UNIVERSITÀ DEGLI STUDI DI PALERMO

NDS NETWORKING AND DISTRIBUTED SYSTEMS RESEARCH GROUP

# *Smartphone Data Analysis for Human Activity Recognition.*

Article

Accepted version

F. Concone, S. Gaglio, G. Lo Re, M. Morana

Proceedings of AI*IA 2017 Advances in Artificial Intelligence:
XVI International Conference of the Italian Association for Artificial Intelligence, Bari, Italy, 2017.

# Smartphone Data Analysis for Human Activity Recognition

Federico Concone, Salvatore Gaglio, Giuseppe Lo Re and Marco Morana(✉)

DIID, University of Palermo – Italy
{federico.concone,salvatore.gaglio,giuseppe.lore,marco.morana}@unipa.it

**Abstract.** In recent years, the percentage of the population owning a smartphone has increased significantly. These devices provide the user with more and more functions, so that anyone is encouraged to carry one during the day, implicitly producing that can be analysed to infer knowledge of the user's context. In this work we present a novel framework for Human Activity Recognition (HAR) using smartphone data captured by means of embedded triaxial accelerometer and gyroscope sensors. Some statistics over the captured sensor data are computed to model each activity, then real-time classification is performed by means of an efficient supervised learning technique. The system we propose also adopts a participatory sensing paradigm where user's feedbacks on recognised activities are exploited to update the inner models of the system. Experimental results show the effectiveness of our solution as compared to other state-of-the-art techniques.



## 1 Introduction

Nowadays, smartphones have become an indispensable tool for everyday life, offering a number of features that go beyond the simple calling or messaging capabilities. In order to let the user perform different tasks, these mobile devices are equipped with many sensors that make them real sensing platforms able to extract relevant information. One of the most attractive scenario in which such information can be exploited is Human Activity Recognition (HAR), where data captured by motion sensors, e.g., accelerometer and gyroscope, can be analysed to infer user's current physical activity.

In this context, human activities can be intuitively considered as sequences of recurrent patterns in raw data captured from smartphone's sensors. Many HAR algorithms have been presented in the literature, however their application is frequently restricted to specific application scenarios, e.g., e-health, or their inner behaviour is unknown. For example, one of the most reliable HAR technique is that implemented in the Google APIs for Android, which unfortunately acts as a black-box not providing neither a way to understand what mechanisms are behind it, nor intermediate information to supervise the running of the recognition process.

In this paper we present a framework for real-time human activity recognition using smartphones. In particular, we address a participatory sensing scenario

where users contribute to the proper functioning of the system by providing i) their own data through a secure client-server architecture, and ii) feedbacks on the correctness of the recognised activities.

Such information is exploited to maintain the inner models of the systems updated, so as to recognize even more instances of known activities performed by different users. The core of the HAR module consists in a state of the art machine learning technique, i.e., k-nearest neighbors (K-NN), which showed the best performances both in terms of algorithmic efficiency and recognition accuracy.

The remainder of the paper is organized as follows: related work is outlined in Section 2. The system architecture is described in Section 3, and the experimental results are shown and discussed in Section 4. Conclusions follow in Section 5.

## 2 Related Work

Most of the methods presented in the literature in the area of human activity recognition can be grouped in two main categories: vision and sensor-based.

The former can infer user's activity by analyzing video sequences [3] captured by a number of different devices. Early techniques were based on the extraction of the users' silhouettes from RGB images, however, the cost of image processing algorithms needed for cleaning noisy input data make these approaches usually unsuitable for real-time applications. More recently, the focus moved to unobtrusive devices capable of capturing both RGB and depth information, e.g., the Microsoft Kinect [21,24]. RGB-D data were proven to improve the recognition process allowing for real-time analysis of the observed scene [14]. Some applications have been proposed in the context of ambient intelligent systems aimed to recognize the user activities [11], e.g., for energy saving [10] purposes, or to improve the user experience [20] providing unobtrusive interfaces. Unfortunately, activity recognition through these sensors is quite limited to indoor environments.

Sensor-based HAR techniques analyze information from various sensors located in different parts of the human body. This approach overcomes the limitations of vision-based methods, but *wearable* sensors are reluctantly accepted by the users due to their intrusiveness [25]. Early experiments were performed using only one accelerometer to capture acceleration values on the three axes [28]. However, since a single sensor is not suitable to describe very complex activities, several studies have been presented merging information provided by multiple sensors [13,23]. For example, in [5] the system acquires data from five biaxial accelerometers, worn simultaneously on different parts of the body, to recognize both simple and complex activities. In [6] the use of wearable devices in a e-health scenario is presented. The authors of [4] describe a method to recognize *walking*, *sitting*, *standing*, and *running* activities by means of five accelerometers. Other studies tried to improve the performances of their recognition systems by relying on the combinations of heterogeneous sensors, e.g., accelerometers and gyroscopes, microphones, GPS, and so on [30,19].

