966 (0.007)	0.975 (0.012)	0.977 (0.005)	C (C
DNN 1L	**0.910** (0.016)	**0.959** (0.006)	0.947 (0.012)	0.892 (0.033)	0.960 (0.007)	0.925 (0.024)	0.976 (0.002)	0.866 (0.019)	C (C
LDA	0.825 (0.003)	0.800 (0.003)	**0.997** (0.001)	**1.000** (0.000)	0.961 (0.002)	0.963 (0.005)	**0.993** (0.001)	0.864 (0.003)	C (C
DNN 2L	0.908 (0.021)	0.956 (0.008)	0.913 (0.027)	0.870 (0.053)	0.948 (0.012)	0.934 (0.025)	0.955 (0.031)	0.858 (0.029)	C (C
SGD	0.862 (0.011)	0.826 (0.043)	0.973 (0.006)	0.923 (0.025)	0.971 (0.005)	0.922 (0.014)	0.983 (0.001)	0.831 (0.034)	C (C
Naive Bayes	0.821 (0.006)	0.604 (0.004)	0.961 (0.001)	0.991 (0.001)	0.922 (0.002)	0.955 (0.001)	0.941 (0.002)	0.874 (0.003)	C (C
k-NN	0.860 (0.003)	0.950 (0.001)	0.855 (0.004)	0.719 (0.005)	0.942 (0.002)	0.886 (0.003)	0.955 (0.003)	0.886 (0.002)	C (C
SVC RBF	0.865 (0.003)	0.864 (0.017)	0.815 (0.006)	0.619 (0.006)	**0.975** (0.001)	0.933 (0.002)	0.869 (0.005)	0.876 (0.005)	C (C
SVC Linear	0.858 (0.004)	0.820 (0.011)	0.816 (0.010)	0.601 (0.009)	0.974 (0.002)	0.917 (0.004)	0.883 (0.007)	0.861 (0.007)	C (C

When considering the average performance, techniques based on ensembles of decision trees outperform the other techniques, followed by deep neural networks and linear discriminant analysis, which also has a very low variance. On the other hand, support vector machines obtain a very low performance; both SVC with linear and RBF kernels and k-NN and naive Bayes result in below-average accuracy when there are no competitive results for any subject.

The results of the meta-classifiers (AdaBoost over both extra trees and random forest) show that they do not significantly affect performance. For the remainder of this paper, the following classifiers will be omitted from the comparisons: AdaBoost extra trees and AdaBoost random forest, because they provide results very similar to those not using meta classifiers; random forest, because it is dominated by extra trees, another decision-tree based ensemble; DNN 2L (featuring two dense layers), because it provides no advantage over DNN 1L and achieves

slightly worse results; and naive Bayes, k-NN and SVC, because they do not achieve competitive performances compared to the other classifiers, obtaining significantly lower averages and medians. Nevertheless, in the paper conclusions, we will place all classifiers in a concept map showing how they compare relative to each other both in accuracy and in sensitivity to the dataset size.

For the remaining classifiers, extra trees, DNN 1L (featuring one dense layer) and LDA, Figure 7 plots the accuracy distributions for each subject when using only 5% of the data. In general, extra trees (Figure 7a) provides very high accuracies for most of the subjects, although DNN (Figure 7 b) provides better results for Subjects 1 and 2. Finally, LDA (Figure 7c) provides extremely high accuracies for Subjects 3 and 7 and perfect accuracy in all 50 runs for Subject 4. The best results found in this work are slightly better than the ones presented in Table 4.9 of [18]; however, in comparing the results, it is important to note that [18] included Subject 9, while we decided to remove him from this work because he performed only the rope jumping activity for 64.90 s, as reflected in Table 3. Nevertheless, because Subject 9's task is quite easy to classify (96% average accuracy), if we had included him in the training sets, our overall results would have improved, outperforming the results presented in [18] even more.

Figure 7

Distribution of accuracy over the test set for each subject, for the outperforming classifiers, using 5% of the training data. Each distribution results from the execution of 50 runs. (a) Extra trees; (b) DNN 1L; (c) LDA.

3.3. Sensitivity to Training Sample Size

In this paper, we also investigated how different classifiers perform when the training dataset is significantly small. This is an interesting research topic for three reasons. First, working with a small sample of the datasets means less power consumption when processing and transmitting data, and a low energy consumption is vital for most wearable devices. Second, it allows us to simulate a cold start situation in which only a few users have been used for training, but new users are starting to use the system. Third, when working with large datasets, reducing the amount of data required would enable the processing of big data, effectively reducing its size by several orders of magnitude simply by performing random sampling of some tranches. We hypothesize that the performance of classifiers is not equal (given the correlations between different training sample sizes and accuracy), but we wanted to validate this hypothesis and explore these effects for the different chosen classifiers.

