IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 14, 2019, accepted May 29, 2019, date of publication June 5, 2019, date of current version June 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2921096

Outlier Detection in Wearable Sensor Data for Human Activity Recognition (HAR) Based on DRNNs

MARIO MUNOZ-ORGANERO, **(Member, IEEE)**
Telematics Engineering Department, UC3M-BS Institute of Financial Big Data, Universidad Carlos III de Madrid, 28911 Leganes, Spain
e-mail: munozm@it.uc3m.es

This work was supported by the "ANALYTICS USING SENSOR DATA FOR FLATCITY" Project (MINECO/ ERDF, EU) funded in part by the Spanish Agencia Estatal de Investigación (AEI) under Grant TIN2016-77158-C4-1-R and in part by the European Regional Development Fund (ERDF).

ABSTRACT Wearable sensors provide a user-friendly and non-intrusive mechanism to extract user-related data that paves the way to the development of personalized applications. Within those applications, human activity recognition (HAR) plays an important role in the characterization of the user context. Outlier detection methods focus on finding anomalous data samples that are likely to have been generated by a different mechanism. This paper combines outlier detection and HAR by introducing a novel algorithm that is able both to detect information from secondary activities inside the main activity and to extract data segments of a particular sub-activity from a different activity. Several machine learning algorithms have been previously used in the area of HAR based on the analysis of the time sequences generated by wearable sensors. Deep recurrent neural networks (DRNNs) have proven to be optimally adapted to the sequential characteristics of wearable sensor data in previous studies. A DRNN-based algorithm is proposed in this paper for outlier detection in HAR. The results are validated both for intra- and inter-subject cases and both for outlier detection and sub-activity recognition using two different datasets. A first dataset comprising 4 major activities (walking, running, climbing up, and down) from 15 users is used to train and validate the proposal. Intra-subject outlier detection is able to detect all major outliers in the walking activity in this dataset, while inter-subject outlier detection only fails for one participant executing the activity in a peculiar way. Sub-activity detection has been validated by finding out and extracting walking segments present in the other three activities in this dataset. A second dataset using four different users, a different setting and different sensor devices is used to assess the generalization of results.

INDEX TERMS Human activity recognition, wearable sensors, outlier detection, machine learning, deep learning, recurrent neural networks, LSTMs.

I. INTRODUCTION

Human Activity Recognition (HAR) is a research area which focuses on automatically detecting/assessing what a particular human user is doing based on related sensor data. Recognizing what the user is doing provides valuable contextual information to help user-centered applications to better adapt to the user needs in many different areas. In fact, HAR has been successfully applied to several areas such as sport training, remote health monitoring, health self-management, military applications, gaming, home behavior analysis, gait analysis and gesture recognition [3], [13]. Based on the granularity of the activity being recognized, activities could be broken into movements or gestures or grouped together into sequences of activities (complex or composed activities). Based on the type of sensor used, the activities could be recognized using wearable sensors, sensors attached to objects which are handled by the user or sensors in the environment (such as cameras and Bluetooth beacons). Different machine learning based algorithms have been used over the past decades to solve the human activity recognition problem [1]–[4]. These algorithms are trained (either in a supervised or semi-supervised way [1]) with labeled data assigning raw sensor data fragments to a particular activity (class).

The associate editor coordinating the review of this manuscript and approving it for publication was Zahid Akhtar.

Several datasets are publically available (such as those found in [5]–[7]) with labeled sensor data which have been commonly used in previous research studies. These datasets assign labels to data which may have information from secondary activities (either executed in a sequential or overlapping way). Training the machine learning based classification algorithms including these secondary-activity sensor data introduces training errors which have an impact in the performance of the recognition phase (when assigning a class or activity to a new non-labeled segment of data).

In this paper, a novel technique for detecting anomalous segments in the raw temporal sequences of sensor data is presented (fragments of data in the temporal sequence recorded while performing a particular activity which deviate so much from other fragments as to arouse suspicions that they were generated by the execution of a different activity). These anomalous sections (or outliers) could be removed from the training data to better train the machine learning algorithms. A second contribution of the paper is using the proposed method to extract (or detect) the execution of particular secondary activities inside the major activity being performed. The proposed algorithm uses a temporal series self-characterization mechanism based on the use of a Deep Recurrent Networks (DRNN) implemented based on Long Short Term Memory (LSTM) cells in order to predict upcoming segments of temporal data. If the statistical information of the upcoming data segment is similar to the recent past (the user is performing the same activity) the prediction will be similar (by using a similarity function) to the real data. If a second sub-activity is performed inside the main activity, the prediction based on the recent sensor data for the main activity will provide a worse similarity to the data generated from the secondary sub-activity. Extracting the anomalous data segments from a particular activity and using them as inputs to the proposed algorithm trained for different activities will provide a mechanism to classify that secondary sub-activity. The proposed algorithm is validated based on wearable sensor data (using a single tri-axial accelerometer) using data for 4 different activities (walking, climbing up stairs, climbing down stairs and running) and using two different datasets and two different scenarios. The results show that it is both possible to detect anomalous data (secondary activities) and to extract segments of a particular sub-activity present inside the main activity (extracting walking segments while running for example).

The paper is organized as follows. Section I, this section, provides an introduction and motivation and outlines the major contributions of the manuscript. Section II and section III are dedicated to summarize previous related work on the use of deep learning methods and techniques both for Human Activity Recognition (HAR) as well as for outlier detection. Section IV presents the proposed architecture for outlier detection in wearable sensor data for Human Activity Recognition (HAR) based on the use of Deep Recurrent Neural Networks (DRNN) using Long Short Term Memory (LSTM) cells. Section V is dedicated to the description of the datasets used for validation. Results are presented and validated in section VI. Finally, section VII captures the major conclusions of the research.

II. DEEP LEARNING METHODS FOR HAR

A generic formulation for the Human Activity Recognition (HAR) problem, as well as a survey of former methods to solve it was captured in [1]. The general dataflow for HAR comprises several common steps including the activity related data acquisition from sensors when performing the activity, the extraction of relevant features describing the sensed information and the use of a learning method which is trained based on known labeled data and applied to new unknown data for activity recognition [1]. A similar survey on former HAR research studies [2] defined the typical activity recognition chain as a set of the following tasks: raw data acquisition, preprocessing, segmentation, feature extraction and classification. The way in which the features are defined and selected plays a very important role in the final performance of the system. Former studies used two major approaches to extract features from time series data: statistical and structural [1]. Both of them are hand-crafted methods which transform the raw sensed data into particular pre-defined characteristics or descriptors. Hand-made features were formerly used with shallow learning algorithms for classification of activities [1], [2]. Shallow structures could be defined by the low depth of the paths of intermediate trainable units between the input and the output layers [8]. These trainable intermediate units are trained to learn the relationship between the input features and the output class. When the depth of the path grows the machine learning methods turn to Deep Learning architectures. In recent years, deep learning methods have been more and more used for HAR, which achieves unparalleled performance in many areas such as visual object recognition, natural language processing, and logic reasoning [3]. Deep learning can largely relieve the effort on designing features and can learn much more high-level and meaningful features [3]. Using multiple layers of abstraction, deep learning methods are able to learn intricate features representations from raw sensor data and discover the best patterns to improve recognition performance [4]. Deep learning automatic feature learning, by using different layers of abstraction, is very well adapted to HAR since the hierarchical structure of human movements. Human basic movements or gestures are combined into basic activities which in turn are linked to compose more complex activities.

A survey on the use of deep learning methods for HAR can be found in [3]. The authors focus on sensor-based activity recognition. They carry out a summary of existing literature making a characterization based on three major aspects: sensor modality, deep model, and application. In terms of modality, the authors distinguish body worn sensors from other types of sensors such as environmental sensors or sensors attached to objects. In terms of the deep learning models, the authors consider Deep fully-connected Networks (DNNs), Convolutional Neural Networks (CNNs), Recurrent

Neural Networks (RNNs), including Long Short Term Memory cells (LSTMs), Deep Belief Networks and Restricted Boltzmann Machines (DBN/RBMs), Stacked Autoencoders (SAEs) and a hybrid combination of some deep models.

A second review of the use of deep learning methods for HAR can be found in [4]. The authors categorized the studies into generative, discriminative and hybrid methods and highlighted their important advantages. From the different deep learning methods, the authors state that Recurrent Neural Networks (RNNs) are naturally designed for time series data in which sensor data is a prominent part. Several previous studies [9]–[17] have applied different architectural and structural variations in RNNs to achieve optimal recognition performance in HAR. The results depend not only on the selected method but also on the underlying dataset used for training and validation. Hammerla *et al.* [11] where able to outperform other similar studies over the dataset in [6] using a bidirectional LSTM (b-LSTM-S) while the research in [14] used a different architecture based on the combination of Deep Convolutional and LSTM recurrent neural networks for best results over the dataset in [7].

Recent studies such as [18] have used modified deep RNNs architectures in order to facilitate the training of the algorithm using residual connections between stacked cells to avoid gradient vanishing problems. Zhao *et al.* [18] were able to improve previous results over the dataset in [6] around 4% by using these changes.

One of the collateral effects of using deep learning architectures in general is that the number of samples (either labeled in supervised learning or unlabeled in semi-supervised learning) required to train them tends to significantly grow compared with algorithms based on shallow architectures. In order to mitigate this issue, Steven Eyobu and Han [19] proposed a data augmentation mechanism improving the recognition accuracy provided by a LSTM architecture.