Over the years, more and more works have focused on HAR using smartphone-based applications. This choice is mainly due to the widespread diffusion of smartphones which exhibited suitable characteristics, e.g., embedded sensors, easy portability of the device, network connectivity, and higher computing power.

In [12] the authors present an approach that combines machine learning and symbolic reasoning for improving the quality of life of diabetic patients. The whole system is based on the recognition of some activities using smartphone sensors in order to trace patients' fatigue and depression while performing the daily routines. The system described in [31] covers a mobile health scenario where neural networks, implemented on Android smartphones, are used to recognize five activities: *static*, *walking*, *running*, *upstairs*, and *downstairs*. The same scenario is addressed in [16], where a fall detection system using accelerometers and magnetometers is proposed. In particular, if a certain threshold value is exceeded, the method is able to recognize falls in four different directions: backward, forward, left and right. In [18], the authors describe an unsupervised learning approach to human activity recognition using smartphone sensors, even when the number of activities is unknown. However, the recognition process is strictly dependent on the number of clusters chosen during the design phase. Thus, distinct activities could be erroneously merged into one, or different instances of the same activity could be seen as unrelated. KAFKA [26] is a system analyzing real-time data collected by inertial sensors mounted on Android devices, i.e. smartphones and smartwatches.

An open framework designed to ease the development of Mobile CrowdSensing (MCS) systems is presented in [9,8]. Mobile Sensing Technology (MoST), available for Android-based devices, provides some activity recognition and geofencing algorithms optimised to meet computational and power constraints of smartphones. Activity recognition is performed by processing raw data captured by on-board sensors, and allows to distinguish between three kinds of activities: *walking*, *running*, and *phone still*. Geofencing aims to find and delimit the geographic area where a certain activity, or event, occurs. This is usually achieved by correlating motion data captured by heterogeneous sensors with geographic coordinates provided by GPS (Global Positioning System) or IPS (Indoor Positioning Systems). The latter can include a number of different algorithms and sensors, for this reason MoST architecture is modular, allowing developers to extend its functionalities by implementing new techniques or considering custom and virtual sensors.

Despite of the efforts made to design efficient frameworks, one of the best performing HAR solution is still that proposed by Google [1] since its API level-1. However, since such a tool is closed source, developers are not able to use intermediate results as part of their systems, nor to provide any feedback to the activity recognition routine.

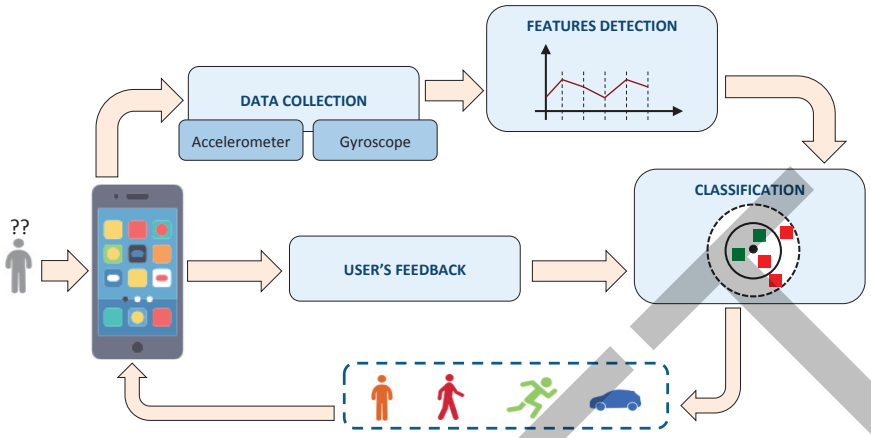



Fig. 1: System overview. The activities performed by the users are captured by means of the smartphone sensors. Collected data are analyzed to detect some relevant features, that are then used for classification. User's feedbacks on recognised activities are exploited to improve the classification process.