In Figure 8, it can be observed that after 0.05%, the accuracy decreases if the sample size is reduced further, resulting in statistically-significant differences in performance. The results obtained within the range of 10% to 0.05% have very small variations that can be explained by the random seed used for each run. The reason these results are nearly identical is because the

FFT collects data from overlapping sliding windows of 5.12 s. Therefore, even when instances are removed from the training set, there is no real loss of data. Loss of data truly begins to affect the tested classifiers in experiments where less than 0.05% of the data is available.
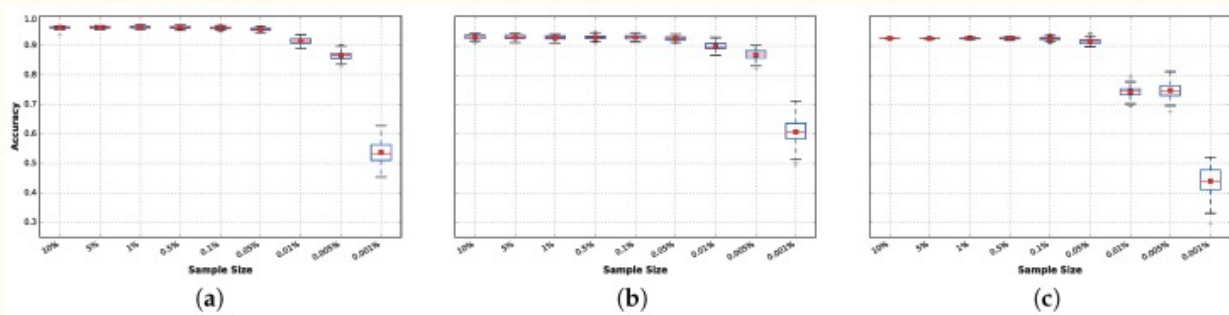

Figure 8

How training sample size affects the accuracy distributions, aggregated for all subjects, for the high-performing classifiers. (**a**) Extra trees; (**b**) DNN 1L; (**c**) LDA.

However, this effect varies from one classifier to another. In general, there is no significant difference in accuracy for sample sizes as low as 0.05% of the complete training dataset for any of the techniques. However, this behavior changes when the training sample size is reduced to 0.01%: At this point, DNN (Figure 8b) and extra trees (Figure 8a) suffer from a small reduction in accuracy, while this reduction is significantly magnified in the case of LDA (Figure 8c), for which the average accuracy drops to approximately 75%.

When reducing the data further (from 0.01% to 0.001%), significant performance differences arise. These are particularly noticeable for LDA, while DNN is the least affected technique. It is noteworthy that the best classifiers obtained results above 90% accuracy on average, even with only 0.005% of the continuous data available, which corresponds to collecting only approximately 9.17 consecutive seconds of each activity per subject.

3.4. Analysis of the Results

A more exhaustive summary of the results is shown in Figure 9, which depicts the confusion matrices for the three selected machine learning techniques: extra trees, deep neural networks and linear discriminant analysis. The confusion matrices are obtained from a random execution using 5% of the training data, and the values shown are obtained by aggregating the results for the eight subjects and, later, normalizing the values for each class.
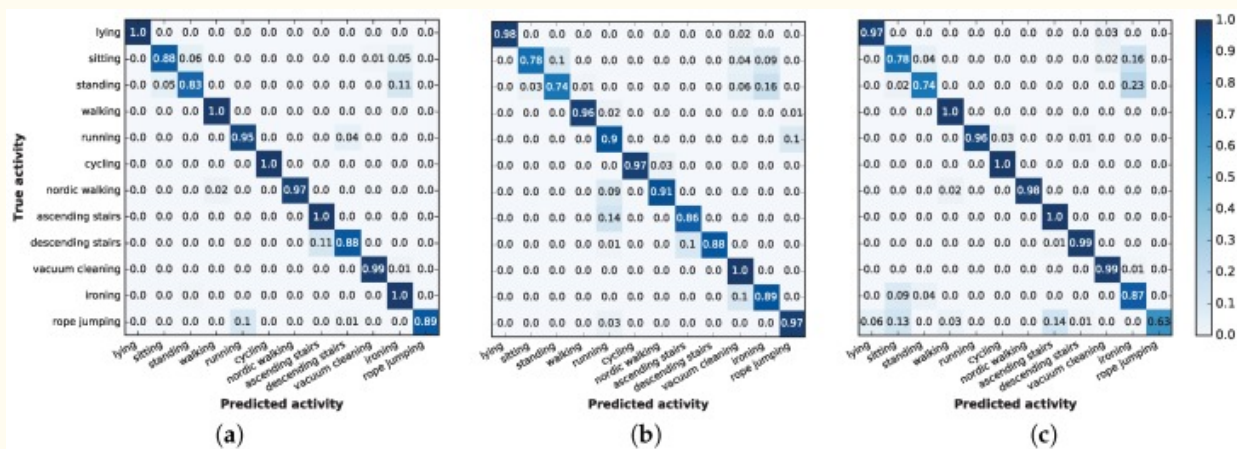
Figure 9

Confusion matrices for the 12 activities aggregated for all subjects, for the high-performing classifiers. (**a**) Extra trees; (**b**) DNN 1L; (**c**) LDA.

The high accuracy levels are reflected in the three matrices by the very high values in the main diagonal. Many of these values are one, implying that certain activities were perfectly recognized.

In general, a study of the confusion matrices shows that, in all cases, standing is sometimes classified as ironing; however, this misclassification never occurs the other way around. This particular misclassification is particularly common using LDA, which incorrectly recognized 23% of the instances. However, this confusion is reasonable: the protocol defined that people performing the standing activity could still gesticulate, and some hand or arm movements may resemble the act of ironing.

Extra trees and DNN sometimes misclassify the descending stairs action as an ascending stairs action. Moreover, DNN incorrectly classifies ascending stairs or Nordic walking as running, while extra trees confuses these activities with rope jumping. However, these issues occur only 10% to 15% of the time, and again, the activities are remarkably similar; therefore, we consider that these misclassifications are reasonable.

