Personal and Ubiquitous Computing manuscript No.
(will be inserted by the editor)

Stairstep Recognition and Counting in a Serious Game for Increasing Users' Physical Activity

Matteo Ciman · Michele Donini · Ombretta Gaggi · Fabio Aiolli

Received: date / Accepted: date

Abstract The high diffusion of smartphones in the users' pockets allows to sense their movements, thus monitoring the amount of physical activity they do during the day. But, it also gives the possibility to use these devices to persuade people to change their behaviors. In this paper, we present *ClimbTheWorld*, a *serious game* which uses a machine learning based technique to recognize and count stairsteps and aims at persuading people to use stairs instead of elevators or escalators. We perform a fine-grained analysis by exploiting smartphone sensors to recognize single stairsteps. Energy consumption is widely investigated to avoid exhausting smartphone battery. Moreover, we present game appreciation and persuasive power results after a trial experiment with 13 participants.

Keywords activity recognition · energy consumption · pervasive and mobile computing · ubiquitous applications

1 Introduction

According to the World Health Organization, at least 3.2 million people die every year due to heart diseases, diabetes, cancer and obesity [33]. One of the main causes of these diseases can be attributed to insufficient physical activity (both in adults and children): in the last few decades the technological progress has completely changed people's lifestyle, making everyday activities much easier. This progress also moved people to a sedentary life, thus increasing the occurrence of these diseases and the medical costs for their treatment.

Unfortunately, the availability of many infrastructures that reduce movements necessary to reach a destination is not balanced by the presence of tools which help people to modify unhealthy behaviors, and to avoid negative consequences of physical inactivity. Since everyday choices between taking a bus and a short distance walk, or between stairs and elevator, could greatly affect the total amount of activity made, we propose as a solution the use of a *serious game*, *ClimbTheWorld*, to incentivize people to use stairs instead of elevators or escalators.

ClimbTheWorld is a smartphone application. The idea underlying the game is simple: the user has to climb real world buildings, e.g., the Empire State Building or the Eiffel Tower, climbing stairs during his/her everyday life. Once started, the game records and analyzes data from the accelerometer and counts the number of stairsteps climbed by the user. The use of mobile technologies in real-time persuasive solutions is a very good combination since smartphones are commonly available in the users' pocket and can invisibly work and help people to change their unhealthy behavior.

To be efficient, the game must fulfill two requirements: it has to count stairsteps without immediately exhausting the smartphone battery, and to persuade users to change their behavior. *ClimbTheWorld* implements a new method for stairsteps recognition and counting. The game requires a fine-grained classification, that is a real-time counting of the number of stairsteps along with a real-time feedback. Moreover, it does not impose any constraints on the smartphone position, and it does not require people to buy expensive tools (e.g.,

M. Ciman, M. Donini, O. Gaggi and F. Aiolli
Department of Mathematics
Via Trieste, 63, University of Padua, Italy
E-mail: {mciman, mdonini, gaggi, aiolli}@math.unipd.it

M. Donini
Computational Statistics and Machine Learning
Istituto Italiano di Tecnologia, Genoa, Italy

the Nike+ FuelBand bracelet[1]), but it uses only the smartphone sensors.

To avoid exhausting the smartphone battery, the whole design and implementation process has taken into account the energy consumption issue, since several studies have shown that energy consumption is a critical aspect for mobile applications [25, 28] and can influence performances on some user experience metrics [8]. In particular, battery lifetime is one of the most important aspects considered by users when dealing with applications on mobile devices. For this reason, applications developers must avoid to waste energy since this can require a user to recharge frequently the smartphone. Therefore, data acquisition frequency must be carefully considered since it is an extremely energy consuming task. Similarly, our approach gathers from data strictly necessary information only, avoiding the use of a high number of features computations that could lead the final classification to become too expensive.

We first implemented the game to prove the feasibility of the stairstep counter: some preliminary results have been presented in [1]. Then we designed the game to be persuasive, and to be able to engage the users. We use a smartphone to avoid requiring the user to buy expensive and bulky devices, since this could be a reason for not using the game. In fact, the use of smartphones allows the implementation of a *pervasive* and *ubiquitous* solution. Moreover, with respect to the first tentative implementation, we have now completely re-designed the game with new social features: the game can now use Facebook to save and share results, and to create some multiplayer game modes. An experiment with a group of 13 users has shown that the engagement of friends in the building climbing can increase the number of stairsteps made by the users. Finally, to improve persuasiveness of the game we followed the Fogg Behaviour Model [15] during the whole design process.

Beside a restyling of the game modes and interface, other important issues that were not considered in [1] are considered in this paper. First of all, the dataset and the size of the training set has been increased. Then, a data smoothing phase, used to clean data from errors due to the sensibility and variability of the sensors, has been added to the application pipeline. Finally, a deep analysis has been performed to find the most relevant (and not redundant) features, thus allowing to remove a huge set of features whose computation would represent a waste of battery power. All these novelties allowed to improve results obtained during the first, initial, study.

Other works in literature and commercial apps propose solutions for activity recognition, e.g., running, walking or driving. These approaches collect data for an interval of time, and, after another interval of time for calculation, they guess which activity was performed by the user in that period. Our approach is different, since we aim at counting, in real-time, the number of stairsteps, which is a more difficult task than recognizing an activity during a longer time interval. Moreover, recognizing stairsteps versus simple steps is clearly far more difficult, since data retrieved by sensors are very similar for staisteps and steps, as we will discuss in Section 4.