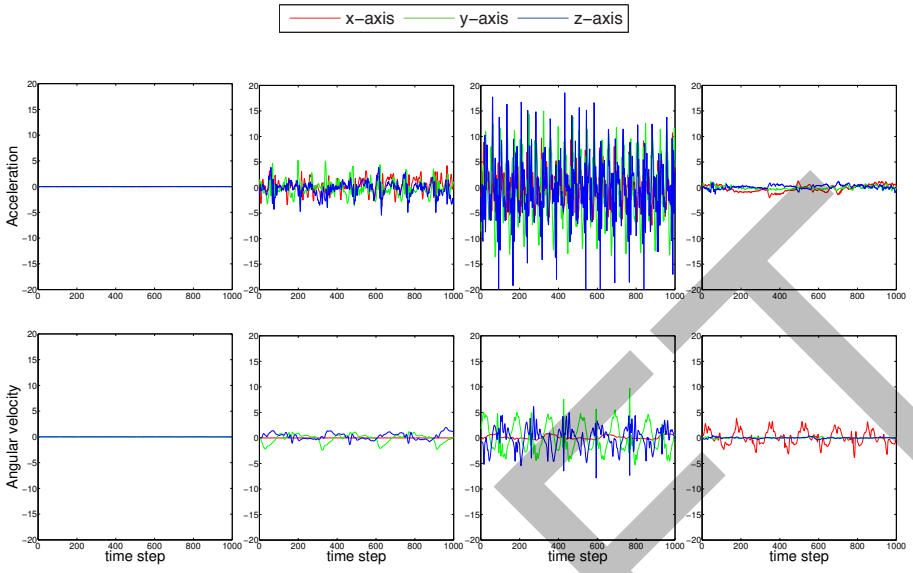
## 3 Activity Recognition System

The architecture we propose here aims at automatically inferring the activity performed by the users, in a generic scenario, according to data collected through their smartphones. The system components can be logically divided in three parts (see Fig.1). The first is responsible for *data collection*, that is for capturing raw data through the smartphone sensors while an activity is performed. The raw values are sent as input to the *features detection* module, where a set of n-dimensional points are extracted to distinguish different activities. Activities are classified using a machine learning algorithm, then user can provide feedbacks on the output of the recognition so as to allow the system to properly perform future classifications.

Data collection is performed by using the MoST open-source library, simply requiring that the user performs different activities while having its smartphone in the pants pocket. Differently from [18], our system does not take into account the gravity acceleration, allowing the user for holding the smartphone without worrying about its orientation.

In order to understand how the activity patterns change according to accelerometer and gyroscope readings, Fig. 2 shows values of three-axes acceleration (top row) and angular velocity (bottom row) captured while performing *still*, *walking*, *running*, and *vehicle* activities respectively.

As regards the acceleration values, even though these four activities look somehow different from each other, some of them, i.e., *still* and *vehicle* share a similar pattern, whilst others, i.e., *walking* and *running* are characterized by high noise as they are intrinsically associated with a significant user movement.




(a) (b) (c) (d)

Fig. 2: Three-axes acceleration (top row) and angular velocity (bottom row) for *still* (a), *walking* (b), *running* (c), and *vehicle* (d) activities .

On the contrary, by analyzing the values of angular velocity we noticed that *still* and *vehicle* exhibit distinct patterns, whilst other activities are generally characterized by oscillations of different width and frequency. For this reason we combined data from the two sensors, so as to get the best from both sides.

To ensure real-time activity recognition, the system must be able to process input data within certain time windows in order to extract the features that will be used in the next classification stage. The entire process of feature extraction is shown in Fig.3.

We define an activity $a$ as the user behaviour observed from initial time $t_i$ to final time $t_f$. Given that the duration, in seconds, of the activity $a_j$ is denoted by $d_j$, data captured within this interval is processed into fixed-length windows of $m \times n$ samples, where $m$ is the number of axes along which measurements are performed. In particular, the activity recognition process is based on $(X_A, Y_A, Z_A)$ values provided by the accelerometer, and $(X_G, Y_G, Z_G)$ values from the gyroscope.

Choosing the proper length for the acquisition window is essential because of the impact it could have on the whole system. Short windows may improve system performance in terms of execution time and CPU load, but may not contain enough information to properly capture the characteristics of the activity. Vice versa, long windows may alter the system performances since information about multiple activities performed in sequence might be analyzed within a sin-

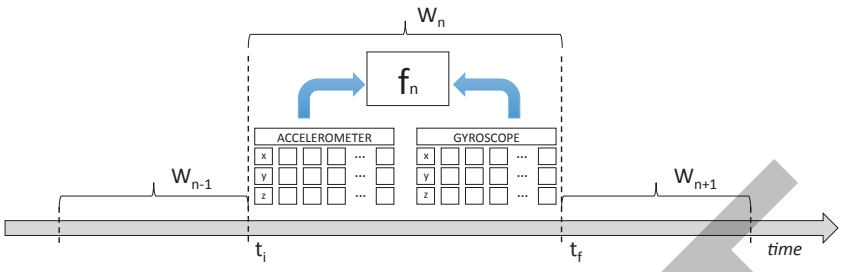



Fig. 3: Feature extraction mechanism. Accelerometer and gyroscope data are processed within the n-th fixed-length time window $W_n$, in order to obtain the corresponding feature vector $f_n$.

gle window. Experimental results using windows of different length, suggested us to use fixed-width windows of 3 seconds with no overlap.