Finally, using LDA led to some remarkable mistakes that cannot be explained by the nature of the activities. In particular, it classified both ironing and jumping rope as sitting, even though these activities are significantly different. The first confusion is symmetric: 16% of the time when a subject was sitting, LDA recognized the activity as ironing. However, LDA confuses jumping rope with several activities, some of which are quite dissimilar. The reason LDA confuses jumping rope with other activities is because only five out eight subjects performed this activity, and those who did spent less time (on average) compared to the time they spent on other activities (one jumped rope for only 2.55 s). This finding is in accordance with our previous findings: LDA requires more data than the other selected methods to achieve an accurate prediction.

For each of these classifiers, other performance metrics, including precision, recall, F_1 score and F_2 score, are shown in Table 6. These metrics were obtained using both 5% and 0.001% of the training data and are the result of 50 runs per subject. The table shows the mean, as well as the standard deviation (enclosed within parentheses).

Table 6

Performance metrics mean (and standard deviation) for the high-performing classifiers, comparing sample sizes of 5% and 0.001%. Higher values (better) are displayed in boldface.

		Extra Trees	**DNN**	**LDA**
5%	Precision	**0.967** (0.003)	0.949 (0.005)	0.926 (0.001)
	Recall	**0.961** (0.003)	0.929 (0.007)	0.925 (0.001)
	F_1 Score	**0.959** (0.004)	0.927 (0.007)	0.912 (0.001)
	F_2 Score	**0.960** (0.004)	0.927 (0.007)	0.917 (0.001)
0.001%	Precision	0.476 (0.005)	**0.547** (0.006)	0.384 (0.005)
	Recall	0.538 (0.004)	**0.607** (0.005)	0.440 (0.005)
	F_1 Score	0.455 (0.005)	**0.536** (0.006)	0.361 (0.005)
	F_2 Score	0.491 (0.004)	**0.567** (0.005)	0.393 (0.005)

For all metrics, when 5% of the training data are used, extra trees outperforms DNN, which outperforms LDA. Still, the variance is smaller for both the latter methods, meaning that their performance is more stable along the 50 runs. However, when the sample size is limited to 0.01% of the training data, DNN begins to outperform extra trees, an effect that can be observed in all of the metrics.

4. Conclusions and Future Work

In this work, we compared different classification techniques for automatic cross-person activity recognition under different scenarios where more or less information is available. We used data from the PAMAP2 dataset collected from eight participants performing 12 activities with wearable devices as explained in detail in Section 2. First, we performed some preprocessing to remove unneeded data and to insert missing values. Then, we extracted time and frequency features from a 5.12-s sliding window using a fast Fourier transform (FFT). As a result, each physiological feature in the preprocessed dataset was replaced with seven features in the resulting dataset (average, median, variance, maximum, minimum, 25th percentile and 75th percentile).

After obtaining the transformed dataset, we compared how the best classifier techniques performed under scenarios with different amounts of training data. We tested sampling rates of 10%, 1%, 0.5%, 0.1%, 0.05%, 0.01%, 0.005% and 0.001% of the complete training set. For each scenario, we tested k-nearest neighbors, Gaussian naive Bayes, linear discriminant analysis, stochastic gradient descent, support vector machines with several kernel functions, decision trees, random forest, extra randomized trees, adaptive boosting and deep neural networks (TensorFlow), among other algorithms. For replication purposes, all of these algorithms were implemented using the scikit-learn library as described in detail in Section 2.5.

When the goal is cross-person prediction, subject-specific training cannot be performed; otherwise, there would be no unseen data for new users. For this reason, we used the leave-one-subject-out (LOSO) cross-validation technique. This evaluation methodology splits the dataset into N folds (here, $N = 8$ subjects), using $N - 1$ folds as the training set and the remaining fold as

the test set. Using this methodology, we can guarantee that the classifier learns nothing about the remaining subject being evaluated for prediction. Consequently, the possible learning biases decrease, and we can obtain reliable results when performing cross-person prediction.

After conducting a preliminary set of experiments, we concluded that with our experimental setup, extra randomized trees performed extremely well, outperforming state-of-the-art results described in previous works [18]. Other related work using the same dataset and performing feature selection with random forest have reported an overall accuracy of 97.9% [76]. However, because their results are not evaluated using LOSO, their training set includes subject information, and therefore, their results are not directly comparable to ours. The results obtained with extra randomized trees were so good (96.1%±0.009% on average) that using AdaBoost did not improve them. The results of the other classifiers are in accordance with reports by previous works [18,45,77]. For instance, when analyzing the confusion matrix, incorrect predictions only appear in neighbor (similar) classes. Random forest performed quite well, achieving second place in overall average accuracy, which is unsurprising considering that extra randomized trees are based on random forest. Again, this finding confirms other results from the literature [18,24], where the authors stated that, overall, boosted decision tree classifiers and k-NN attained the best performances.

Analyzing the results in further detail, we found that LDA was able to achieve the best results in three of eight subjects (those whose activities were easiest to predict), and LDA obtained an outstanding 100% for Subject 4. The deep neural networks were able to achieve the best results for Subjects 1 and 2, whose activities are more diverse and difficult to predict. At this point, we can confirm that we found competitive results for deep neural networks and have successfully used TensorFlow to classify physical activities. To our knowledge, this is one of the first works in the literature to do so.

However, as mentioned above, more complex deep architectures did not translate to better results, mainly because, after processing the dataset, the problem was not complex enough and because deeper architectures tend to overfit the training sets. We note that these simple architectures are conceptually not pure deep learning. Nevertheless, the results attained in this work outperform previous works using deep learning over raw data [32], and it is even more interesting considering that our results are cross-subject. We think that this topic itself is significantly interesting and extensive, and we have decided to leave a whole analysis of the behavior of deep learning techniques with raw/preprocessed data as a future work.