Since the required data analysis frequency is extremely high (each stairstep just requires about 500ms to be completed), we implemented a data-dependent window segmentation, instead of the traditional sliding window with a fixed time duration, and the classification is applied only if that window is suitable to represent a stairstep. In this way, the number of windows to analyze is reduced and hence the computational cost.

The paper is organized as follows. Section 2 discusses related works. Section 3 presents all the principles behind the design and implementation of the *serious game*. From Section 4 to Section 8 we present all the technical details of the system, following the path from data acquired with smartphone sensors to the classification output. In Section 9 we present the experimental results, in terms of precision of stairsteps recognition and energy consumption. Section 10 describes results of an experiment with 13 participants. We finally conclude in Section 11.

2 Background and Related Works

2.1 Activity recognition

Before the usage of smartphone sensors to perform activity recognition (and in particular step counting), an analysis of performances reached by pedometer during stair climbing was made by Ayabe et al. [6]. The purpose of their experiments was to understand how well pedometers perform during stair climbing and descending. They evaluated three different commercial pedometers and different stepping rate (from 40 to 120 steps$\cdot$min^{-1}). Although they do not distinguish between steps and stairsteps, results showed that pedometer can assess stairstep counting within an error rate of $\pm 5\%$, being a great tool to count number of stairsteps (and steps) made by each user.

Maurer et al. [23] created eWatch, a monitoring system based on the usage of an accelerometer to classify different user activities: running, sitting, climbing stairs and walking. eWatch can be used in six fixed different positions of the body, and data is acquired for each of these positions. Even in this case, the authors stated

[1] http://www.nike.com/us/en_us/c/nikeplus-fuelband

that the more difficult classification is to distinguish between stairs and steps.

Thanks to the increasing performances of smartphones and their ability to acquire lots of data from the surrounded environment, mobile applications using data from sensors for activity recognition and better lifestyle promoting, received considerable interest from the research community.

A first tentative to use a mobile application to recognize user activity can be find in [20]. The authors used a cellphone augmented with a Bluetooth-connected sensor board. The system works for only three fixed positions of the cellphone, the ones considered by the authors asthe most common positions. The authors collected a total of 651 features, but this approach makes their computation and the following classification step computationally very expansive.

Anjum and Ilyas [4] developed an application for *online* activity recognition like walking, running, climbing stairs, descending stairs, cycling, driving, and remaining inactive. They provided an analysis of 5-seconds length windows using several classification algorithms like k-NN, Naive Bayes, Decision Trees, and Support Vector Machines. They collected data from the accelerometer, the gyroscope, and the GPS. The accuracy of the results ranged between 79% (using Support Vector Machine) and 94% (Decision Tree). Their analysis also showed that data retrieved from the gyroscope do not provide any useful information, therefore they removed all the features related to this sensor. Moreover, they stated that the recognition of stairs climbing and descending is a really difficult task, achieving a much lower accuracy, and often stairs climbing is confused with walking (accuracy of 84.6% for going upstairs, 90.5% for going downstairs).

Shoaib et al. [30] analyzed activity recognition using accelerometer, gyroscope, and magnetometer, alone or combined together, at a frequency of 50Hz. They considered four different positions of the smartphone (arm, belt, pocket and wrist). They showed that the magnetometer, both alone or in combination with other sensors, performs poorly since it causes overfitting for the classifier. Moreover, they found that the accelerometer performs better than the gyroscope in recognizing the six different activities, but, in contrast with the results provided by Anjum and Ilvas [4], this work showed that the combination of accelerometer and gyroscope data performs better than the two taken individually.

Brajdic and Harle [9] discussed about walk detection and step counting algorithms with data acquired from a smartphone; the user can choose to carry the smartphone in six different positions. This work discusses some issues in common with our approach, but in our case we count stairsteps (so we need to discriminate between steps and stairsteps which is a more difficult task). The authors evaluated several algorithms using a dataset of 130 walks traces from 27 different users, getting that most of the algorithms are able to detect walking within a trace that contains only walking and idle periods, with a best median error of 1.3 using thresholding technique, but no algorithm is 100% reliable.

Wu et al. [34] analyzed activity recognition using an iPod Touch, acquiring data from the accelerometer and the gyroscope. They recognize activity like sitting, walking, jogging, going upstairs and downstairs. They extracted both time, frequency and Fast Fourier Transform features. They make data analysis offline, using a 2-second window size with data acquisition at 30Hz. They achieved the best results using k-NN. When sitting, walking and jogging, the accuracy is very high (90.1%-94.1%), while up and down stair walking is classified with a lower accuracy (52.3%-79.4%).

2.2 Persuasive mobile applications

Several *serious games* and applications have been developed to promote better lifestyle using smartphone activity recognition.

Ubifit Garden [11], one of the first mobile persuasion system for improving physical wellbeing, uses the wallpaper of mobile phones to dynamically provide feedback about the different types of physical exercises performed by the user. BeWell [19] is a smartphone application that aims at monitoring user behavior and wants to promote wellbeing. The application continuously tracks activities that impact several aspects of daily life of people, like social interactions or physical activity, using smartphone's sensor, and provides several feedback to promote better health. The application is based on a *serious game* and displays the "state" of the user as an aquatic ecosystem that changes behavior of different animals depending on the changes in user wellbeing. The entire system is based on a cloud infrastructure used to analyze the data acquired with the smartphone and to build a model of the user based on his/her behavior. The second version of BeWell, called BeWell+, introduces several improvements [22]. Firstly, daily activity goals are adjusted depending on the person. This means, for example, that expected activity for an older man is different from a young one, or from a student and an employee. Secondly, energy consumption is considered, reducing resources used by the application when monitoring if the behavior of a user tends closely to healthy norms. In this way, less system resources are required.