Recent publications have also proposed optimizations in the architecture of Deep Recurrent Neural Networks (DRNNs) to achieve better performance. Inoue *et al.* [20] optimized a DRNN to achieve high throughput and a short time of recognition. The authors measured the time required by previous shallow based methods such as SVM and decision trees in order to compute the required hand-crafted features, and showed that an optimization in the DRNN architecture could make the time required by the DRNN to recognize movements an order of magnitude smaller than the feature computation time required by shallow methods. The authors also found out that increasing the depth of the RNN architecture will not always achieve a better performance, and found optimal results for a three-layer architecture for their particular dataset.

In summary, Deep Recurrent Neural Networks (DRNNs) have proven to be able to capture the underlying structural patterns of time series generated from wearable sensors and achieve state of the art results for HAR. This paper proposes a novel architecture which makes use of a deep DRNN based on LSTM cells to automatically characterize the statistical patterns of recent values in a wearable sensor time series data and using them to try to predict the evolution of the time series in the upcoming future. A background of former research studies combining deep learning techniques and outlier detection is first presented in the next section.

III. OUTLIER DETECTION BASED ON DEEP LEARNING

Hawkins [21] provided a definition of an outlier as 'an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism'. Several methods have been proposed since then in order assess those deviations. Ruiz Blázquez *et al.* [22] provided a categorization of anomaly detection algorithms and methods including the following categories: statistical methods, distance-based methods, density-based methods, clustering techniques and adaptations of machine learning algorithms for classification problems to novelty detection (methods such as Neural Networks, and One-class Support Vector Machines). Agrawal and Agrawal [23] provided a deeper study of some of the data mining techniques that could be used to detect the surprising behavior hidden within data such as clustering methods and classifiers. A similar paper in [24] also concluded that anomaly detection has been widely studied by using adaptations of machine learning techniques, where it is also known as outlier detection, deviation detection, novelty detection, and exception mining.

The development of deep learning algorithms and techniques has also been applied for outlier detection in recent years. Xu *et al.* [25] made use of several stacked autoencoders and a single class SVM to detect anomalies in images. By stacking together several autoencoders, the architecture focuses on automatically learning hierarchical features in an unsupervised way. A similar idea based on Deep Structured Energy Based Models (DSEBMs) is presented in [26]. The authors also solve the anomaly detection problem based on the direct modeling of the data distribution with deep learning architectures. The authors state that using Energy Based Models (EBMs) naturally corresponds to identifying data points that are assigned low probability values by the model. Using a trained DSEBM, the samples that are assigned a probability value bellow a pre-chosen threshold will be marked as outliers. Zhou and Paffenroth [27] used a similar method based on denoising autoencoders to detect outliers.

Previously proposed methods in [25]–[27] for outlier detection based on deep learning architectures do not consider the sequential nature of the time series generated by wearable sensors. In a similar way that Deep Recurrent Neural Networks (DRNNs) have outperformed other deep learning architectures for Human Activity Recognition (HAR) as described in the previous section, it is likely that they also perform well for outlier detection in wearable sensor data. This paper proposes and analyses a novel architecture for outlier detection in a HAR using a DRNN based on LSTM cells. If the number of outlier segments in the training data is small, training the algorithm with the entire data sequence will

produce similar results as performing a two-step training based on removing the outliers detected in a first round for a second training. An intra as well as an inter-subject validation has also been performed in order to assess whether the differences in the statistical properties of different participants carrying out the same activity are smaller than those of the same participant interleaving secondary activities inside the main one. An inter-dataset validation has also been performed to assess the generalization of results.

IV. PROPOSED ARCHITECTURE

The proposed architecture is captured in Figure 1. The circles in the lower part of the image represent the recent values of the input signal. In our case, the input signal will be based on the data samples obtained from a sensor in a wearable device although the algorithm could be applied to other types of sequential data as well. Depending on the type of sensor and the way in which the device is attached to the body, a preprocessing phase would be convenient in order to improve the quality of the raw data. In the case of using tri-axial accelerometers (as used in the validation part of this paper), using data from sensors from different datasets using different sampling frequencies, a resampling mechanism should be executed in order to generate data sequences at a predefined rate. The resampling process will also be used to regenerate missing values in the sequence by interpolating the information from the closest data points. Moreover, the acceleration values in the device coordinate system do not provide an optimal representation of the data since the device orientation could vary from user to user, from a data recording from a single user to a different recording for the same user and even for intra-recording samples if the device is not attached completely tight to the body. A better representation of the input acceleration signal is obtained by projecting the raw acceleration values to a geo-referenced coordinate system. Further preprocessing in order to filter noise has not been used in this paper.

The preprocessing algorithm is described as follows:

1. The tri-axial acceleration is sampled at a particular sampling frequency f. Each sample comprises 3 acceleration values (one per axis in the sensor coordinate system). The sample j can be represented as $\overrightarrow{a_t(j)} = \left(a_{tx}(j), a_{ty}(j), a_{tz}(j)\right)$
2. A linear interpolation is applied to generate samples at a frequency f'. In order to generate the sample i at a requency f' we select the closest j and j+1 samples at frequency f and estimate $\overrightarrow{a_t(i)}$ as: $\overrightarrow{a_t(i)} = (t_{j+1} - t_i)\frac{\overrightarrow{a_t(j)}}{t_{j+1}-t_j} + (t_i - t_j)\frac{\overrightarrow{a_t(j+1)}}{t_{j+1}-t_j}$. where t_k is the time at which $\overrightarrow{a_t(k)}$ is sampled.
3. The gravity force vector for the sample number i in the sensor coordinate system could beestimated by using a moving average based low pass filter using the following equation (similar to the method proposed in [28]): $\overrightarrow{g(i)'} = \frac{\sum_{j=i-\frac{N}{2}}^{j=i+\frac{N}{2}} (a_{tx}(j), a_{ty}(j), a_{tz}(j))}{N1}$, where N is the number of samples in the averaging window.
4. The acceleration caused by the movement of the user is estimated according to: $\overrightarrow{a_m(i)} = \overrightarrow{a_t(i)} - \overrightarrow{g(i)'}$
5. $\overrightarrow{a_m(i)}$ could then be divided into vertical and horizontal acceleration components following the Equation: $\overrightarrow{a_{mv}(i)} = \frac{\overrightarrow{a_m(i)} \cdot \overrightarrow{g(i)'}}{\left\|\overrightarrow{g(i)'}\right\|^2} \overrightarrow{g(i)'}$ & $\overrightarrow{a_{mh}(i)} = \overrightarrow{a_m(i)} - \overrightarrow{a_{mv}(i)}$
6. $\vec{a_t}(i)$ could also be divided into vertical and horizontal acceleration components following a similar equation: $\overrightarrow{a_{tv}(i)} = \frac{\overrightarrow{a_t(i)} \cdot \overrightarrow{g(i)'}}{\left\|\overrightarrow{g(i)'}\right\|^2} \overrightarrow{g(i)'}$ & $\overrightarrow{a_{th}(i)} = \overrightarrow{a_t(i)} - \overrightarrow{a_{tv}(i)}$
7. Both $\left\|\overrightarrow{a_{tv}(i)}\right\|$ and $\left\|\overrightarrow{a_{mv}(i)}\right\|$ are geo-referenced acceleration sequences compensating the sensor orientation variations over time.

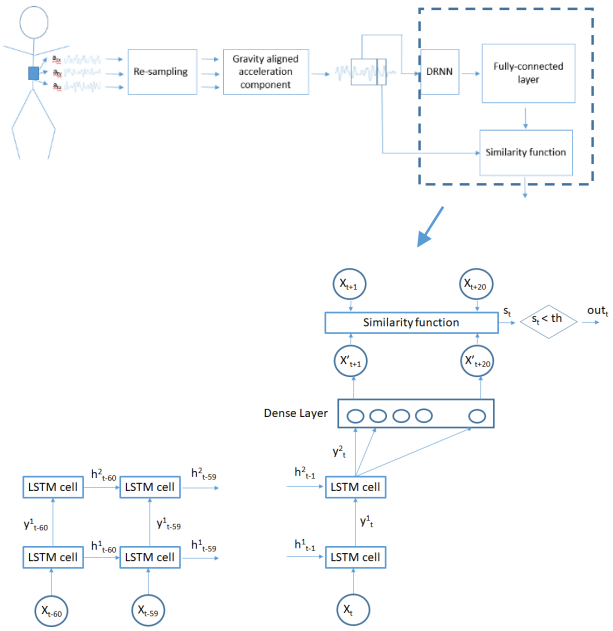

FIGURE 1. Proposed architecture.

The input samples in Figure 1 are fed into a Recurrent Neural Network (RNN). Stacking together several layers, a deep representation will be created. A deeper representation will be able to better adjust to finer details in the data but will also require more samples and more time to train and is prone to show overfitting and generalization issues. In the particular case of [20] the authors found out that increasing the depth of the RNN architecture achieved a better performance only for a three-layer architecture. In the proposed architecture in Figure 1, the output of the DRNN part will be connected to a dense connected layer which will be trained to minimize the mean square error with the upcoming samples in the input (preprocessed) sequence.