After conducting a sensitivity analysis of different training sample sizes, we concluded that deep neural networks were the techniques whose average accuracies were least affected (see Table 6). Another interesting conclusion regarding training sample size is that sample sizes of 0.05% obtain results very similar to experiments using sample sizes of 10%. This occurs for two reasons: there is intrinsic redundancy in the data, and this type of data is easily generalizable from a small sample. The most important consequence of this finding is that, using this dataset, we could obtain essentially identical results by collecting only 22.67% of continuous data from the devices rather than the complete set of labeled activities. It is remarkable that the best classifiers obtained results above 90% accuracy on average using only 4.53% of the complete dataset, which corresponds to an average of only 9.17 consecutive seconds of each activity by each subject.

To put it together, Figure 10 shows the relative strength of each classifier both in terms of accuracy and sensitivity to the dataset size (i.e., variance in the accuracy as the dataset is reduced). The best performing classifiers are shown in the top right quadrant, and we can see how extra trees and DNN 1L are non-dominated choices.
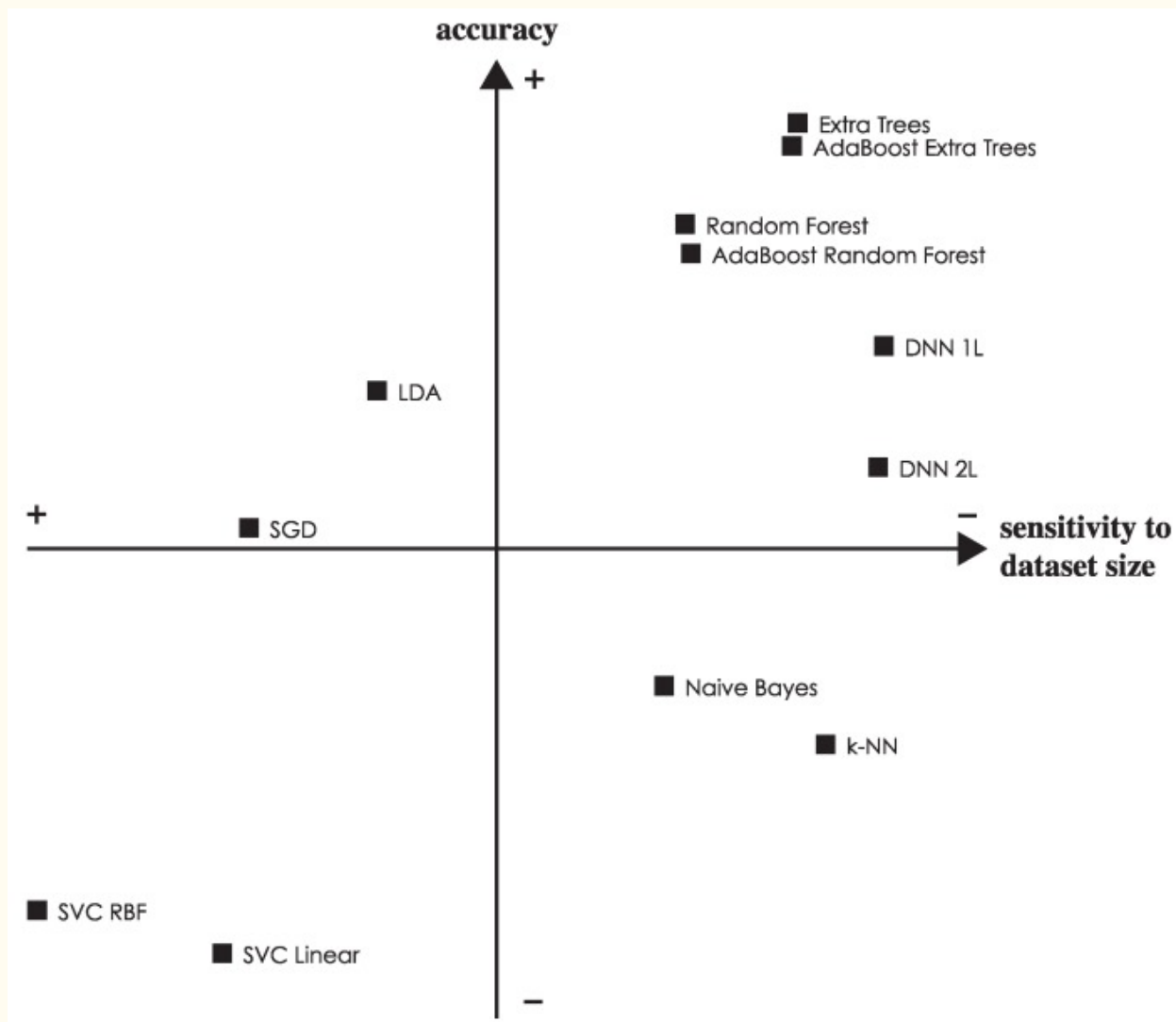
[Figure 10](#)

Concept map showing the relative performance of each classifier both in terms of accuracy and sensitivity to the dataset size. In the former metric, more (up) is better, while in the latter, less (right) is better, as lower values represent less variance in the accuracy when reducing the dataset size.

As said before, this is one of the first works using the scikit-learn library for physical activity recognition. After working extensively with scikit-learn, we can strongly recommend it to the scientific community. The library reduces the overall development time and simplifies the experimental replication process. Moreover, because the programming language is Python, many additional libraries are available, which are, based on our experience, much more powerful tools than WEKA.