In order to obtain a compact representation of input data, feature vectors are built similar to [8] by considering (i) *max value*, (ii) *min value*, (iii) *mean*, (iv) *standard deviation*, and (v) *root mean square* over the three accelerometer and gyroscope axes. Therefore, each feature vector $f$ contains 30 elements, i.e., 15 values of acceleration and 15 values of angular velocity.

The classification process is based on the k-nearest neighbors (K-NN) technique. Given the set of feature vectors $(f_1, f_2, \cdots, f_m)$, the key principle behind K-NN is that an unknown feature point $f$, projected into a *large* training set of labeled data, would be ideally surrounded by samples of its same class. If this happens, the algorithm could assign to $f$ the same class of its closest neighbor, i.e., the closest point in the feature space. More generally, the set $S$ containing the $k$ closest neighbors of $f$ is selected and the most recurrent class in $S$ is assigned to $f$.

As mentioned earlier, our system adopts a participatory sensing paradigm [7], which aims at exploiting information provided by a community of users to improve the system performances. To this end, a client/server architecture has been designed, allowing each user to i) share its own data captured by the smartphone, and ii) use the same device to leave feedbacks about the recognition process, indicating, every time an activity has been recognized, whether the output class is correct or not. Data sent by the client are analyzed within the server to determine if the models of the different activities need to be updated. In particular, the feature vector $f_{new}$ received from the client is projected into the current feature space, together with the class declared by the user. Then, $f_{new}$ is compared with existing data from the same class, and if they are similar above a certain threshold, the activity models are re-computed and sent back to the client for future classifications.

| Model | CPU | RAM |
|---|---|---|
| Galaxy S7 Edge | 8 Core 2 GHZ | 4 GB |
| Galaxy S5 Neo | 8 Core 1.6 GHZ | 2 GB |
| Galaxy S4 | 4 Core 1.9 GHZ | 2 GB |
| Galaxy S2 NFC | 2 Core 1.2 GHZ | 1 GB |
| Galaxy Note | 2 Core 1.4 GHZ | 2 GB |

Table 1: Smartphone models used for the experiments.

## 4 Experiments

In order to evaluate the effectiveness of our framework, several experiments were performed. First, we present a comparison between the results obtained while using the K-NN method, and a classifier based on K-means clustering. Then, the overall performances of the HAR system we propose are compared with two state of the art techniques, i.e., the activity recognition tools provided by MosT and Google.

The experiments were carried out using five different models of smartphones, as summarized in Table 1, equipped with built-in accelerometer and gyroscope sensors. Our application can be installed on any Android device with Ice Cream Sandwich OS or higher.

The choose of the specific classification algorithm to adopt is mainly dependent on two aspects: its accuracy, and its efficiency in terms of time complexity and memory consumption. For this reason, we firstly compared K-NN with a widely used clustering algorithm that can be adapted to classification.

### 4.1 Choosing the classification algorithm

In the considered scenario, given the set of feature vectors $(f_1, f_2, \cdots, f_m)$, the most common application of the K-means algorithm consists in partitioning the $m$ observations into $k$ sets, $\mathbf{C} = (C_1, C_2, \cdots, C_k)$, so as to minimize the intra-cluster error:

$$E = \sum_{k=1}^{K} \sum_{f_i \in C_k} \|f_i - \mu_k\|^2 , \tag{1}$$

where $\mu_k$ is the mean value of the k-th cluster $C_k$.