Once the architecture in Figure 1 is trained with data segments for a particular activity, outliers will be detected by applying the trained network to each data segment and comparing the similarity of the reconstructed (predicted) output with the real upcoming values in the sequence. Different correlation coefficients could be used as a similarity measure [29]. The Pearson correlation coefficient will be used in the experimental part of this paper following equation 1.

$$r_{x,x'} = \frac{\sum_{i=1}^{M} \left((x_i - \hat{x}) \left(x'_i - \hat{x}' \right) \right)}{\sqrt{\sum_{i=1}^{M} \left(x_i - \hat{x} \right)^2 \sum_{i=1}^{M} \left(x'_i - \hat{x}' \right)^2}} \quad (1)$$

where $\hat{x}$ represents the mean value for sequence x. Calculated as $\frac{1}{M} \sum_{i=1}^{M} x_i$

The training of the architecture in Figure 1 will be optimally done with single activity data in order to create a model adapted to that activity. Using input data which contains anomalous segments of a different activity (such as walking segments of data while running, or walking on a flat stretch interconnecting two flights of stairs in a climbing up activity) will make the algorithm in Figure 1 to wrongly learn some features from the secondary activity as if belonging to the main one. The algorithm in Figure 1 can be executed twice in order to minimize the impact of anomalous data in the training of a particular activity. The first execution will be trained with the all the data samples tagged as belonging to a particular activity in the dataset (including those corresponding to anomalous data). Once trained, in this first execution, the algorithm will be used to discard the data whose reconstructed similarity with the real data is below a certain threshold. The training in the second execution of the algorithm will be based on the remaining data in order to maximize the learning of activity dependent features while minimizing the learning of features associated to other activities.

The length of the input sequence as well as the output prediction could be adapted depending on the characteristics of the data. In the experimental part of this paper, 60 data samples from the sensor data have been used since they were able to capture at least one period of the data sequence in the periodic activities used (walking, climbing up, climbing down and running). The output length has been limited to one third of the input (20 samples) as a balanced solution which combines having enough information in the input to properly reconstruct the output (to limit the maximum length) but representing a significant segment of data to differentiate among different activities (to limit the minimum length).

The basic element in the RNN is a Long Short Term Memory (LSTM) cell [30]. The architecture for a single LSTM cell is captured in Figure 2. Considering our particular case of a recurrent length of 60 input samples, the size of the memory cells has been limited to 50 in order to avoid overfitting.

The structure in Figure 2 is defined using the following equations:

$$f_t = \sigma \left(W_f x_t + U_f h_{t-1} + b_f \right) \quad (2)$$

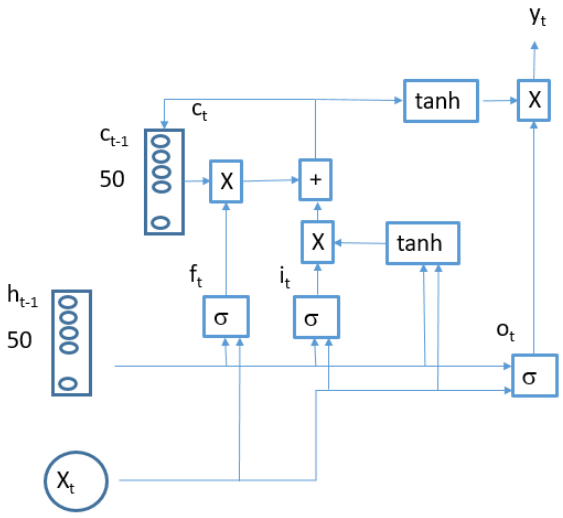

FIGURE 2. LSTM cell architecture.

$$i_t = \sigma \left(W_i x_t + U_i h_{t-1} + b_i \right) \quad (3)$$

$$o_t = \sigma \left(W_o x_t + U_o h_{t-1} + b_o \right) \quad (4)$$

$$c_t = f_t \times c_{t-1} + i_t \times tanh \left(W_c x_t + U_c h_{t-1} + b_c \right) \quad (5)$$

$$h_t = y_t = o_t \times tanh \left(c_t \right) \quad (6)$$

The $\times$ symbol represents a bitwise multiplication and the σ symbol represents the sigmoid function. All the LSTM cells in each layer share the same weights (W_f, U_f, W_i, U_i, W_o, U_o, W_c, U_c and biases b_f, b_i, b_o, b_c). The output of each LSTM cell is used as the input for the next cell. Moreover, the outputs of each LSTM cell in the first layer in the DRNN are fed as inputs to the second layer.

V. DESCRIPTION OF THE DATASETS

A. REALWORLD (HAR) DATASET

The first dataset used for the validation of the architecture presented in Figure 1 is the RealWorld (HAR) dataset in [5]. Fifteen subjects (age 31.9± 12.4, height 173.1 ± 6.9, weight 74.1 ± 13.8, eight males and seven females) performed 8 different activities (sampled at 50 Hz):

- climbing stairs down (A1)
- and up (A2),
- jumping (A3),
- lying (A4),
- standing (A5),
- sitting (A6),
- running/jogging (A7),
- and walking (A8)

From the 8 activities we have selected activities A1, A2, A7 and A8 for the validation experiment in this paper. Some running/jogging sections contain several walking segments inside (as can be appreciated in the video recordings in the dataset) and are therefore a case of interest to apply the architecture proposed in this paper in order to detect particular secondary activities hidden in the data of a different main activity. The same happens for climbing up and down

segments, more prominently those recorded climbing up and down stairs with flat segments interconnecting stair fragments. Moreover, the results reported in [5] for the confusion matrix (applying several classification algorithms [5]), using the same dataset, showed that many walking segments were classified as climbing up and down and vice-versa (showing that the statistical characterization of these activities shared similar properties and that there are some inter-activity data mixed within the main activities in the dataset).

For each participant, the dataset recorded the time series provided by sensors in 7 different body locations:

- chest (P1),
- forearm (P2),
- head (P3),
- shin (P4),
- thigh (P5),
- upper arm (P6),
- and waist (P7).

The experimental part for the validation of the architecture in Figure 1 has used the accelerometer sensor in the waist device according to previous studies [31], [32], which identified the waist as the most suitable location at which the acceleration patterns better generate similar data for different participants.

The dataset combined not only a wide variety of participants but also a set of different settings in which the sensor data was recorded for each participant. Some recordings were taken in an urban environment while others were recorded in the countryside or in a rural area. Some data was taken outdoors but for some cases the recordings were taken indoors (such as climbing up and down stairs in a building or running on a treadmill). The dataset captures therefore a significant inter-user diversity.

B. THE OPPORTUNITY DATASET

The opportunity dataset [6] contains the information of 4 different participants executing different activities while wearing a set of sensors over 5 different runs. This dataset is of particular interest in order to validate the generalization of results (using a different set of users, sensors and setting to validate the algorithm trained for the users in the dataset in the previous section) since one of the sensors is located in a similar location as the one used in the previous dataset (accelerometer 11, columns 5 to 7 in the dataset files) and one of the activities is similar to the one used in the training of the algorithm (walking outdoors).

Each run consists of the sequential execution of different common activities in a kitchen environment such as preparing and drinking coffee or cleaning up the kitchen. In a particular point in the sequence, the participants are asked to go outside and have a walk around the building. Except for this walking outside the building activity, the rest of the activities only involve walking very short distances inside the kitchen.

The sampling frequency in this dataset is 30 Hz. A resampling to a 50 Hz frequency has to be performed in order to be able to use the algorithm in Figure 1 trained with the information in the previous dataset for the users in this dataset.

VI. RESULTS

The architecture proposed in section IV has been validated using the datasets described in section V. Both an intra-subject and an inter-subject validation for a given dataset and both an intra-dataset and an inter-dataset analysis have been performed. Intra-subject validation uses the data recorded from a single participant to train the architecture in Figure 1 and the data from the same participant in order to both detect outliers in a particular activity and to find hidden secondary activity segments in a different activity recording. Inter-subject intra-dataset validation uses the data of all participants expect one for training and the data of the left aside subject for detection. Due to the diversity included in the dataset in [5], as previously described, the intra-subject intra-dataset validation is expected to show better results since some peculiar characteristics of some users are not included in the data from the rest of participants. Inter-subject inter-dataset will use all the participants in the dataset in [5] to train the architecture in Figure 1 and the participants in the dataset in [6] for validation.

The following values will be used to compare the results with previous studies:

$$precision = \frac{TP}{TP + FP} \tag{7}$$

$$recall = \frac{TP}{TP + FN} \tag{8}$$

$$F_1 score = 2 \times \frac{precision \times recall}{precision + recall} \tag{9}$$

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{10}$$

where TP (true positive) is the number of samples from the objective class/activity correctly classified/detected, TN (true negative) captures the number of the samples from non-objective classes/activities correctly detected as such, FP (false positive) represents the number of samples from non-objective classes/activities classified as belonging to the objective class/activity, and finally, FN (false negative) captures the number of samples from the objective class/activity wrongly classified.