As future work, in addition to further investigating deep learning and raw data, as mentioned above, there are two other correlated research lines that require attention. First, there is a need to test more datasets collected from smartphones or wearable devices, such as [17,47,78,79] or collected by our team. Second, there is a need to build a benchmark tool that includes more classifiers and feature selection methods. We also would like to compare performances between datasets of different physical activities using template matching, as observed in [47]. After accomplishing these objectives, we would like to release this benchmark as open-source code, including all of the implemented feature selection methods and classifiers and the collected datasets.

We are aware that the levels of accuracy found in this work may decrease in the real world; here, sensors position might vary or interference from other devices may occur. In future work, it would be ideal to compare laboratory results with subjects in real-world environments, as studied in [80].

Regarding feature selection, issues such as finding the optimal combination of the time and frequency domain for features and analyzing how feature selection affects each classifier's performance should be explored. In this sense, in addition to the features used in this paper, we could test the signal-magnitude area [81], energy [11], entropy [82], binned distribution histogram of FFT and time between peaks [83], among others.

Finally, as our main conclusion, when performing cross-subject physical activity recognition in situations with limited data, it is possible to obtain high prediction accuracies (>90%) using the extra randomized trees, random forest and deep neural networks method. Using basic parameters, these methods perform well when generalizing among subjects and with limited data. Other techniques, such as linear discriminant analysis, Gaussian naive Bayes or k-nearest neighbors, support vector machines (with linear and radial kernel functions) or stochastic gradient descent, are not able to achieve high prediction accuracies with such limited data. It is also important to mention that normalizing the input data may positively or negatively affect classifiers. Normalizing the input data is a required step for some classifiers. At present, we can confirm these findings only under the conditions explained in Section 2, but in the future, we will attempt to extrapolate them to different physical activity datasets.

Acknowledgments

Author Contributions

A.B. and P.I. analyzed the data; A.B. preprocessed the dataset, Y.S. conceived and designed the experiments; Y.S. and A.B. implemented and performed the experiments; Y.S., A.B. and P.I. analyzed the results; Y.S. and A.B. wrote the paper.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Janssen I., Leblanc A.G. Systematic review of the health benefits of physical activity and fitness in school-aged children and youth. Int. J. Behav. Nutr. Phys. Act. 2010;7:40. doi: 10.1186/1479-5868-7-40. [PMC free article] [PubMed] [CrossRef]
2. Merglen A., Flatz A., Bélanger R.E., Michaud P.A., Suris J.C. Weekly sport practice and adolescent well-being. Arch. Dis. Child. 2014;99:208–210. doi: 10.1136/archdischild-2013-303729. [PubMed] [CrossRef]
3. Biddle S.J.H., Asare M. Physical activity and mental health in children and adolescents: A review of reviews. Br. J. Sports Med. 2011;45:886–895. doi: 10.1136/bjsports-2011-090185. [PubMed] [CrossRef]

4. Chodzko-Zajko W. Exercise and physical activity for older adults. Univ. Access Inf. Soc. 2014;3:101–106. doi: 10.1123/kr.2014-0043. [CrossRef]

5. Andreu-Perez J., Leff D.R., Ip H.M.D., Yang G.Z. From Wearable Sensors to Smart Implants —Toward Pervasive and Personalized Healthcare. IEEE Trans. Biomed. Eng. 2015;62:2750–2762. doi: 10.1109/TBME.2015.2422751. [PubMed] [CrossRef]

6. Vock C.A., Flentov P., Darcy D.M. Activity Monitoring Systems and Methods. No. 8,352,211 B2. U.S. Patent. 2013 Jan 8;

7. Yuen S.G.J., Park J., Friedman E.N. Activity Monitoring Systems and Methods of Operating Same. No. 8,386,008 B2. U.S. Patent. 2013 Sep 24;

8. Kahn P.R., Kinsolving A., Christensen M.A., Lee B.Y., Vogel D. Human Activity Monitoring Device with Activity Identification. No. 8,949,070 B1. U.S. Patent. 2015 Feb 3;

9. Morris D., Kelner I., Shariff F., Tom D., Saponas T.S., Guillory A. Personal Training with Physical Activity Monitoring Device. No. 8,951,165 B2. U.S. Patent. 2015 Feb 10;

10. White K.L., Orenstein M.L., Campbell J., Self C.S., Walker E., Micheletti M., McKeag G., Zipperer J., Lapinsky M. Activity Recognition with Activity Reminders. No. 20,150,042,468 A1. U.S. Patent. 2015 Feb 12;

11. Ravi N., Nikhil D., Mysore P., Littman M.L. Activity recognition from accelerometer data; Proceedings of the 17th Conference Innovative Applications Artificial Intell (IAAI'05); Pittsburgh, PA, USA. 9–13 July 2005; pp. 1541–1546.

12. Pirttikangas S., Fujinami K., Nakajima T. Ubiquitous Computing Systems. Volume 4239. Springer; Berlin, Germany: 2006. Feature Selection and Activity Recognition from Wearable Sensors; pp. 516–527. Lecture Notes in Computer Science.