Nevertheless, K-means can also be used for classification, i.e., supervised learning, according to two different schemes [15]. The first, straightforward, solution is to apply K-means to the whole blended training set and observe how data from $k$ different classes are associated with each of the $k$ centroids $C_k$. Then, each centroid is marked as representative of a certain class $i$, with $i = 1, ..., k$, if most of the samples in the cluster associated with $C_k$ belong to $i$. Classification of a new, unknown, feature vector $f$ is performed by finding its closest centroid and then assigning to $f$ the same class of $C_k$. The major drawback to this method is

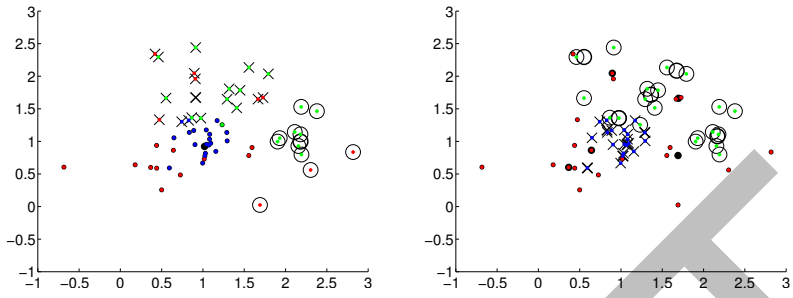



Fig. 4: Examples of k-means classification adopting two different schemes. Data from three classes (red, green, and blue) are classified into three clusters (crosses, circles, and dots) using blended (a) or separated (b) training sets.

that performing K-means on blended data produces heterogenous clusters. i.e., there is no guarantee that all the points in the same cluster represent the same class.

The second approach helps to overcome this limitation by separating the training data in $n$ distinct groups, each containing samples from one of the $n$ classes we want to recognize. The K-means algorithm is applied on each group/class separately, so as to obtain $k$ homogeneous clusters, i.e., all the centroids within a single group represents the same class. Thus, classification can be performed by comparing a new, unknown, feature vector $f$ with the $k \times n$ labeled centroids, and assigning to $f$ the class of the closest one.

The differences between the two methods are summarized in Fig. 4. Original samples from three classes are represented by red, green, and blue points. As a result of the classification, points are marked as belonging to one of three clusters denoted by crosses, circles, and dots. When using the first scheme, K-means is applied to the whole blended dataset (Fig. 4-a) so that each cluster contains data from different classes, e.g., the cluster denoted by crosses includes red, green, and blue points. On the contrary, the second scheme (Fig. 4-b) is preferable since it allows to apply K-means on each class separately, so obtaining homogeneous clusters, e.g., the elements of the cluster denoted by circles are all greens.

As regards the choice of $K$, some experiments were conducted to determine the best value for K-means and K-NN.

For K-means, the value of $K$ is simply the number of activities to be recognized, i.e., $K = 4$. In order to find the best value of $K$ for the K-NN algorithm, two techniques for predictive accuracy evaluation have been used, i.e., Resubstitution and Cross Validation. Since we addressed a scenario where limited resource devices are employed, only odd values of $K$ in the range $[3, 9]$ were considered. Experiments showed that best results are achieved with $K = 7$.

Then, tests were performed to compare K-NN with K-means (scheme 2) in terms of *accuracy*, *precision*, and *recall* [29]. Fig. 5 shows that slightly better

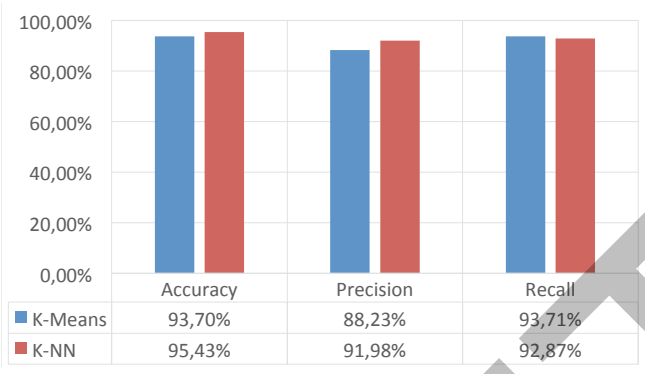



Fig. 5: Accuracy, precision, and recall of K-means and K-NN.

| | *still* | *walking* | *running* | *vehicle* |
|---|---|---|---|---|
| *still* | .87 | .03 | 0 | .1 |
| *walking* | 0 | 1 | 0 | 0 |
| *running* | 0 | 0 | 1 | 0 |
| *vehicle* | .12 | .03 | .06 | .79 |

Table 2: K-means confusion matrix.