The implementation of the architecture in Figure 1 has been done in Python, using the Keras library to train the model. The following Python code describes the details of the implemented model:

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
model_w1 = Sequential()
model_w1.add(LSTM(units=50, return_sequences=True,
input_shape=(input.shape[1], 1)))
model_w1.add(Dropout(0.2))
```

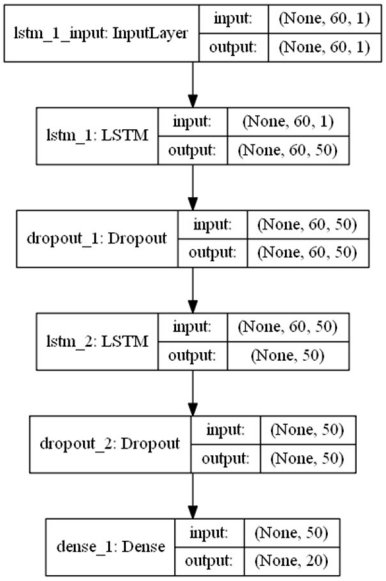

FIGURE 3. Graphical representation of the implemented model.

model_w1.add(LSTM(units=50, return_sequences=False))
model_w1.add(Dropout(0.2))
model_w1.add(Dense(units = 20))
model_w1.compile(optimizer = 'adam', loss = 'mean_squared_error')
model_w1.fit(input, output, epochs = 200, batch_size = 32)

The *w1* suffix refers to the model trained using the data of the walking activity for participant 1. Different models have been trained for each particular scenario. A dropout layer has been added in order to optimize the convergence of the model. The number of epochs has been selected so that the residual error value did not improve above a threshold value of 10^{-5}. The mean squared error function has been used in conjunction with the Adam optimizer.

The graphical representation of the model is captured in Figure 3.

The input data is segmented into 60-sample windows (containing 1.2 seconds of acceleration data) and the algorithm is trained to reconstruct a 20-sample segment (0.4 seconds of data). These values have been selected so that at least a full period of the walking activity is contained in the training segment for the slowest participant and so that there is a significant part of the sequence to be predicted. If the length of the reconstructed sequence is short, the discrimination among different activities will show poor results. If the length is high, the algorithm will fail in providing a good estimation even for the activity used during the training phase. A length of 20 samples (0.4 seconds) has been empirically selected.

The architecture proposed in section IV has a double purpose as previously captured: to be able to detect outlier segments in the execution of a main activity and to be able to find particular sub-activity fragments in a different main activity. The results for both cases for intra and inter-subject scenarios are captured in different sub-sections.

A. INTRA-SUBJECT OUTLIER DETECTION

Intra-subject outlier detection is based on using the acceleration data from each user to detect outliers for that particular user. The architecture proposed in section IV, when trained with the data of a particular user for a particular activity, will learn the characteristic patterns in the data in order to minimize the error when predicting the upcoming segment of data for that particular participant. If the trained architecture is presented with a segment of acceleration data from a different activity, the expected quality of the predicted sequence will decrease. In that way, comparing the Pearson correlation coefficient between the predicted data and the real data for the upcoming data segment will provide a mechanism to estimate whether the current data belongs to the activity used to train the algorithm or to a different one. As in previous outlier detection algorithms, the dataset used in the training phase should optimally be outlier-free, containing only data for the particular activity to be modeled. In the particular case of the dataset used in the validation experiment in this paper [5] the data labeled as belonging to each activity contained several segments corresponding to different activities. In order to cleanse the data from potential outlier segments, the algorithm proposed in section IV can be executed in 2 iterations. A first iteration is executed in order to remove the segments which show a lower reconstructed similarity as compared to the majority of the segments in the dataset. A percentage value could be used for selecting the most similar data segments. In our case, the Pearson correlation coefficient in the reconstruction below a similarity threshold has been used (setting the threshold in $r = 0.8$ for the results presented in this section; other values such as $r = 0.7$ and $r = 0.9$ have provided similar results). Once the estimated outlier segments have been removed, a second iteration with the remaining data is executed in order to train the architecture in section IV for a particular activity.

In order to validate the intra-subject outlier detection results, the walking activity for each participant has been selected. The dataset in [5] uses several accelerometers attached to different parts of the body in the recording of the data. The accelerometer sensor in the waist device has been used in this paper according to previous studies [31], [32], The data for the walking activity for each participant has been used to train the algorithm in section IV and then used to detect outliers for the same participant. Table 1 captures the results for the number of outliers detected (adding the results obtained for all the participants) using 2 different similarity thresholds and both for the case in which outliers are removed before training (2 iterations) and when all the data segments for each participant are used (single iteration).

The results in Table 1 show that removing outliers before training the architecture in section IV is able to detect more outlier segments for the same similarity threshold. This result is intuitive since training the algorithm with outlier corrupted data will make the architecture in Figure 1 to learn some particular patterns for outlier segments and therefore, those segments will no longer be detected as outliers.

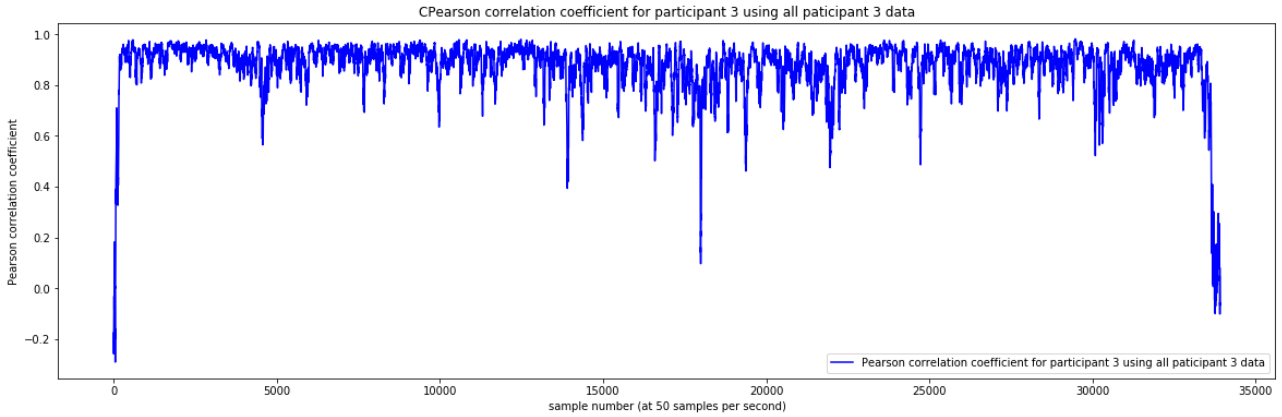

FIGURE 4. Single iteration similarities for participant 3.

TABLE 1. Number of outlier detected for 2 different thresholds.

Type of outlier	rth=0.3	rth=0.4
Single iteration	38	46
2 iterations	46	53
Common outliers	35 (92%)	39 (85%)

TABLE 2. Major outlier types detection for intra-subject single iteration rth = 0.3.

Type of outlier	Detected	Non detected	recall
Turn around	8	3	0.73
Full stop	30	0	1
Significant slow down	3	0	1

TABLE 3. Major outlier types detection for intra-subject 2 iterations rth = 0.3.

Type of outlier	Detected	Non detected	recall
Turn around	11	0	1
Full stop	30	0	1
Significant slow down	3	0	1

TABLE 4. Turn around detection for different r values.

Detected (out of 11)	r=0.3	r=0.4	r=0.5
Single iteration	8	8	11
2 iterations	11	11	11

In the particular case of the dataset in [5], the number of outliers is relatively small and the results in Table 1 show that many of the outliers detected after a 2 iteration process are also detected if no outlier pre-detection and removing is performed (for both similarity thresholds).