13. Antonsson E., Mann R. The frequency content of gait. J. Biomech. 1985;18:39–47. doi: 10.1016/0021-9290(85)90043-0. [PubMed] [CrossRef]

14. Lara O.D., Labrador M.A. A Survey on Human Activity Recognition using Wearable Sensors. IEEE Commun. Surv. Tutor. 2013;15:1192–1209. doi: 10.1109/SURV.2012.110112.00192. [CrossRef]

15. Mukhopadhyay S. Wearable Sensors for Human Activity Monitoring: A Review. IEEE Sens. J. 2015;15:1321–1330. doi: 10.1109/JSEN.2014.2370945. [CrossRef]

16. Shoaib M., Bosch S., Incel O., Scholten H., Havinga P. A Survey of Online Activity Recognition Using Mobile Phones. Sensors. 2015;15:2059–2085. doi: 10.3390/s150102059. [PMC free article] [PubMed] [CrossRef]

17. Tapia E. PhD Thesis. Massachusetts Institute of Technology; Cambridge, MA, USA: 2009. USing Machine Learning for Real-Time Activity Recognition and Estimation of Energy Expenditure.

18. Reiss A. PhD Thesis. Technical University of Kaiserslautern; Kaiserslautern, Germany: 2013. Personalized Mobile Physical Activity Monitoring for Everyday Life.

19. Ponce H., Miralles-Pechuán L., Martínez-Villaseñor M. A Flexible Approach for Human Activity Recognition Using Artificial Hydrocarbon Networks. Sensors. 2016;16:1715 doi: 10.3390/s16111715. [PMC free article] [PubMed] [CrossRef]

20. Maguire D., Frisby R. Comparison of feature classification algorithm for activity recognition based on accelerometer and heart rate data; Proceedings of the 9th Information Technology and Telecommunications Conference (IT&T'09); Dublin, Ireland. 22–23 October 2009; p. 11.

21. Ayu M., Ismail S., Matin A., Montoro T. A Comparison Study of Classifier Algorithms for Mobile-Phone's Accelerometer Based Activity Recognition. Procedia Eng. 2012;41:224–229. doi: 10.1016/j.proeng.2012.07.166. [CrossRef]

22. Wu W., Dasgupta S., Ramirez E.E., Peterson C., Norman G.J. Classification accuracies of physical activities using smartphone motion sensors. J. Med. Internet Res. 2012;14:e130. doi: 10.2196/jmir.2208. [PMC free article] [PubMed] [CrossRef]

23. Mannini A., Intille S.S., Rosenberger M., Sabatini A.M., Haskell W. Activity recognition using a single accelerometer placed at the wrist or ankle. Med. Sci. Sports Exerc. 2013;45:2193–2203. doi: 10.1249/MSS.0b013e31829736d6. [PMC free article] [PubMed] [CrossRef]

24. Reiss A., Hendeby G., Stricker D. A novel confidence-based multiclass boosting algorithm for mobile physical activity monitoring. Pers. Ubiquitous Comput. 2014;19:105–121. doi: 10.1007/s00779-014-0816-x. [CrossRef]

25. Reiss A., Weber M., Stricker D. Exploring and extending the boundaries of physical activity recognition; Proceedings of the 2011 IEEE International Conference on Systems, Man, and Cybernetics (ICSMC'11); Anchorage, AK, USA. 9–12 October 2011; pp. 46–50.

26. Kim H.J., Lee J.S., Yang H.S. Human Action Recognition Using a Modified Convolutional Neural Network; Proceedings of the 4th International Symposium on Neural Networks: Part II—Advances in Neural Networks (ISSN'07); Nanjing, China. 3–7 June 2007; pp. 715–723.

27. Baccouche M., Mamalet F., Wolf C., Garcia C., Baskurt A. Sequential Deep Learning for Human Action Recognition; Proceedings of the 2nd International Conference on Human Behavior Understanding (HBU'11); Amsterdam, The Netherlands. 16 November 2011; pp. 29–39.

28. Ji S., Xu W., Yang M., Yu K. 3D Convolutional Neural Networks for Human Action Recognition. IEEE Trans. Pattern Anal. Mach. Intell. 2013;35:221–231. doi: 10.1109/TPAMI.2012.59. [PubMed] [CrossRef]

29. Charalampous K., Gasteratos A. On-line deep learning method for action recognition. Pattern Anal. Appl. 2014;19:337–354. doi: 10.1007/s10044-014-0404-8. [CrossRef]

30. Jiang W., Yin Z. Human Activity Recognition Using Wearable Sensors by Deep Convolutional Neural Networks; Proceedings of the 23rd ACM International Conference on Multimedia (MM'15); Brisbane, Australia. 26–30 October 2015; pp. 1307–1310.

31. Ordoñez F.J., Roggen D. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. Sensors. 2016;16:115 doi: 10.3390/s16010115. [PMC free article] [PubMed] [CrossRef]

32. Hammerla N.Y., Halloran S., Ploetz T. Deep, Convolutional, and Recurrent Models for Human Activity Recognition using Wearables; Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16); New York, NY, USA. 9–15 July 2016; pp. 1533–1540.

33. Wang Y., Lin J., Annavaram M., Jacobson Q., Hong J., Krishnamachari B., Sadeh N. A Framework of Energy Efficient Mobile Sensing for Automatic User State Recognition; Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys'09); Kraków, Poland. 22–25 June 2009; pp. 179–192.

34. Yan Z., Subbaraju V., Chakraborty D., Misra A., Aberer K. Energy-Efficient Continuous Activity Recognition on Mobile Phones: An Activity-Adaptive Approach; Proceedings of the 2012 16th Annual International Symposium on Wearable Computers (ISWC'12); Newcastle, UK. 18–22 June 2012; pp. 17–24.