Authors	Use of cell phone	Real-time	Sensors	Orientation	Window	Activity	Stairstep counting
[6]	No	Yes	Acc., Gyr.	1	-	stairs	Yes
[23]	No	Yes	Acc., Light sensor	6	4 s	Walk, run, sitting, stairs, standing	No
[20]	Yes	-	Acc., Microphone, Bar.	3	15 s	Sitting, stand, walk, stairs, elevator	No
[4]	Yes	Yes	Acc., Gyr., GPS	4	5 s	stand, walk, run, cycle, stairs	No
[30]	Yes	Yes	Acc.	1	4 s	stand, walk, cycle, drive, run	No
[9]	Yes	No	Acc.	6	(*)	Walking	No
[34]	No	No	Acc., Gyr.	1	2 s	Stand, walk, run, stairs	No
[7]	Yes	-	Acc., GPS, WiFi	Free	-	Walk	No
[11]	No	Yes	Acc., Bar.	Fixed	-	Cardio, walk, resistance, flexibility	No
[19, 22]	Yes	Yes	Acc., GPS	Free	-	Walk, drive, stand, run	No
[13]	Yes	No	Acc., GPS	4	-	Walk, run, stand	No
[17]	No	Yes	Bracelet	Free	-	Walk, run, dance, sleep	No
[12]	Yes	Yes	Acc.	Free	-	Steps	No
[5]	Yes	Yes	Acc., Alt., GPS	Free	-	Walk, run	No
[32]	Yes	Yes	Acc., GPS	Free	-	Walk, run, riding	No
[24]	Yes	Yes	Bracelet	Free	-	Walk, run, sports	No
[14]	Yes	Yes	Bracelet, GPS	Free	-	Walk, run, sleep	No
Our solution	Yes	Yes	Acc., Rot.	Free (**)	(***)	Stairstep	Yes

(*) [9] allows different window sizes
(**) The only constraint is not to use the trouser pocket
(***) Our solution uses a data dependent window size

Table 1: Resume of the different approaches presented in Section 2 compared with our solution

HealthyLife [13] is a smartphone application that automatically recognizes users activities. It recognizes activities like walking, running, driving and staying-still. It uses the accelerometer data and the GPS signal. Moreover, it applies *Ambiguity reasoning* to increase classification results and to disambiguate data. The idea is to apply a set of weak constraints that attaches to any possible answer of the classifier a violation cost that depends on the number and type of violated constraints. The right classification answer will be the one with violation cost equals to 0. The final precision of the system ranges from 100% (staying-still) to 73% (walking).

Move2Play [7] is a smartphone application which encourages a healthier lifestyle and motivates users to participate in daily physical activities. It combines daily targets, for short term motivation, with longer term targets, mainly using social elements, i.e., targets sharing, competitions, etc., in order to avoid dropouts from the game and to constantly try to keep users engaged.

2.3 Commercial applications

We also analyzed some commercial applications used to monitor physical activity. Jawbone Up [17] is a wearable system to monitor physical activity, sleep and resting hearth rate. It uses a bracelet to record physical activity and a mobile application to monitor it. It does not distinguish between stairsteps and simple steps, it simply measures the amount of physical activity.

Accupedo [12] is a smartphone application which implements a pedometer using an intelligent 3D motion recognition algorithm. The user is not forced to use the smartphone in a particular position, so this work seems to be similar to our approach, but Accupedo is not able to distinguish between stairsteps and simple steps, but it counts both as steps.

Apple Health [5] is another smartphone application for monitoring physical activity. The application tracks simple steps and the distance of walking or running. Apple Health simulates stairstep counting: it recognizes a flight of stairs, using altimeter, as approximately 10 feet (3 meters) of elevation gain, and it counts it as approximately 16 stairsteps. Therefore, the application does not recognize stairstep but flight of stairs. Moreover, it uses the altimeter sensor that requires more battery energy.

Endomondo [32] is a smartphone application which can be used as fitness tracker. It is able to track a wide set of fitness activities, e.g., running, walking, riding, but it does not perform a fine-grained recognition of these activities, i.e., it is not able to count the number of steps, and therefore, the number of stairsteps. Moreover, it is not able to recognize a stairstep versus a simple step. Finally, it uses external devices like bracelets or a cardio frequency meter.

Misfit and Fitbit produce bracelet for activity and sleep tracking. Misfit Flash [24] is a sporty fitness and sleep tracker. The bracelet can be worn anywhere, it is connected to a smartphone application and it is able to recognize different activities like running, walking, cycling, playing tennis, soccer or basketball. Unlike our application, it does not recognize and count stairsteps, not even flight of stairs. Bracelets produced by Fitbit [14] work in a very similar manner, but they are also able to recognize flight of stairs. Even in this case, the application does not perform a fine grained activity recognition and does not count stairsteps, it only recognizes flight of stairs and count the number of climbed floors.

To the best of our knowledge, commercial applications are not able to count in real-time the number of climbed stairsteps and address a different task, monitoring physical activity and sleep. These works do not count stairsteps but simple steps and, in fact, they either are not able to distinguish a step from a stairstep [17,12,32] or they only calculate the number of climbed floors using the altimeter, as in [5,14], or to approximate the number of stairsteps using the altimeter and the average height of a stairstep, as in [5]. On the other side, our application counts stairsteps in real-time and it is able to distinguis them from normal steps. Additionally, the above mentioned applications often use external (intrusive) device like bracelets [17,32,24, 14], while we aim at creating a more pervasive solution which does not require anything more than a smartphone which, nowadays, is present in almost all users' pockets.