Table 2 captures the detailed analysis of the detection (in a single iteration) of 3 particular outlier types in the dataset which are common among several participants: turning around, being stopped, and significantly slowing down the pace without fully stopping. A threshold for the Pearson correlation coefficient of 0.3 has been used. Table 3 captures the same results for the case in which pre-estimated outliers are removed first (i.e. 2 iterations are performed for the architecture in section IV).

In the dataset [5], 11 out of the 15 participants turned around when walking. Using a threshold of 0.3 for the similarity in the reconstruction of the upcoming data, only 8 out of 11 turn-around segments were identified as outliers if no outlier pre-detection and removal is performed. The 11 segments were detected as outliers after the removal of pre-detected outliers (2 iterations). Table 4 shows that the similarity threshold has to be raised to 0.5 in order to be able to detect the 11 turn-around segments in the case of no outlier pre-filtering. Raising the similarity threshold will increase the number of false positive segments detected as outliers.

Tables 2 and 3 show that being completely stopped and very slow walking segments are so different in characteristics to normal walking segments that all of the cases are detected by both approaches.

In order to illustrate the difference in the results when pre-filtering outlier segments, the output of the algorithms, both in terms of the Pearson correlation coefficient and of the outliers detected, for participant 3 are captured in Figures 4 to 7.

Figure 4 shows the similarity (as measured by the Pearson correlation coefficient), with no outlier pre-filtering, when using the 60 previous samples at each particular sample to predict the upcoming 20 samples. The x axis shows the sample number (sampled at 50 Hz). There are 2 clear outlier regions at the beginning and at the end of the recording corresponding to fragments in which the user is fully stopped. There is also a prominent fragment towards the middle of the recording which corresponds to the turn-around segment.

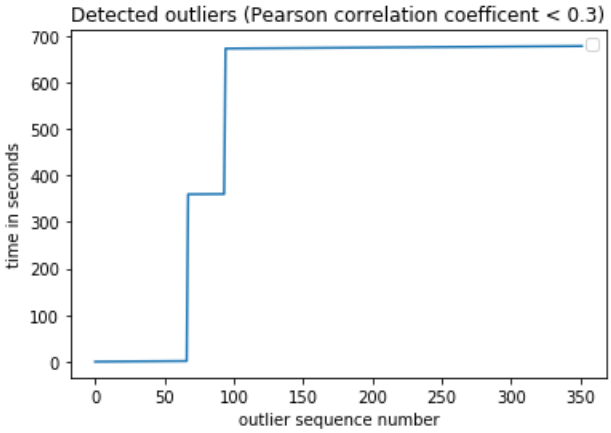

FIGURE 5. Single iteration outliers for participant 3.

Figure 5 shows all the outliers detected and the particular time in the recording in which they are detected. The turn-around region corresponds to a fragment around 360 seconds from the start of the recording (6 minutes).

Figures 6 and 7 show the results for the case in which an outlier pre-filtering is performed. The visual observation of the differences between Figures 4 and 6 is that the regions reconstructed with high similarity to the real data tend to increase their similarity and the regions with low reconstruction similarity (outliers) also emphasize their behavior. In the case of using outlier pre-filtering, two new segments are detected with a similarity bellow 0.3 as shown in Figure 7. These 2 new segments show a similar difference in time with respect to the turn-around point. This is intuitive since both correspond to the same part of the walking path when traversed in both directions which is consistent with the dataset used.

The same type of results for participant 11 are captured in Figures 8 to 10. In this case there is a clear difference in the length of the regions detected as outliers which better encompasses the entire duration of each outlier.

B. INTER-SUBJECT OUTLIER DETECTION

In this section, the data used to train the algorithm will be the complementary set as in section A. Now, the data of all the walking activity recordings for all participants except one are going to be used to train the algorithm to try to reconstruct generic-user walking segments. The data for the left aside participant will be used in the outlier detection phase. The training will be based on the 2-iteration variant previously described since it has shown a better performance in the previous section.

Table 5 captures a comparison of the results in table 1 (for the intra-user case) with the inter-user training in this section (for the same 2 similarity thresholds). The inter-subject case shows a slightly bigger number for the detected outliers. This result is intuitive since the particular way in which the participant used in the outlier detection phase tended to walk may not be exactly the same as the way in which the other participants used to train the algorithm did. Therefore, the algorithm will detect as outliers those particular walking segments which are peculiar to the participant used in the detection phase. Table 5 also shows that the majority of the outliers detected (more than 80%) are commonly detected by the intra and inter-subject scenarios. Table 6 shows a slightly better performance of the inter-subject training case than Table 2 for the intra-subject case with no outlier pre-processing and removal. However, the results in Table 6 are slightly worse than those in Table 3 for the intra-subject case with a previous outlier pre-processing and removal.

The results in Table 7 also show a peculiar characteristic for the inter-subject case. There is one particular user for which the turn-around outlier is not properly detected even if the similarity threshold is raised to the 0.5 level. In a generic dataset, it is possible to find users which may execute a particular activity in a different way as the other users and for those users the algorithm is likely to produce bad results. The results for the Pearson correlation coefficient for this participant (participant 5 in the dataset) are captured in Figure 11. The outliers detected under the 0.5 similarity threshold are captured in Figure 12. Figure 11 shows a poor similarity of the way in which the user walks compared to the rest of the users, so when training the algorithm skipping this user's data the algorithm will not be able to accurately reconstruct acceleration segments for this user. The results in Figure 12 show that a single outlier region covers the majority of the entire walking data and since the algorithm identifies the outlier region as the central point, it will not correspond to the turn-around section but to a mixture of different types of outliers.

TABLE 5. Number of outliers detected for 2 different thresholds.

	rth=0.3	rth=0.4
Inter-subject	50	54
Intra-subject 2 iterations	46	53
Common outliers	38 (83%)	43 (81%)

TABLE 6. Major outlier types detection for inter-subject rth = 0.3.

Type of outlier	Detected	Non detected	recall
Turn around	10	1	0.91
Full stop	30	0	1
Significant slow down	3	0	1

TABLE 7. Turn around detection for different r values.

Detected (out of 11)	r=0.3	r=0.4	r=0.5
Inter-subject training	10	10	10

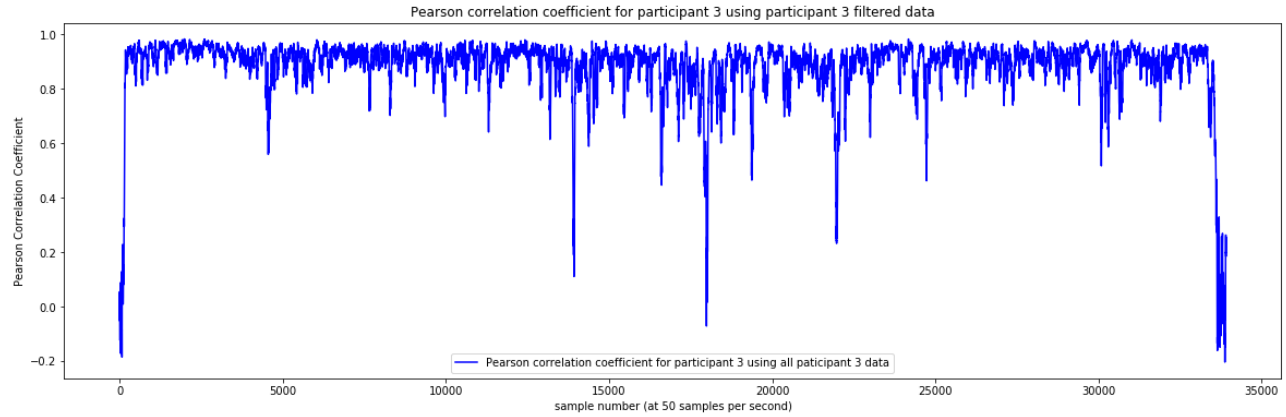

FIGURE 6. iterations (outlier pre-filtering) similarities for participant 3.

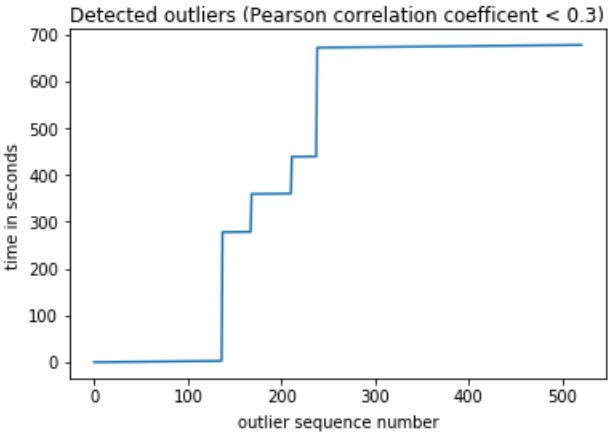

FIGURE 7. iterations (outlier pre-filtering) outliers for participant 3.

C. INTRA-SUBJECT DETECTION OF PARTICULAR SUB-ACTIVITIES