35. Khan A., Siddiqi M., Lee S.W. Exploratory Data Analysis of Acceleration Signals to Select Light-Weight and Accurate Features for Real-Time Activity Recognition on Smartphones. Sensors. 2013;13:13099–13122. doi: 10.3390/s131013099. [PMC free article] [PubMed] [CrossRef]

36. Morillo L., Gonzalez-Abril L., Ramirez J., de la Concepcion M. Low Energy Physical Activity Recognition System on Smartphones. Sensors. 2015;15:5163–5196. doi: 10.3390/s150305163. [PMC free article] [PubMed] [CrossRef]

37. Google Trends Web Search Interest: Weka vs. Scikit-Learn. 2016. [(accessed on 14 April 2016)]. Available online: https://www.google.com/trends/explore#q=725%2Fm%2F0h97pvq%2C%20%2Fm%2F0b2358.

38. Bulling A., Blanke U., Schiele B. A tutorial on human activity recognition using body-worn inertial sensors. ACM Comput. Surv. (CSUR) 2014;46:33. doi: 10.1145/2499621. [CrossRef]

39. Reiss A., Stricker D. Towards global aerobic activity monitoring; Proceedings of the 4th International Conference on PErvasive Technologies Related to Assistive Environments (PETRA'11); Heraklion, Greece. 25–27 May 2011.

40. Reiss A., Stricker D. Introducing a new benchmarked dataset for activity monitoring; Proceedings of the 2012 16th Annual International Symposium on Wearable Computers (ISWC'12); Newcastle, UK. 18–22 June 2012; pp. 108–109.

41. Reiss A., Stricker D. Creating and benchmarking a new dataset for physical activity monitoring; Proceedings of the 5th International Conference on Pervasive Technologies Related to Assistive Environments (PETRA'12); Heraklion, Greece. 6–8 June 2012; pp. 40:1–40:8.

42. Reiss A., Stricker D., Lamprinos I. An integrated mobile system for long-term aerobic activity monitoring and support in daily life; Proceedings of the 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'12); Liverpool, UK. 25–27 June 2012; pp. 2021–2028.

43. Reiss A., Stricker D. Aerobic activity monitoring: Towards a long-term approach. Univ. Access Inf. Soc. 2013;13:101–114. doi: 10.1007/s10209-013-0292-5. [CrossRef]

44. Ainsworth B., Haskell W., Whitt M., Irwin M., Swartz A., Strath S., O'Brien W., Bassett D., Schmitz K., Emplaincourt P., et al. Compendium of physical activities: An update of activity codes and MET intensities. Med. Sci. Sports Exerc. 2000;32:498–516. doi: 10.1097/00005768-200009001-00009. [PubMed] [CrossRef]

45. Baldominos A., Sáez Y., Isasi P. Feature Set Optimization for Physical Activity Recognition Using Genetic Algorithms; Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO'15); Madrid, Spain. 11–15 July 2015; pp. 1311–1318.

46. Guyon I., Elisseeff A. An Introduction to Variable and Feature Selection. J. Mach. Learn. Res. 2003;3:1157–1182.

47. Margarito J., Helaoui R., Bianchi A., Sartor F., Bonomi A. User-Independent Recognition of Sports Activities From a Single Wrist-Worn Accelerometer: A Template-Matching-Based Approach. IEEE Trans. Biomed. Eng. 2016;63:788–796. doi: 10.1109/TBME.2015.2471094. [PubMed] [CrossRef]

48. Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., et al. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 2011;12:2825–2830.

49. Wu X., Kumar V., Ross Quinlan J., Ghosh J., Yang Q., Motoda H., McLachlan G.J., Ng A., Liu B., Yu P.S., et al. Top 10 Algorithms in Data Mining. Knowl. Inf. Syst. 2007;14:1–37. doi: 10.1007/s10115-007-0114-2. [CrossRef]

50. Nef T., Urwyler P., Büchler M., Tarnanas I., Stucki R., Cazzoli D., Müri R., Mosimann U. Evaluation of Three State-of-the-Art Classifiers for Recognition of Activities of Daily Living from Smart Home Ambient Data. Sensors. 2015;15:11725–11740. doi: 10.3390/s150511725. [PMC free article] [PubMed] [CrossRef]

51. Cover T., Hart P. Nearest Neighbor Pattern Classification. IEEE Trans. Inf. Theory. 2006;13:21–27. doi: 10.1109/TIT.1967.1053964. [CrossRef]

52. Lombriser C., Bharatula N.B., Roggen D., Tröster G. On-body Activity Recognition in a Dynamic Sensor Network; Proceedings of the ICST 2nd International Conference Body Area Networks (BodyNets'07); Brussels, Belgium. 11–13 June 2007; pp. 17:1–17:6.

53. Mitchell T.M. Machine Learning. McGraw-Hill; New York, NY, USA: 1997.

54. Chan T.F., Golub G.H., LeVeque R.J. Updating Formulae and a Pairwise Algorithm for Computing Sample Variances. Stanford University; Stanford, CA, USA: 1979. Technical Report.

55. Fisher R.A. The Use of Multiple Measurements in Taxonomic Problems. Ann. Eugen. 1936;7:179–188. doi: 10.1111/j.1469-1809.1936.tb02137.x. [CrossRef]

56. Johnson R.A., Wichern D.W. Applied Multivariate Statistical Analysis. Prentice-Hall; Upper Saddle River, NJ, USA: 1988.

57. Bottou L., Bousquet O. Advances in Neural Information Processing Systems. Volume 20. NIPS Foundation; La Jolla, CA, USA: 2008. The Tradeoffs of Large Scale Learning; pp. 161–168.

58. Ruppert D. Efficient Estimators from a Slowly Convergent Robbins-Monro Procedure. School of Operations Research and Industrial Engineering, Cornell University; Ithaca, NY, USA: 1988. Technical Report.