2.4 Comparison

Table 1 provides a comparison between our approach and the others presented in this section. The analysis of the related works suggests several open issues which need a further study. First of all, since we are pursuing a smartphone based real-time recognition, energy consumption is a fundamental aspect, never considered before. For example, the usage of multiple sensors at the same time (accelerometer, gyroscope, magnetometer and GPS) will rapidly drain the battery, causing a lot of stress to the user. For this reason, we rely on the accelerometer and the rotation sensor only, avoiding the usage of more expensive sensors. Moreover, differently from other solutions that recognize an activity over a large window of time, we aim at recognizing each single stairstep, and distinguish it from a simple step. As shown in Figure 3b and Figure 3c, this is not an easy task, since the signal obtained for a stairstep and a step has a similar behavior. Finally, most of the previously proposed approaches impose a set of fixed positions for the smartphone[2] and the training of a different classifier for each supported position. As we will see, we do not impose any position of the smartphone, we only ask to avoid the trousers pocket, and no (expensive) external devices are required to acquire data.

3 Game Design

The aim of *ClimbTheWorld* is to persuade people to choose stairs instead of escalators, thus increasing their physical activity, using *gamification*, i.e., providing a way to have fun while climbing stairs. The whole design process of the game followed the Fogg Behaviour Model (FBM) [15] to improve game's persuasiveness. FBM shows that three elements must converge at the same moment for a behaviour to occur: *Motivation*, *Ability* and *Triggers*. Since the game is not intended for impaired people, *Ability* is not a problem since our target audience is able to climb the stairs. But it is also true that, even if climbing the stairs can be considered an easy activity, it can become tiring or even frustrating if the goal is too far away. For this reason, *ClimbTheWorld* has different *sub-goals*, denoted by the stars in Figure 1(a), to encourage users to never give up.

In the same way, to avoid discouraging the users, the game proposes different difficulty levels: easier levels correspond to a lower number of stairsteps necessary to reach the top of the building. Each stairstep in real life corresponds to one (or more) stairstep in the game: higher difficulty means less stairsteps in the game. Once the user reaches the top of the building, a slideshow of pictures is displayed, showing the view from the top of the building. Different difficulty levels also bring different quality and number of provided photos. Figure 1(e) shows a screenshot of the gallery of the Eiffel Tower.

To improve *Motivation* and *Ability*, we designed four different game modes, which can involve or not the user's friends[3]. The first game mode requires the user to climb a building alone (see Figure 1(a)). Since some building may have a large number of stairsteps, the user can also invite friends to help him/her to climb the building. This second mode is called *Social Climb*. Figure 1(b) shows a screenshot where the user has in-

[2] We must note here that even if most of the commercial apps do not impose a fixed position to wear the smartphone, they often require an external device, which is even worst in terms of cost and intrusiveness of the system.

[3] In this case, we require the user to connect to Facebook and to give the application the right to explorer his/her network of friends.

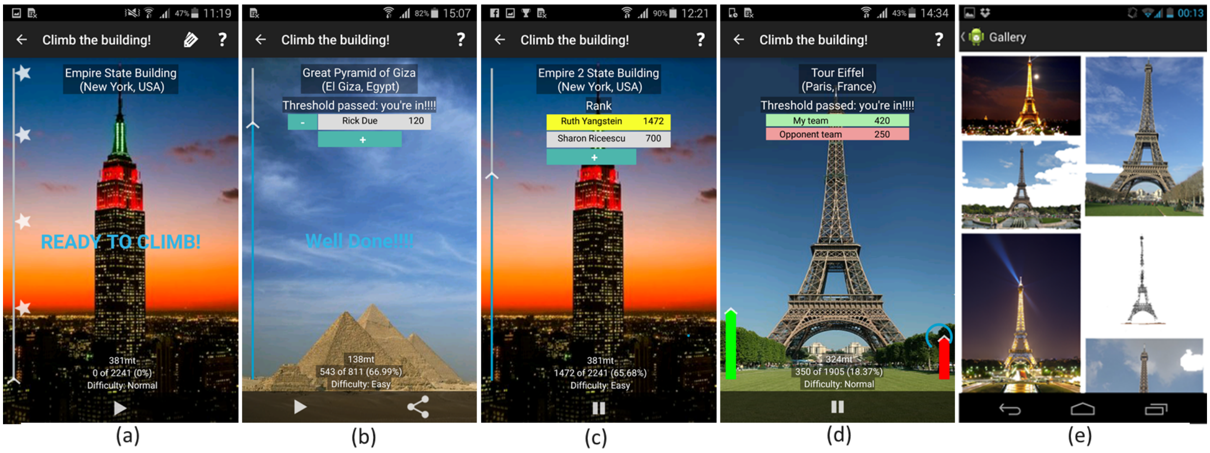


Fig. 1: Five screenshots of *ClimbTheWorld*

vited one of his/her Facebook friends to help him/her to reach the top of the Pyramid of Giza. This game mode improves *Ability*, since it lowers the required number of stairsteps.