Once the architecture in section IV has been trained to recognize a particular activity for a particular user, the trained algorithm could be used to try to recognize segments of that activity inside a wider execution of a different activity. In this section, the results for detecting walking segments when running, climbing down and climbing up stairs for the case of intra-user training are captured. From the 15 users in the dataset, user 4 has been selected for 2 main reasons: the running data contains 4 walking segments inside and the user is one of the few that performs the climbing up and down activities using an indoors staircase with 24 flat segments connecting stair fragments.

Figure 13 shows the results for the execution of the algorithm proposed in section IV, trained with the walking data for participant 4, to detect similarities in the running activity recording. The 60-sample windows (1.2 seconds) predicted with a similarity higher than 0.9 (as measured by the Pearson correlation coefficient) are shown in blue. The ground truth consisting of the regions visually assessed in the video information in the dataset containing walking data segments is shown in red. All the 4 walking areas are detected as such (at least some sub-regions inside them).

The climbing up and down activity combines 2 major sub-activities inside (together with several outlier fragments such as opening a door). The main sub-activity is composed by the data recorded actually climbing up and down. The second major sub-activity contains acceleration data recorded when walking on the flat segment connecting ascending-descending stair fragments. The results in Table 8 show the number of flat segments detected when climbing down for different similarity threshold values. The algorithm is trained with the walking data for participant 4. Once the algorithm is trained, the similarity metrics is applied to the climbing up and down recordings. Since those recordings contain similar data segments to those found in the walking activity, using a similarity threshold, it should be possible to detect them. Table 8 shows that when the threshold is increased the number of segments detected decreases. The same happens for the detection of false positives (climbing areas detected as walking on a flat surface). By lowering the threshold, we are able to increase the detection rate at the price of getting a higher number of false positives. The recall, precision and F_1 score are also shown in the table. The best F_1 score value is achieved for a threshold similarity value of 0.93. The F_1 score value in this case (0.78) is a bit lower than the value obtained by Sztyler and Stuckenschmidt [5] for the same sensor (located in the waist) and for the walking down data. However, Sztyler and Stuckenschmidt [5] used a 10-fold cross-validation schema in which data fragments from the climbing down dataset are used in the training of the algorithm. The results presented here use the data in the walking dataset file for training and the data in the climbing down file for validation.

Table 9 shows the same results for the climbing up activity. In this case, the similarity between the acceleration data recorded in the flat segments when climbing up stairs and

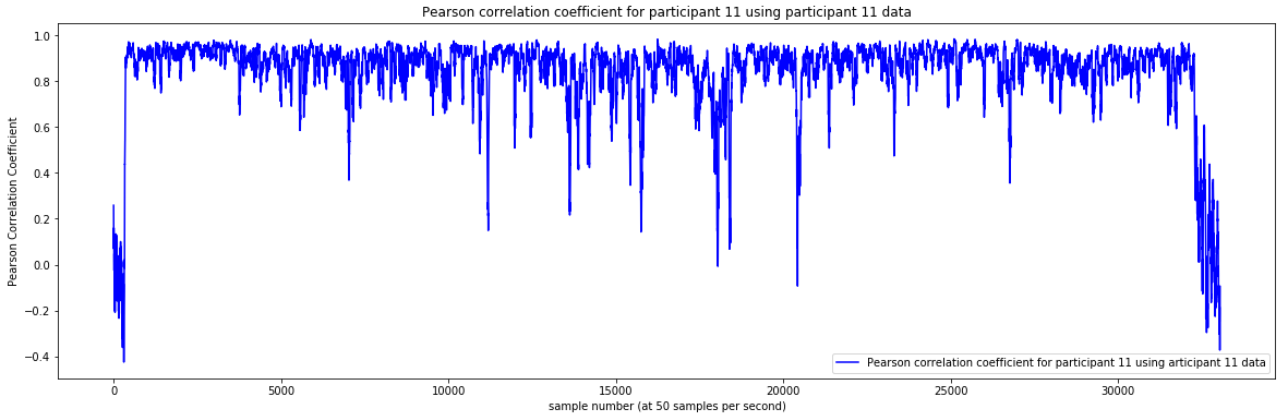

FIGURE 8. Single iteration similarities for participant 11.

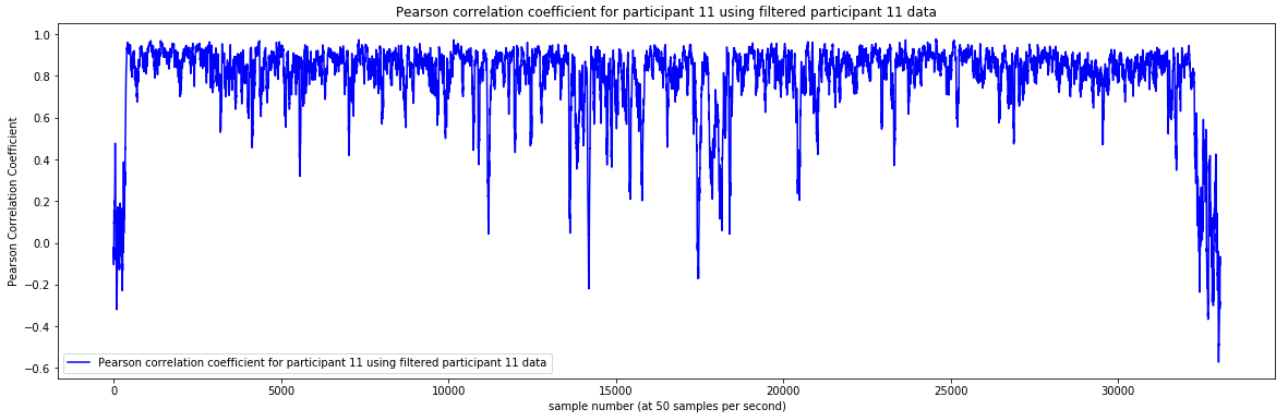

FIGURE 9. iterations (outlier pre-filtering) similarities for participant 11.

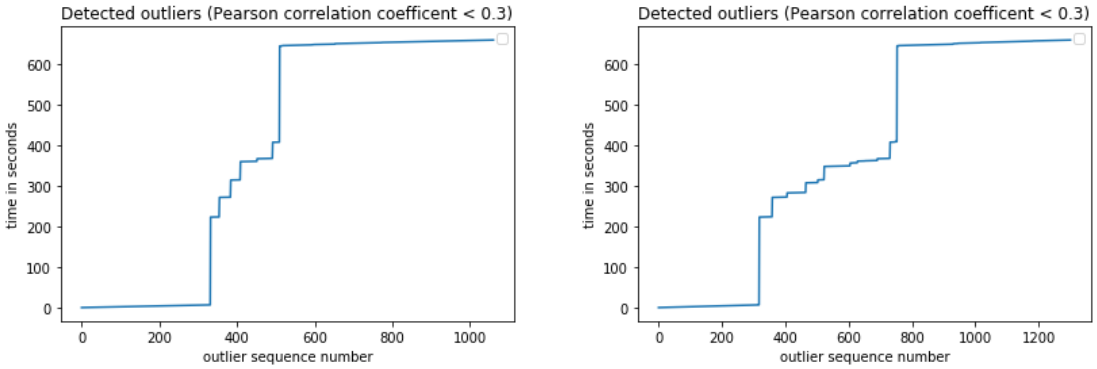

FIGURE 10. single and 2 iteration (outlier pre-filtering) outliers for participant 11.

the data recorded when performing the walking activity is higher and a higher threshold could be used to detect those flat segments in the climbing up stairs recording. Using a threshold of 0.94 the algorithm is able to detect the 24 regions with only 6 false positive values. Using a threshold value of 0.92, the same number of false positives is detected for the climbing down activity but only detecting 19 out of the 24 flat segments. In this particular case, the walking speed used to train the algorithm is more similar to the speed in which the user traverses the flat regions when climbing up stairs and the algorithm outperforms the recognition for the climbing down activity. Training the algorithm for the user walking at different paces could improve the recognition in such cases.

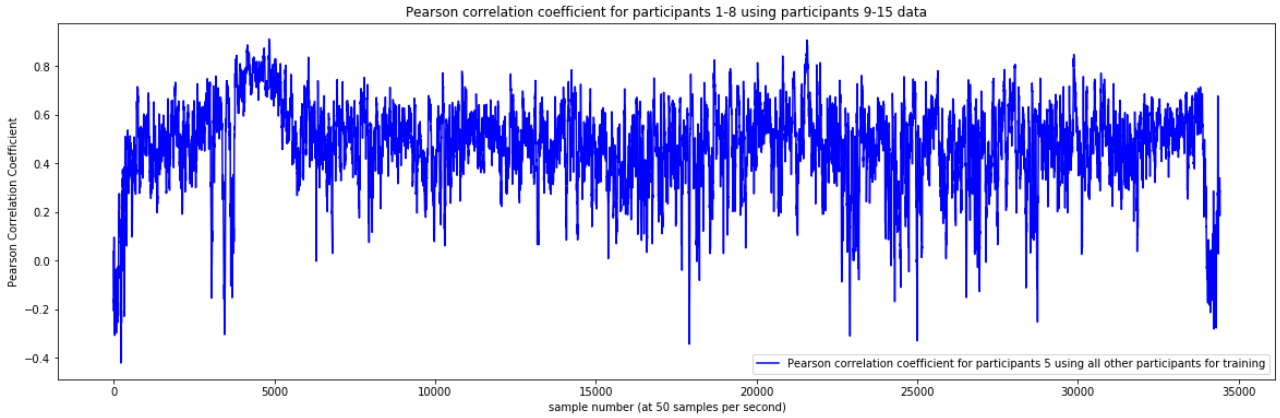

FIGURE 11. Similarities for participant 5 training with the rest of participants.

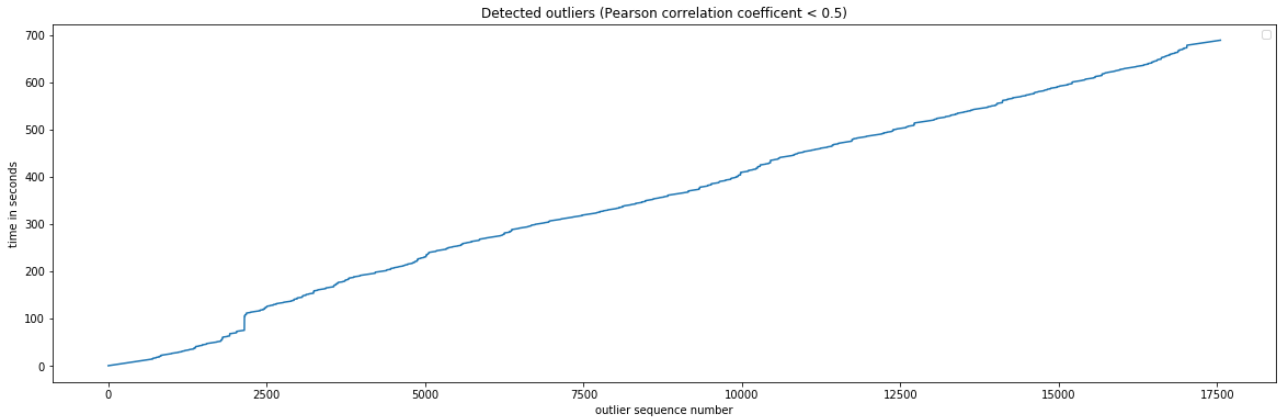

FIGURE 12. Outliers for participant 5 training with the rest of participants.

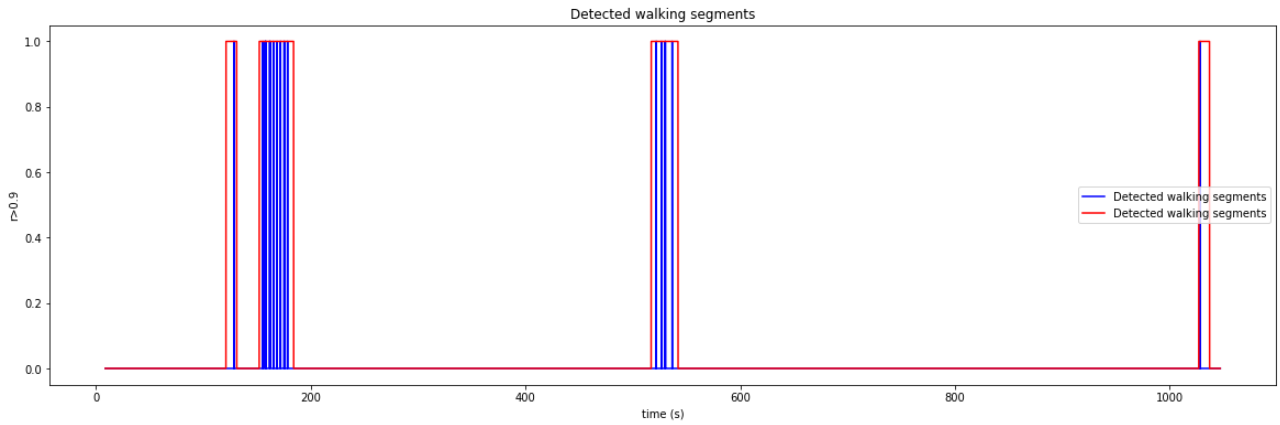

FIGURE 13. Detecting walking segments in a running recording.

The recall, precision and F_1 score are also shown in table 9. The best F_1 score value (achieved for a threshold similarity value of 0.94) is 0.89. The F_1 score value obtained by Sztyler and Stuckenschmidt [5] for the same sensor and activity is 0.8. The approach presented in this paper is able to improve a 9% the F_1 score and using a validation schema that does not use the data in the validation activity in the training phase.

D. INTER-SUBJECT DETECTION OF PARTICULAR SUB-ACTIVITIES

The final evaluation experiment has used the data from all the users except for the participant 4 to train the architecture in section IV. Once the algorithm is trained the same sub-activity detection test as in the previous section has been performed. The results are shown in Figure 14 and Tables 10 and 11.

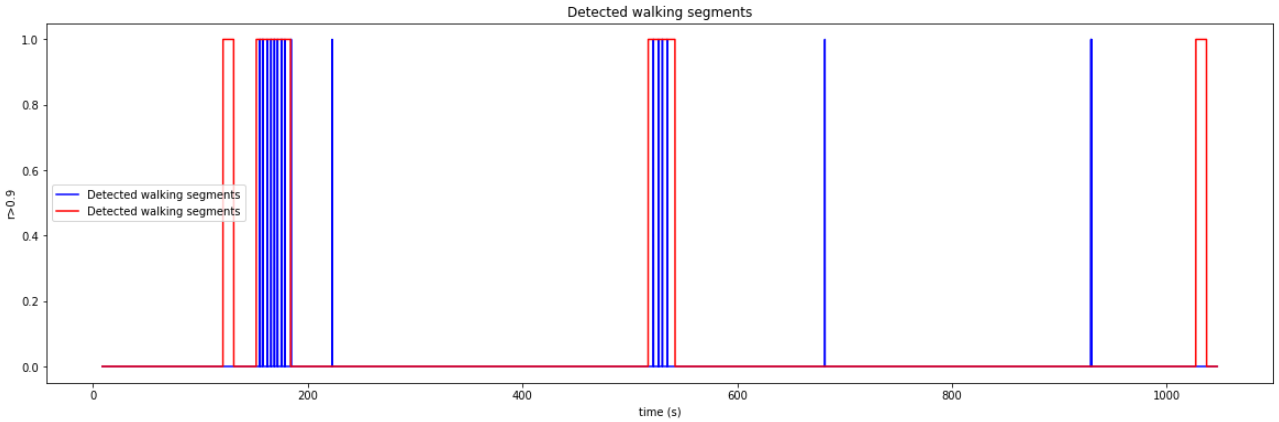

FIGURE 14. Detecting walking segments in a running recording.

TABLE 8. Number of flat section when climbing down stairs detected (24 flat sections in total).

rth	0.92	0.93	0.94	0.95	0.96
True positives	19	19	12	8	5
False positives	6	6	2	2	1
Recall	0.79	0.79	0.50	0.33	0.21
precision	0.76	0.76	0.86	0.80	0.83
F_1 *score*	0.78	0.78	0.63	0.47	0.33

TABLE 9. Number of flat section when climbing up stairs detected (24 flat sections in total).

rth	0.94	0.95	0.96	0.97	0.98
True positives	24	20	16	14	4
False positives	6	6	2	2	1
Recall	1.00	0.83	0.67	0.58	0.17
precision	0.80	0.77	0.89	0.88	0.80
F_1 *score*	0.89	0.80	0.76	0.70	0.28

The results in this case are worse than in the intra-subject training showing that important information about the way participant 4 walks is missing when using the data from the rest of participants.

Figure 14 captures the same results as Figure 13 for the case of inter-subject training. Only the 2 major walking segments are detected. Moreover, there are 3 false positives (running segments detected as walking). However, the length of these false positives is small which provides a simple mechanism to filter results.

The results in Table 10 show that a similar number of flat regions is detected when walking down stairs compared to the intra-subject case for the same threshold similarity values. However, the number of false positives increases. The values for the F_1 score decrease a 4% as compared to the intra-user case.

Table 11 shows a degradation both in terms of detected regions as well as in the false positive segments for the inter-subject case. The time series generated in the walking segments of the rest of the participants in the dataset show a higher similarity to the climbing up regions for user 4.