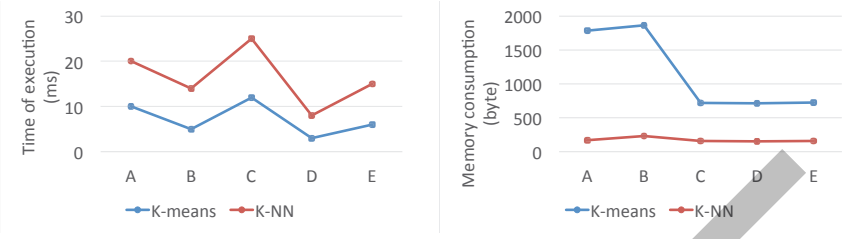
| | *still* | *walking* | *running* | *vehicle* |
|---|---|---|---|---|
| *still* | .97 | 0 | .01 | .02 |
| *walking* | 0 | 1 | 0 | 0 |
| *running* | 0 | 0 | 1 | 0 |
| *vehicle* | 0 | .01 | .02 | .97 |

Table 3: K-NN confusion matrix.

results are obtained while applying the K-NN algorithm. This is mainly due to the incapacity of K-means to distinguish between some similar activities. In particular, as highlighted by the confusion matrices shown in Table 2 and Table 3, *still* and *vehicle* activities are frequently mistaken because of their similar acceleration values (see Fig. 2). This error is almost negligible when adopting the K-NN classifier.

The next set of experiments were aimed at comparing K-means and K-NN in terms of time of execution and memory consumption. We performed some tests to measure these two parameters while varying the duration of the processing windows. Results are shown in Fig. 6(a) and Fig. 6(b). For the first two cases, i.e., test A and test B, the duration of the window is about 2 minutes, whereas smaller windows of about 50 seconds where used for C, D, and E. K-means is generally faster than K-NN, whilst K-NN, independently of the length of time windows, requires almost constant memory consumption.

Thus, since we want the HAR application to be as lightweight as possible so as to prevent the system resources from being consumed more than necessary, we decided to build the classification module on the K-NN algorithm. Such analysis is also confirmed by the results reported in [22].




(a) (b)

Fig. 6: Time of execution (a) and memory consumption (b) of K-means and K-NN classifiers measured under five different conditions denoted by {A,B,C,D,E}.

### 4.2 Comparison with MosT and Google

The last set of experiments was designed for comparing the system described in Fig. 1 with two state of the art HAR techniques, i.e., those implemented by the MosT framework and by the Google APIs.

As discussed in section 1, Google does not provide any detail of the algorithms behind their products, thus we treated their recognition algorithm as a black-box. On the contrary, MosT is based on a well known algorithm to efficiently build decision trees, namely C4.5 [27].

Since MosT and Google are able to recognize different types of activities, in order to perform a meaningful assessment two distinct subsets have been defined. More precisely, the class *other* was added to cover the activities not considered in both of the systems alternately compared. Thus, since Google technique is unmodifiable, and it recognizes a greater number of activities than our system, the comparison was based on a subset formed by *still*, *walking*, *running*, *vehicle*, and *other*. On the contrary, even if MosT originally included only *still*, *walking*, and *running*, we have been able to add the *vehicle* activity obtaining the same set addressed by our system.

Accuracy, precision, and recall achieved by the system we propose as compared to MosT are showed in Fig. 7. MosT results are detailed in the confusion matrix reported in Table 4. As expected, *walking* and *running* are almost correctly classified, whilst the recognition of *vehicle* and *still* is more difficult to perform. This can be explained because MosT considers only accelerometer data, that, as shown in Fig. 2, are not useful enough for discriminating between a still user and one driving at constant velocity. Moreover, decision trees are generally less predictive than other classification approaches, since a small change in the data can cause a large change in the final estimated tree [17].

As regards the results obtained comparing the proposed system with the Google activity recognition tool (see Fig. 7), we can notice that Google performances are quite lower than ours, and this can be explained by referring to the Table 5. In fact, the implementation provided by Google is not able to correctly distinguish between *walking* and *running* activities. In addition, as already dis-

| | *still* | *walking* | *running* | *vehicle* |
|---|---|---|---|---|
| *still* | .33 | 0 | 0 | .67 |
| *walking* | 0 | .96 | .04 | 0 |
| *running* | 0 | .02 | .98 | 0 |
| *vehicle* | .43 | 0 | 0 | .57 |

Table 4: MosT confusion matrix.

| | *still* | *walking* | *running* | *vehicle* | *other* |
|---|---|---|---|---|---|
| *still* | .92 | 0 | 0 | 0 | .08 |
| *walking* | 0 | .56 | .44 | 0 | 0 |
| *running* | 0 | .45 | .55 | 0 | 0 |
| *vehicle* | 0 | 0 | 0 | .91 | .09 |

Table 5: Google confusion matrix.