Motivation is strongly affected by two other game modes, *Social Challenge* and *Team vs. Team*. The first one implements a challenge between two (or more) players. Differently from *Social Climb*, the players do not collaborate but compete. The winner is the first user who reaches the top of the building. Figure 1(c) shows a screenshot during a challenge between three players. The game reports the number of climbed stairsteps for each user. The player with yellow background is the one who started the challenge and can decide to add or remove other players from the game.

The *Team vs. Team* mode implements a challenge between teams of equal number of players. In Figure 1(d) the two bars on the sides show the progress of the two teams.

Since smartphones are not always connected to the Internet, the game continues working even in absence of network connection. If the game is in one of the multiplayer modes, and some players are not connected, the game pauses at the end of the climbing and waits for data from all the users before declaring the winner.

Another problem related to the multiplayer modes are the management of *freeloaders*, i.e., players who join a team or a social climb but do not contribute to the climb with stairsteps. To avoid this type of players, *ClimbTheWorld* imposes a threshold (see Figure 1(b,d)): players who do not contribute with a minimal set of stairsteps are not rewarded even in case of victory.

The last element of the Fogg Behaviour Model, the *Triggers*, are implemented with a push notification that remembers the player to use stairs when he/she lowers the number of stairsteps climbed during the day with respect to the day before. According to the FBM, this type of *Triggers* are called *signal*. More details about the use of *Triggers* in *ClimbTheWorld* can be found in [10].

To increase the engagement of the user, the game provides a set of bonuses, depending on his/her performances. For example, if the user improves his/her performance with respect to the day before, he/she gets a 30% increase on the total number of stairsteps made. These bonuses are used to constantly encourage and help the user. Since we aim at designing a non-invasive *pervasive* application, we do not fix the smartphone orientation and we consider energy consumption issues.

Finally, the game provides the possibility for a user to share his/her performance with friends through *Facebook* to further increase the user engagement and *Motivation*. Therefore the application, which records confidential data, e.g., user movements and physical activity, has also the possibility to post a message on the Facebook wall of the user. This situation could rise privacy issues, therefore we decided to separate data about user and his/her friends from data about his/her movements recorded through the smartphone sensors, which are recorded and stored separately. The application has only the possibility to share game scores and results, e.g., the winner of a challenge or the result of a climbing, and not the user location or other data about his/her movements. Data about stairsteps and physical activity is stored only locally on the smartphone.

4 Pipeline overview

In this section, we give a broad overview of the system for stairsteps recognition and counting, whose modules will be discussed in details in the following sections.

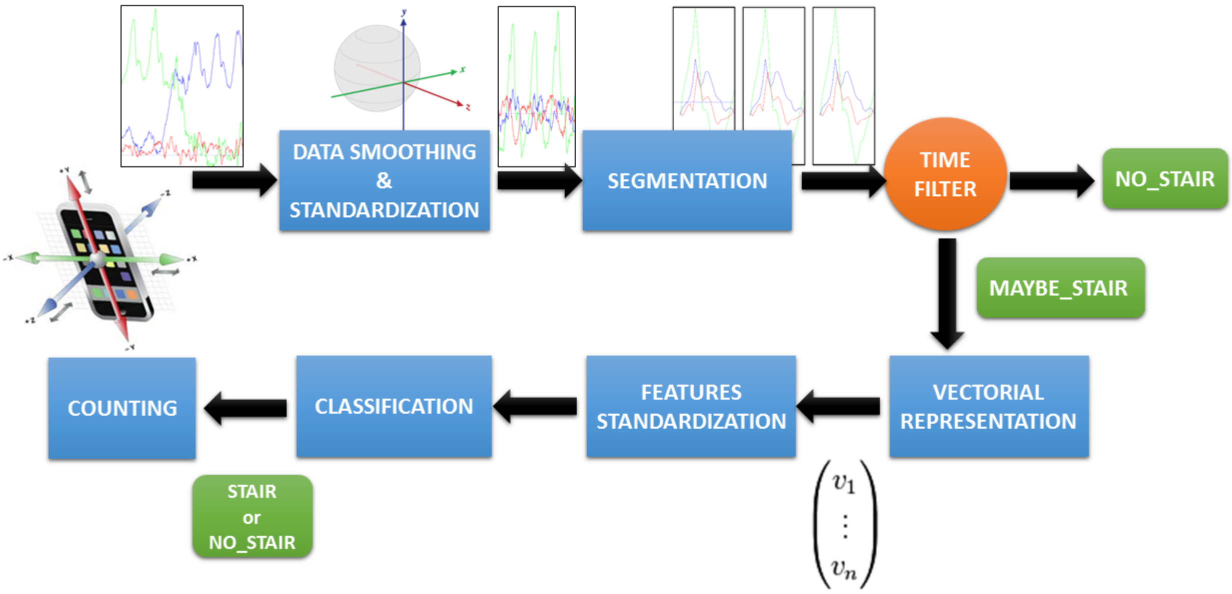


Fig. 2: Pipeline overview of the system

The application's pipeline is depicted in Figure 2. When active, the application constantly acquires data from smartphone' sensors to identify whether the user is climbing or descending stairs. Since it is very important to provide real-time feedback to increase user engagement, the application does not rely on a server that analyzes data and provides classification output, but everything is performed on the smartphone. In this way, we avoid network availability and delay problems. Moreover, the use of a server can create a bottleneck in the system. On the other side, if all the computational tasks reside on the smartphone, energy consumption has to be considered and deeply investigated, to avoid wasting energy and drain smartphone battery.

The first step of the pipeline is data acquisition: the smartphone acquires data about movements of the device through its accelerometer and rotation sensor.