TABLE 10. Number of flat section when climbing down stairs detected (24 flat sections in total).

rth	0.92	0.93	0.94	0.95	0.96
True positives	19	15	14	8	4
False positives	8	7	7	4	2
Recall	0.79	0.63	0.58	0.33	0.17
precision	0.70	0.68	0.67	0.67	0.67
F_1 *score*	0.75	0.65	0.62	0.44	0.27

TABLE 11. Number of flat section when climbing up stairs detected (24 flat sections in total).

rth	0.94	0.95	0.96	0.97	0.98
True positives	19	16	8	5	2
False positives	14	12	5	3	2
Recall	0.79	0.67	0.33	0.21	0.08
precision	0.58	0.57	0.62	0.63	0.50
F_1 *score*	0.67	0.62	0.43	0.31	0.14

In fact, the results for the mean confusion matrix presented in [5] show that 9% of the walking up segments are classified as walking segments and 8.4% of the walking segments are classified as climbing up.

E. GENERALIZATION OF RESULTS

In order to assess the generalization of results, the information from the sensor in the waist region in the dataset described in [6] has been used for validation. The preprocessing algorithm described in section IV has been used in order to re-sample the acceleration data in the dataset in [6] to match the sampling frequency in [5]. The architecture in Figure 1 is trained with the data from all the participants in the dataset in [5] and validated with the data from participants in the dataset in [6]. In order to compare results with previous studies recognizing activities from the dataset in [6] the runs 4 and 5 from participants 2 and 3 have been used for validation.

The participants in [6] recorded the execution of a sequence of kitchen activities intercalating a large walking fragment

TABLE 12. Results for different threshold values.

rth	*RECALL*	*PRECISION*	F_1 *SCORE*	Accuracy
0.45	1.00	0.84	0.91	0.98
0.5	1.00	0.90	0.94	0.99
0.55	1.00	0.93	0.96	0.99
0.6	1.00	0.93	0.96	0.99
0.65	0.99	0.94	**0.97**	0.99
0.7	0.95	0.95	0.94	0.99
0.75	0.81	0.95	0.87	0.98

TABLE 13. Comparative results with previous studies.

Reference	F_1 score achieved over the dataset in [6]
[11]	0.93
[14]	0.92
[18]	0.94
This paper	0.97

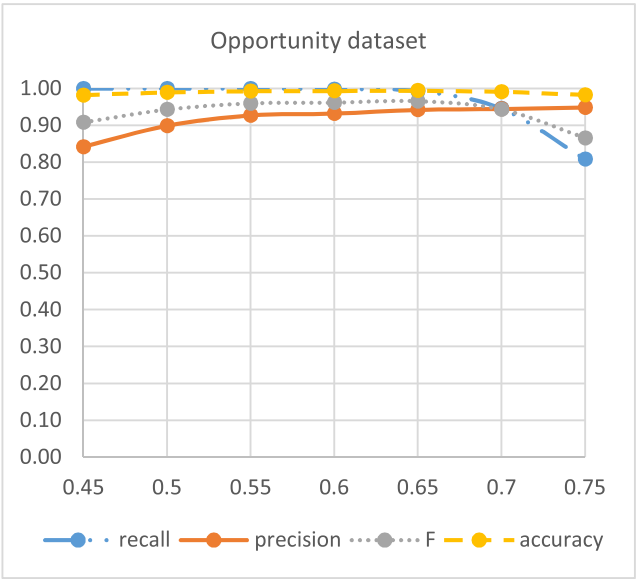

FIGURE 15. Detection of walking segments in the opportunity dataset.

around the building. While in the kitchen, the user performed short walking segments. In order to use the algorithm in Figure 1 (trained with the data from the walking activity from participants in [5]) to detect the walking around the building sub-activity for participants in [6], the Pearson correlation similarity measure has been averaged over 5 second periods. For short distance walking segments present in the execution of the different kitchen activities, the average similarity over a 5 second period will include data from non-walking activities and the average similarity is expected to be smaller compared to pure 5 second walking segments.

The results for the recall, precision, F_1 score and accuracy for different similarity thresholds are captured in Figure 15 and in table 12. The best F_1 score (for a similarity threshold of 0.65) is 0.97. Table 13 captures a comparative analysis with previous studies applying LSTM based alternative architectures for HAR over the dataset in [6]. The results in this paper outperform by 3% the best previous results in related studies.

VII. CONCLUSIONS

A novel architecture based on the use of Deep Recurrent Neural Networks (DRNN) in Human Activity Recognition (HAR) able to both detect the presence of anomalous segments of data inside the execution of a main activity and detect regions of a particular secondary activity inside a main activity has been presented and validated in three different scenarios:

- Intra-dataset, intra-subject validation: using the data for a single user for training and validation.
- Intra-dataset, inter-subject validation: using the data of all the users in a single dataset except one for training and the data for the left-aside user for validation.
- Inter-dataset, inter-subject validation: using the data for all the users in a dataset for training and the users in the second dataset for validation.

One of the major applications of outlier detection methods is to perform a data pre-processing cleansing task. In fact, the proposed algorithm for outlier detection shows better results if executed in 2 iterations: a first one to cleanse the dataset in order to remove secondary activity related data and a second one to optimally train the algorithm with single activity information and optimal outlier final detection.

In order to validate the results for the outlier detection approach, 3 major outlier types have been identified in the dataset in [5] and the algorithm has been configured with different thresholds to assess the results. For outlier types generating acceleration information significantly different from the one in the main activity (such as very slow walking, or being completely stopped), all the configurations both for the intra-subject and inter-subject cases are able to detect all of them. For other outlier types (such as turning around when walking), the optimal detection values have been obtained for the intra-user case (detecting all of the cases in the dataset in the case of turning around outliers). The inter-user case validation for the dataset in [5] showed that there was a participant that was badly characterized when using the information from the rest of the users and the outlier detection did not provide good results independently of the selected similarity threshold.

The sub-activity detection has been validated by detecting walking on a flat surface segments (training the algorithm with the walking activity data) inside the running/jogging and climbing up and down stairs activities. The intra-user results show that all walking segments were detected inside the running and in the climbing up activities (with no false positives inside the running activity and 6 inside the climbing up activity) and the majority (19 out of 24) in the climbing down activity (with a limited number of 6 false positives). The inter-user results have shown a certain degradation in the achieved performance, being able to detect 2 out of 4 walking sections inside the running activity (with 3 false positives) and

19 out of 24 for the climbing up activity (with 14 false positives). However, the degradation for the detection of walking on flat segments while climbing down stairs was small.

Finally, the inter-dataset scenario has shown that the architecture proposed in this paper is able to detect walking segments inserted in a run of different activities outperforming by around 3% previous results, even if the algorithm is trained with the data from a different dataset, using a different set of sensors in a different recording scenario.

As a future work, a wider set of activities will be analyzed. Moreover, the optimization of the architecture in terms of reducing the computational costs and hardware requirements will also be proposed.