59. Vapnik V.N. The Nature of Statistical Learning Theory. Springer-Verlag New York; New York, NY, USA: 1995.

60. Cortes C., Vapnik V. Support-Vector Networks. Mach. Learn. 1995;20:273–297. doi: 10.1007/BF00994018. [CrossRef]

61. Quinlan J.R. Induction of Decision Trees. Mach. Learn. 1986;1:81–106. doi: 10.1007/BF00116251. [CrossRef]

62. Mingers J. An Empirical Comparison of Selection Measures for Decision-Tree Induction. Mach. Learn. 1989;3:319–342. doi: 10.1007/BF00116837. [CrossRef]

63. Breiman L. Random Forests. Mach. Learn. 2001;45:5–32. doi: 10.1023/A:1010933404324. [CrossRef]

64. Geurts P., Ernst D., Wehenkel L. Extremely Randomized Trees. Mach. Learn. 2006;63:3–42. doi: 10.1007/s10994-006-6226-1. [CrossRef]

65. 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. doi: 10.1006/jcss.1997.1504. [CrossRef]

66. Bauer E., Kohavi R. An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. Mach. Learn. 1999;36:105–139. doi: 10.1023/A:1007515423169. [CrossRef]

67. Breiman L. Bagging Predictors. Mach. Learn. 1996;24:123–140. doi: 10.1007/BF00058655. [CrossRef]

68. Hastie T., Rosset S., Zhu J., Zou H. Multi-class AdaBoost. Stat. Its Interface. 2009;2:349–360. doi: 10.4310/SII.2009.v2.n3.a8. [CrossRef]

69. Rumelhart D.E., Hinton G.E., Williams R.J. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1. MIT Press; Cambridge, MA, USA: 1986. Learning Internal Representations by Error Propagation; pp. 318–362.

70. Le Q.V., Zou W.Y., Yeung S.Y., Ng A.Y. Learning Hierarchical Invariant Spatio-temporal Features for Action Recognition with Independent Subspace Analysis; Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11); Colorado Springs, CO, USA. 20–25 June 2011; pp. 3361–3368.

71. Dean J., Corrado G., Monga R., Chen K., Devin M., Mao M., Ranzato M., Senior A., Tucker P., Yang K., et al. Advances in Neural Information Processing Systems. Volume 25. NIPS Foundation; La Jolla, CA, USA: 2012. Large Scale Distributed Deep Networks; pp. 1232–1240.

72. Deng L., Yu D. Deep Learning: Methods and Applications. Found. Trends Signal Process. 2014;7:197–387. doi: 10.1561/2000000039. [CrossRef]

73. Abadi M., Agarwal A., Barham P., Brevdo E., Chen Z., Citro C., Corrado G.S., Davis A., Dean J., Devin M., et al. TensorFlow: Large-Scale Machine Learning on Heterogeneus Distributed Systems. Google Research; Mountain View, CA, USA: 2015. Technical Report.

74. Allwein E.L., Schapire R.E., Singer Y. Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers. J. Mach. Learn. Res. 2001;1:113–141.

75. Landgrebe T., Duin R.P.W. Approximating the multiclass ROC by pairwise analysis. Pattern Recognit. Lett. 2007;28:1747–1758. doi: 10.1016/j.patrec.2007.05.001. [CrossRef]

76. Arif M., Kattan A. Physical Activities Monitoring Using Wearable Acceleration Sensors Attached to the Body. PLoS ONE. 2015;10:e0130851 doi: 10.1371/journal.pone.0130851. [PMC free article] [PubMed] [CrossRef]

77. Gupta P., Dallas T. Feature Selection and Activity Recognition System Using a Single Triaxial Accelerometer. IEEE Trans. Biomed. Eng. 2014;61:1780–1786. doi: 10.1109/TBME.2014.2307069. [PubMed] [CrossRef]

78. Shoaib M., Scholten H., Havinga P. Towards physical activity recognition using smartphone sensors; Proceedings of the 10th IEEE International Conference on Ubiquitous Intelligence and Computing and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC); Vietri sul Mare, Italy. 18–21 December 2013; pp. 80–87.

79. Anguita D., Ghio A., Oneto L., Parra X., Reyes-Ortiz J.L. A Public Domain Dataset for Human Activity Recognition Using Smartphones; Proceedings of the 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2013; Bruges, Belgium. 24–26 April 2013.

80. Gyllensten I., Bonomi A. Identifying Types of Physical Activity With a Single Accelerometer: Evaluating Laboratory-trained Algorithms in Daily Life. IEEE Trans. Biomed. Eng. 2011;58:2656–2663. doi: 10.1109/TBME.2011.2160723. [PubMed] [CrossRef]

81. Khan A.M., Lee Y.K., Lee S.Y., Kim T.S. A Triaxial Accelerometer-based Physical-activity Recognition via Augmented-signal Features and a Hierarchical Recognizer. IEEE Trans. Inf. Technol. Biomed. 2010;14:1166–1172. doi: 10.1109/TITB.2010.2051955. [PubMed] [CrossRef]

82. Bao L., Intille S.S. Pervasive Computing. Volume 3001. Springer; Berlin, Germany: 2004. Activity recognition from user-annotated acceleration data; pp. 1–17. Lecture Notes in Computer Science.

83. Kwapisz J.R., Weiss G.M., Moore S.A. Activity Recognition Using Cell Phone Accelerometers. ACM SIGKDD Explor. Newsl. 2011;12:74–82. doi: 10.1145/1964897.1964918. [CrossRef]