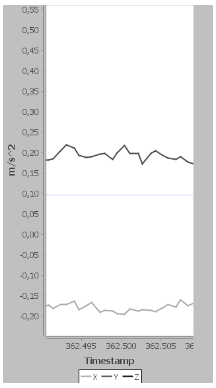
The second step of the pipeline is called "data - smoothing & standardization" and performs two operations. First of all, it cleans data by reducing sensor noise and variability. This operation helps to increase the accuracy of the classification task. After that, data is standardized, since sensor data completely changes depending on how the user carries the smartphone, as we will discuss in Section 6, and the application does not impose a fixed smartphone position. To overcome this problem, smoothed data is translated to a fixed coordinate system, using information from the rotation sensor. In this way, the same activity has the same data pattern independently from the orientation of the smartphone.

The segmentation step splits the standardized data into consecutive segments or windows. In this module, a window segmentation technique based on data analysis (i.e., without fixing the time size of a window) is used to reduce the computational cost, thus reducing energy consumption, and to improve the classification accuracy. The proposed method allows to consider time as a parameter. In this way, windows that do not fall into a predefined time duration, i.e., the time necessary to complete a stairstep, can be automatically labeled as not a stairstep without forwarding it to the next steps (thus reducing energy consumption).

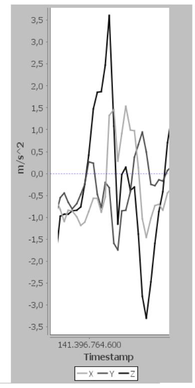
After segmentation, features extraction and standardization for the classifier take place.

The stairstep recognition task is a classification task, where there are two possible labels ("NO_STAIR" and "STAIR") and the examples consist of the vector representations of segmented windows. The training of the model is performed offline only once (one single model shared by all different users), while the recognition phase has to be made in real-time. This implies that the classification outputs must be promptly provided to the user without delays. We must note here that a simple high-pass filter or a peak detection algorithm would not be sufficient in this context. In fact, as it is shown in Figure 3b and in Figure 3c, a stairstep and a step have almost the same shape, that is, a data peak on one axis. For this reason, a more complex method able to distinguish between this two different activities is necessary.

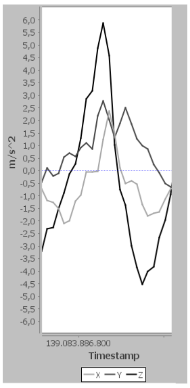
In the following sections, we provide a deep analysis of the main modules compounding the complete sys-



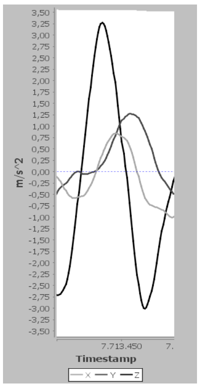
(a) Noise that affects accelerometer data



(b) Example of a step



(c) Example of a stairstep



(d) Example of a stairstep using data smoothing

Fig. 3: Several examples of data acquired from the accelerometer

tem. Finally, we present the results of the classification algorithms in a simulation of the system, an analysis of issues related to energy consumption and the results of a small test for user engagement and behavior change with 13 participants.

5 Data smoothing

One of the biggest problems that affects data retrieved from the accelerometer and any other sensor of a smartphone acquiring real-time information from the environment is the noise that affects this data. This noise is caused, for example, by sensor precision and sensibility, or by the external environment itself. Let us consider, for example, data from the accelerometer. If the smartphone is on a table with the screen facing out the sky, the expected accelerometer data is a series of vectors equal to $v = (0.0, 0.0, 9.8)$, where X- and Y- axes should record values equals to 0.0 m/s^2, since there is no movement, and with the gravity force that influences only the Z-axis, whose value should be equal to 9.8 m/s^2.

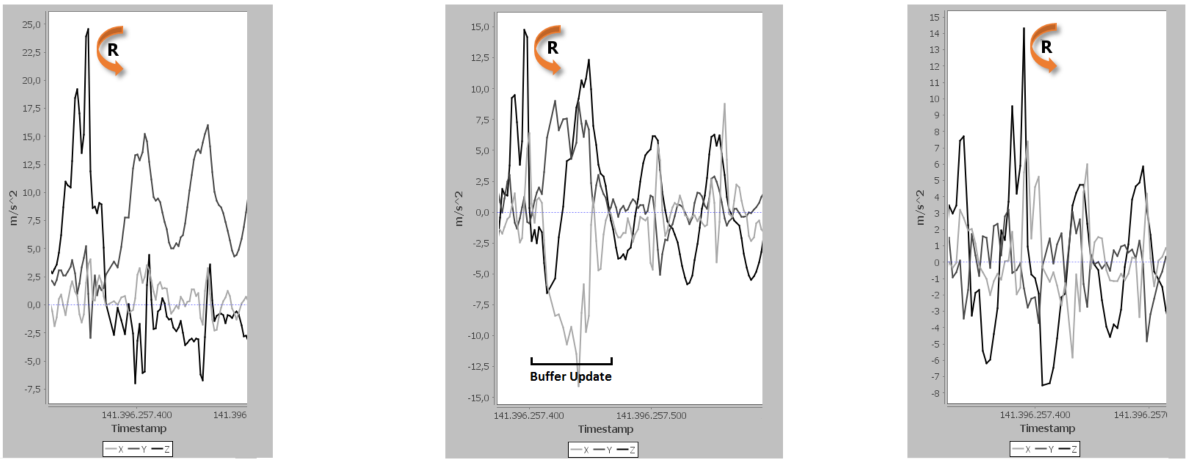
What actually happens is shown in Figure 3a. Each accelerometer axis is affected by both background noise and sensor sensibility that make the signal very unstable. The presence of this noise makes the classification task more difficult, as data is affected by this background error that disturbs the recording of the real movement. The idea behind the data smoothing step is to reduce the effect of background noise and hence to have cleaner data.