REFERENCES

[1] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, 2013, pp. 1192–1209, 3rd Quart., 2013.
[2] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surv*, vol. 46, no. 3, Jan. 2014, Art. no. 33.
[3] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognit. Lett.*, vol. 119, pp. 3–11, Mar. 2019.
[4] H. F. Nweke, Y. W. Teh, M. A. Al-Garadi, and U. R. Alo, "Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges," *Expert Syst. Appl.*, vol. 105, pp. 233–261, Sep. 2018.
[5] T. Sztyler and H. Stuckenschmidt, "On-body localization of wearable devices: An investigation of position-aware activity recognition," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, Mar. 2016, pp. 1–9.
[6] (2012). *Opportunity Dataset*. Accessed: Apr. 9, 2019. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/OPPORTUNITY+Activity+Recognition
[7] (2008). *Skoda Dataset*. Accessed: Apr. 9, 2019. [Online]. Available: http://har-dataset.org/lib/exe/fetch.php?media=wiki:dataset:skodaminicp:skodaminicp_2015_08.zip
[8] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[9] M. Edel and E. Köppe, "Binarized-BLSTM-RNN based human activity recognition," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Oct. 2016, pp. 1–7.
[10] Y. Guan and T. Ploetz, "Ensembles of deep LSTM learners for activity recognition using wearables," 2017, *arXiv:1703.09370*. [Online]. Available: https://arxiv.org/abs/1703.09370
[11] N. Y. Hammerla, S. Halloran, and T. Ploetz, "Deep, convolutional, and recurrent models for human activity recognition using wearables," 2016, *arXiv:1604.08880*. [Online]. Available: https://arxiv.org/abs/1604.08880
[12] M. Inoue, S. Inoue, and T. Nishida, "Deep recurrent neural network for mobile human activity recognition with high throughput," 2016, *arXiv:1611.03607*. [Online]. Available: https://arxiv.org/abs/1611.03607
[13] A. Murad and J.-Y. Pyun, "Deep recurrent neural networks for human activity recognition," *Sensors*, vol. 17, no. 11, p. 2556. Nov. 2017.
[14] F. J. Ordóñez and D. Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, Jan. 2016.
[15] A. Sathyanarayana, S. Joty, L. Fernandez-Luque, F. Ofli, J. Srivastava, A. Elmagarmid, S. Taheri, and T. Arora, "Impact of physical activity on sleep: A deep learning based exploration," 2016, *arXiv:1607.07034*. [Online]. Available: https://arxiv.org/abs/1607.07034
[16] M. S. Singh, V. Pondenkandath, B. Zhou, P. Lukowicz, and M. Liwicki, "Transforming sensor data to the image domain for deep learning—An application to footstep detection," 2017, *arXiv:1701.01077*. [Online]. Available: https://arxiv.org/abs/1701.01077
[17] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher, "Deepsense: A unified deep learning framework for time-series mobile sensing data processing," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 351–360.
[18] Y. Zhao, R. Yang, G. Chevalier, X. Xu, and Z. Zhang, "Deep residual bidir-LSTM for human activity recognition using wearable sensors," *Math. Problems Eng.*, vol. 2018, Dec. 2018, Art. no. 7316954.
[19] O. S. Eyobu and D. Han, "Feature representation and data augmentation for human activity classification based on wearable IMU sensor data using a deep LSTM neural network," *Sensors*, vol. 18, no. 9, p. 2892, Aug. 2018.
[20] M. Inoue, S. Inoue, and T. Nishida, "Deep recurrent neural network for mobile human activity recognition with high throughput," *Artif. Life Robot.*, vol. 23, no. 2, pp. 173–185, Jun. 2018.
[21] D. M. Hawkins, *Identification of Outliers*, vol. 11. London, U.K.: Chapman-Hall, 1980.
[22] R. R. Blázquez, M. M. Organero, L. S. Fernández, "Evaluation of outliers detection algorithms for traffic congestion assessment in smart city traffic data from vehicle sensors," *Int. J. Heavy Vehicle Syst.*, vol. 25, nos. 3–4, pp. 308–321, Sep. 2018. doi: 10.1504/IJHVS.2018.10016106.
[23] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Procedia Comput. Sci.*, vol. 60, pp. 708–713, Jan. 2015.
[24] G. O. Campos, A. Zimek, J. SanderRicardo, J. G. B. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle, "On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study," *Data Mining Knowl. Discovery*, vol. 30, no. 4, pp. 891–927, Jul. 2016.
[25] D. Xu, E. Ricci, Y. Yan, J. Song, and N. Sebe, "Learning deep representations of appearance and motion for anomalous event detection," 2015, *arXiv:1510.01553*. [Online]. Available: https://arxiv.org/abs/1510.01553
[26] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, "Deep structured energy based models for anomaly detection," 2016, *arXiv:1605.07717*. [Online]. Available: https://arxiv.org/abs/1605.07717
[27] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 665–674.
[28] D. Mizell, "Using gravity to estimate accelerometer orientation," in *Proc. 7th IEEE Int. Symp. Wearable Comput.*, White Plains, NY, USA, Oct. 2003. pp. 21–23.
[29] S. D. Bolboaca and L. Jäntschi, "Pearson versus Spearman, Kendall's tau correlation analysis on structure-activity relationships of biologic active compounds," *Leonardo J. Sci.*, vol. 5, no. 9, pp. 179–200, Jul. 2006.
[30] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in *Proc. 9th Int. Conf. Artif. Neural Netw. (ICANN)*, 1999, pp. 850–855.
[31] M. Munoz-Organero, "Human activity recognition based on single sensor square HV acceleration images and convolutional neural networks," *IEEE Sensors J.*, vol. 19, no. 4, pp. 1487–1498, Feb. 2019.
[32] T. Sztyler, H. Stuckenschmidt, and W. Petrich, "Position-aware activity recognition with wearable devices," *Pervasive Mobile Comput.*, vol. 38, pp. 281–295, Jul. 2017.

MARIO MUNOZ-ORGANERO (M'08) received the M.Sc. degree in telecommunications engineering from the Polytechnic University of Catalonia, Barcelona, Spain, in 1996, and the Ph.D. degree in telecommunications engineering from the Universidad Carlos III de Madrid, Leganes, Spain, in 2004.

From 2015 to 2016, he was a Visiting Professor with the CATCH Centre, University of Sheffield. He is currently a Professor of telematics engineering with the Universidad Carlos III de Madrid, where he is currently the Head of the Telematics Engineering Department. He has published more than 160 research papers both on journals and international conferences. He has participated in several European-funded projects such as E-LANE and Spanish-funded projects, such as MOSAIC learning, Learn3, and OSAMI. He is currently the PI of the Spanish-funded FLATCity project and was PI for the HERMES, ARTEMISA, HAUS, COMINN, REMEDISS, and IRENE projects. He was also PI for the GEEWHEZ EU-FP7 project. His research interests include ambient intelligence, human activity recognition, machine learning techniques applied to human behavior, health, and independent living systems, ITS, open architectures for e-learning systems, open service creation environments for next-generation networks, advanced mobile communication systems, pervasive computing, and convergent networks. He was a MC member of TU1305 EU funded COST action about social networks and travel behavior.