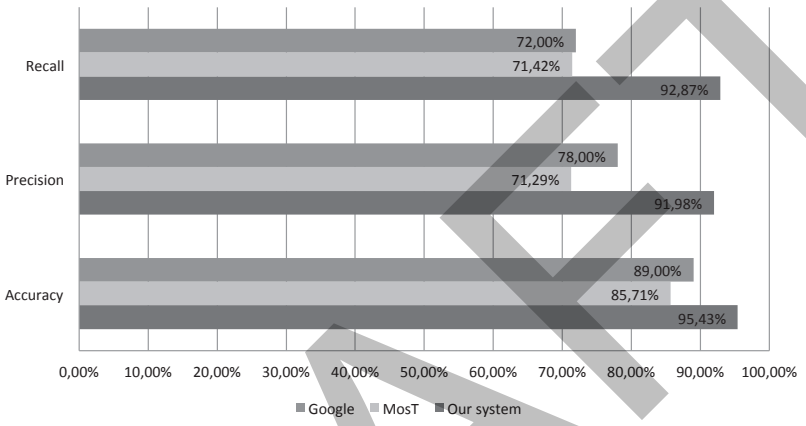



Fig. 7: Comparison between the proposed system, MosT, and Google.

cussed, it is not possible to run further experiments, similar to those described for MosT, to analyze how changing the set of activities would impact on the system performance. This represents a further advantages of our system compared with Google method.

## 5 Conclusions

In this paper we presented a novel human activity recognition framework based on smartphone built-in accelerometer and gyroscope sensors. The characteristics of four activities, i.e., *still*, *walking*, *running*, and *vehicle*, are represented through some feature vectors obtained by processing input data within fixed-length time windows. A K-NN classifier is then applied to recognize the activity performed by the user. Moreover, our architecture includes a participatory sensory module which allows to exploit user's feedbacks on recognised activities to keep updated the inner models of the system as more examples become available.

A set of experiments were run to compare the proposed solution with two state of the art techniques, i.e., the MosT framework, and the activity recognition tool provided by Google. Experimental results showed the effectiveness of our implementation both in terms of accuracy and efficiency.

As future works, we want to introduce a dynamic mechanism to filter out noisy data captured just before or after an activity is performed, e.g., when the user interacts with its smartphone to start or stop the Android application. Moreover, the adoption of variable length detection windows will be investigated.

We also plan to extend the set of activities by including some frequently performed in everyday life, e.g., *stairs down*, *stairs up*, *on bicycle*, and so on. This would be very useful in order to design a system that is able to recognize complex activities composed of simple tasks, such as those described in this paper. For example, the complex activity *from home to work*, could be recognized as a concatenation of *stairs down*, *walking*, *vehicle*, *walking*, *stairs up*, and *still*. In order to improve the user feedback mechanism, we want to investigate the adoption of reputation management techniques, as discussed in [2].

## References


1. Activity recognition api. `https://developers.google.com/android/reference/com/google/android/gms/location/ActivityRecognitionApi/` (Nov 2016)
2. Agate, V., de Paola, A., Lo Re, G., Morana, M.: A simulation framework for evaluating distributed reputation management systems. Advances in Intelligent Systems and Computing 474, 247–254 (2016)
3. Aggarwal, J.K., Xia, L.: Human activity recognition from 3d data: A review. Pattern Recognition Letters 48, 70–80 (2014)
4. Banos, O., Damas, M., Pomares, H., Prieto, A., Rojas, I.: Daily living activity recognition based on statistical feature quality group selection. Expert Systems with Applications 39(9), 8013–8021 (2012)
5. Bao, L., Intille, S.S.: Activity recognition from user-annotated acceleration data. In: Int. Conf. on Pervasive Computing. pp. 1–17. Springer (2004)
6. Baretta, D., Sartori, F., Greco, A., Melen, R., Stella, F., Bollini, L., D'addario, M., Steca, P.: Wearable devices and ai techniques integration to promote physical activity. In: Proc. of the 18th Int. Conf. on Human-Computer Interaction with Mobile Devices and Services Adjunct. pp. 1105–1108. ACM (2016)
7. Burke, J.A., Estrin, D., Hansen, M., Parker, A., Ramanathan, N., Reddy, S., Srivastava, M.B.: Participatory sensing. Center for Embedded Network Sensing (2006)
8. Cardone, G., Cirri, A., Corradi, A., Foschini, L., Maio, D.: Msf: An efficient mobile phone sensing framework. Int. Journal of Distributed Sensor Networks 9(3), 538937 (2013), `http://dx.doi.org/10.1155/2013/538937`
9. Cardone, G., Corradi, A., Foschini, L., Ianniello, R.: Participact: A large-scale crowdsensing platform. IEEE Transactions on Emerging Topics in Computing 4(1), 21–32 (2016)
10. Cottone, P., Gaglio, S., Lo Re, G., Ortolani, M.: User activity recognition for energy saving in smart homes. Pervasive and Mobile Computing 16(PA), 156–170 (2015)
11. Cottone, P., Lo Re, G., Maida, G., Morana, M.: Motion sensors for activity recognition in an ambient-intelligence scenario. In: 2013 IEEE Int. Conf. on Pervasive Computing and Communications Workshops (PERCOM Workshops). pp. 646–651 (2013)
12. Cvetković, B., Janko, V., Romero, A.E., Kafalı, Ö., Stathis, K., Luštrek, M.: Activity recognition for diabetic patients using a smartphone. Journal of medical systems 40(12), 256 (2016)