The idea is the following. A buffer is used to store the accelerometer records of the last 300ms. The time length of the buffer is short enough to keep all the information useful for the classification task (e.g., it is shorter than the time length of a stairstep). Let us suppose that a new vector value $a = (a_x, a_y, a_z)$ of acceleration data is received from the accelerometer sensor. If the buffer is not full, i.e., its time length is lower than 300ms, the vector a is simply stored into the buffer and nothing more happens. If the buffer is full, the average vector $m = (m_x, m_y, m_z)$ of every reading stored in the buffer is calculated, and the vector m is sent forward to the data standardization step. Then, the original vector a is stored into the buffer. In particular, a cyclic buffer is used, so the first element of the buffer is deleted when a new record is added to the buffer. In this way, the background noise is reduced since variations of a single reading is smoothed from all the other readings.

The result of this technique in a stairsteps pattern is shown in Figure 3, where it is possible to see the same pattern of a stairstep without (Figure 3c) and with (Figure 3d) the data smoothing step. As we can see, the signal is less disturbed and clearer after the data smoothing step, making the learning task, and hence the classification task, easier and more accurate.

6 Data standardization

One of the main issue which is necessary to address when working with activity recognition using data from smartphone sensors like the accelerometer, is smartphone orientation. This problem is related to the fact that data from motion sensors changes depending on how the smartphone is carried by the user. Consider, for example, a walking activity, as depicted in Figure 4a. When the smartphone is rotated to a different position from the initial one (the "R" arrow), the data completely changes. In particular, it is possible to observe how accelerometer data completely switches before and after the rotation of the smartphone. Consequently, the learning task of the classifier becomes extremely difficult. One of the possible solution to this problem is to



(a) Raw data acquired from the accelerometer: same activity but different axis values after rotation of the smartphone

(b) Our method applied to a walking activity while rotating the smartphone

(c) Native linear acceleration plus rotation method

Fig. 4: The comparison among the raw signal (a) and the signal from the methods for the orientation-independence: our method (b), Linear (c). The instant in which the rotation takes place is denoted by the letter "R". Each line represents data acquired with reference to one axis X, Y and Z.

force users to keep the smartphone in a particular position. For example, some previous works followed this approach, but this is clearly a strong limitation, since imposing a fixed smartphone position reduces pervasiveness of our application and its persuasive power. Another solution would be to train the classifier with every possible orientation of the smartphone. This is clearly an unpractical solution, since there are infinite possible orientations, and thus it becomes unfeasible to train a proper classifier.

The proposed solution aims at standardizing data to a fixed coordinate system, independently from the orientation of the smartphone. In this way, the same activity is represented by a similar signal behavior, meaning that how the user carries the smartphone becomes an insignificant information, and the task to train an accurate classifier becomes easier.

A first method for gravity influence removal, to capture the real movement of the user, was proposed by Mizell [26]. The solution is the following: given a sampling interval, the gravity component $\mathbf{g} = (g_x, g_y, g_z) \in \mathbb{R}^3$ on each axis can be estimated by averaging over data read on each axis. When an accelerometer produces the original signal $\mathbf{a} = (a_x, a_y, a_z) \in \mathbb{R}^3$, it is possible to calculate the so called *dynamic component* of $\mathbf{a}$ as $\mathbf{d} = (a_x - g_x, a_y - g_y, a_z - g_z)$ where the influence of the gravity is eliminated. Finally, the vertical part $\mathbf{p}$ of the dynamic component $\mathbf{d}$ (parallel to the gravity) is computed as $\mathbf{p} = (\frac{\mathbf{d} \cdot \mathbf{m}}{\mathbf{m} \cdot \mathbf{m}})\mathbf{m}$, and the horizontal part (orthogonal to the gravity) as $\mathbf{h} = \mathbf{d} - \mathbf{p}$. As a positive aspect, this method requires only accelerometer data, meaning that energy consumption is very low. On the other side, it loose relevant information, since it translates a three-dimensional movement (the one captured by the device) into a two-dimensional coordinate system (vertical and horizontal directions).

The proposed method aims at increasing the precision of the transformation and preserving the orientation-invariance property. How it works and how it changes the signal acquired by the accelerometer is shown in Figure 4b. The fixed target coordinate system is the following: the X-axis is defined as the one tangential to the ground pointing approximately toward East, the Y-axis is tangential to the ground pointing toward the geomagnetic North, and the Z-axis is orthogonal to the ground plane and points toward the sky[4]. To implement this method, a buffer used to estimate the gravity component acting on each axis and data from the rotation vector sensor are necessary.