13. De Paola, A., La Cascia, M., Lo Re, G., Morana, M., Ortolani, M.: Mimicking biological mechanisms for sensory information fusion. Biologically Inspired Cognitive Architectures 3, 27–38 (2013)
14. Gaglio, S., Lo Re, G., Morana, M.: Human activity recognition process using 3-d posture data. IEEE Transactions on Human-Machine Systems 45(5), 586–597 (Oct 2015)
15. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning: data mining, inference and prediction. Springer, 2 edn. (2009)
16. Hwang, S., Ryu, M., Yang, Y., Lee, N.: Fall detection with three-axis accelerometer and magnetometer in a smartphone. In: Proc. of the Int. Conf. on Computer Science and Technology, Jeju, Korea. pp. 25–27 (2012)
17. James, G., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning: With Applications in R. Springer Publishing Company, Incorporated (2014)
18. Kwon, Y., Kang, K., Bae, C.: Unsupervised learning for human activity recognition using smartphone sensors. Expert Systems with Applications 41(14), 6067–6074 (2014)
19. Lester, J., Choudhury, T., Borriello, G.: A practical approach to recognizing physical activities. In: Int. Conf. on Pervasive Computing. pp. 1–16. Springer (2006)
20. Lo Re, G., Morana, M., Ortolani, M.: Improving user experience via motion sensors in an ambient intelligence scenario. In: Proc. of the 3rd Int. Conf. on Pervasive Embedded Computing and Communication Systems - Volume 1: PECCS,. pp. 29–34. INSTICC, SciTePress (2013)
21. Manzi, A., Dario, P., Cavallo, F.: A human activity recognition system based on dynamic clustering of skeleton data. Sensors 17(5), 1100 (2017)
22. Munther, A., Razif, R., AbuAlhaj, M., Anbar, M., Nizam, S.: A preliminary performance evaluation of k-means, knn and em unsupervised machine learning methods for network flow classification. Journal of Electrical and Computer Engineering 6(2), 778–784 (2016)
23. Paola, A.D., La Cascia, M., Lo Re, G., Morana, M., Ortolani, M.: User detection through multi-sensor fusion in an ami scenario. In: 2012 15th Int. Conf. on Information Fusion. pp. 2502–2509 (2012)
24. Park, S., Park, J., Al-masni, M., Al-antari, M., Uddin, M.Z., Kim, T.S.: A depth camera-based human activity recognition via deep learning recurrent neural network for health and social care services. Procedia Computer Science 100, 78–84 (2016)
25. Patel, S., Park, H., Bonato, P., Chan, L., Rodgers, M.: A review of wearable sensors and systems with application in rehabilitation. Journal of neuroengineering and rehabilitation 9(1), 21 (2012)
26. Pinardi, S., Sartori, F., Melen, R.: Integrating knowledge artifacts and inertial measurement unit sensors for decision support. In: KMIS. pp. 307–313 (2016)
27. Quinlan, J.R.: C4. 5: programs for machine learning. Elsevier (2014)
28. Ravi, N., Dandekar, N., Mysore, P., Littman, M.L.: Activity recognition from accelerometer data. In: Aaai. vol. 5, pp. 1541–1546 (2005)
29. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Information Processing & Management 45(4), 427–437 (2009)
30. Subramanya, A., Raj, A., Bilmes, J.A., Fox, D.: Recognizing activities and spatial context using wearable sensors. arXiv preprint arXiv:1206.6869 (2012)
31. Torres-Huitzil, C., Alvarez-Landero, A.: Accelerometer-based human activity recognition in smartphones for healthcare services. In: Mobile Health, pp. 147–169. Springer (2015)