Firstly, the gravity component is removed from the accelerometer signal using a buffer which stores data acquired during the last 500ms, and averaging over the axes. The vector $\mathbf{g} = (g_x, g_y, g_z)$ is used to compute the dynamic component vector $\mathbf{d} = (a_x - g_x, a_y - g_y, a_z - g_z)$ where $\mathbf{a} = (a_x, a_y, a_z)$ is the original accelerometer reading. Once the vector $\mathbf{d}$ has been computed, it is rotated to the fixed coordinate system using data from the rotation vector sensor. This sensor provides infor-

[4] `http://developer.android.com/guide/topics/sensors/sensors_overview.html`

mation about the current orientation of the device with a three-component vector, where each component represents a rotation angle around an axis. Now, it is possible to apply a three dimensional rotation to the vector $\mathbf{d} = (d_x, d_y, d_z)$ to obtain a new vector $\mathbf{d}' = (d'_x, d'_y, d'_z)$, representing the real movement of the user with respect to the target coordinate system. As we can see in Figure 4b, the signal remains affected from the rotation for a very short interval of time, which is bounded by the buffer size.

Another possibility to capture data about the real movement of the user, e.g., without the influence of the gravity, is to use the linear sensor, natively available on each smartphone. An example of a stairstep data retrieved using the linear sensor is provided in Figure 4c. In the following, we will compare our method for gravity removal against the native solution both from a stairstep recognition and an energy consumption point of view. As we will show, the proposed solution can reach better performances in terms of F_1 evaluation while consuming less energy.

7 Segmentation with data-dependent window

Once data is standardized to a fixed coordinate system, then the data segmentation step is used to divide data into time segments, i.e., *windows*, from which we capture the features that the algorithm uses to understand if the user is climbing stairs or not.

The standard approach for data segmentation is based on *sliding windows*, that is a set of readings of a fixed interval of time, usually a time length sufficiently long to contain the activity to recognize. Since the activity could start at any point inside a *sliding window*, an improvement can be obtained using the *overlapping sliding windows*, in order to increase the recognition probability. In this case, even if time duration remains the same, two consecutive *sliding windows* overlap, typically for about 50% of the total time duration. Figure 5a gives an example of a segmentation using sliding windows which overlaps on 50% of the time length.

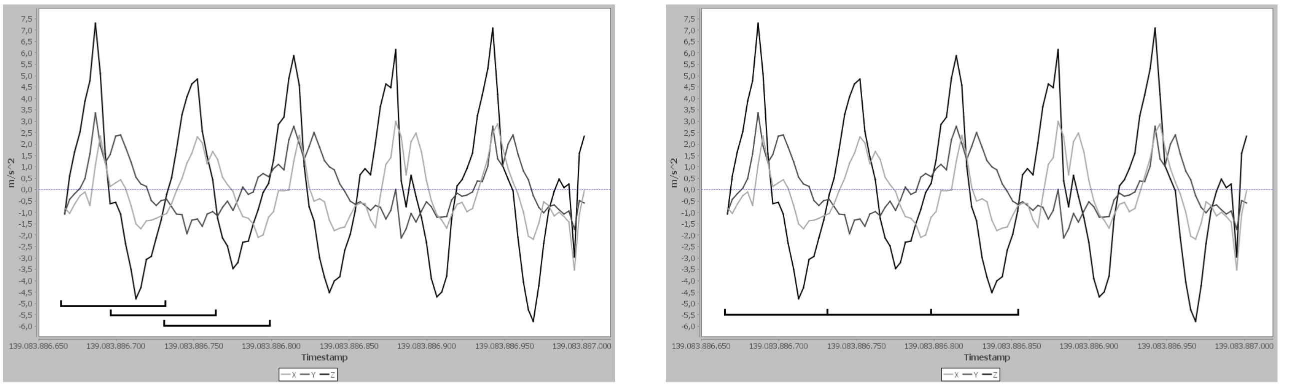
The approaches described above have several drawbacks. First of all, since time duration is fixed, there is no user adaptation, especially for short activities, e.g., the time necessary to perform a stairstep may vary between a child and an elderly. Another issue is that, since data pattern inside a window is not considered, windows are always considered for classification, even when it is clear that they are not the target activity. Therefore, this approach wastes energy, since battery is consumed for activities and analysis not strictly needed. Moreover, the usage of overlapping sliding windows requires to consider the same data twice (for two different windows).

Our approach focuses on data patterns (and not on time) in order to decide the starting and the ending point of a sliding window. Since the data pattern of a stairstep is generally the same, i.e., a local maximum followed by a negative local minimum for the Z-axis, while the X-axis and the Y-axis get a much smaller variation, we suggest to use this pattern to divide data coming from sensors into windows. We must note here that, thanks to the previous orientation-independence step, this pattern is not affected by smartphone orientation. An example of how windows are built is provided in Figure 5b.

Since time length of the windows is no more fixed but becomes variable, it can be used for energy saving purposes. Analyzing the training set with stairsteps, we found out that, on average, people need a time span between 300ms and 2 seconds to complete a stairstep. Using this information, all the windows that do not fall into this time length interval can be automatically discarded and considered as not stairsteps, even when they follow the data pattern. In this case, we can omit the real classification step of our pipeline, hence saving energy. Finally, our approach adapts a classification task to the user variability, since the window are appropriately re-sized to contain a stairstep without the need of a previous step of calibration.

Another solution for data segmentation was presented in [18]. In this paper, authors proposed a solution for data segmentation using a bottom up approach, where data segments are combined together depending on a cost calculated as the error of approximating the signal with its linear regression. If this cost exceeds a threshold value, the two segments are not combined together. It is clear that, if we compare this method with the solution proposed in this section, our approach requires less operations and calculations, thus requiring less battery energy, being more suitable for mobile applications.

Summing up, the proposed solution has several advantages with respect to the time-based sliding window. The first one is that we are sure that each stairstep will fit in a window, and this makes the learning task much more easier. Moreover, it reduces the possible errors in the training set. The second advantage is that just the windows suitable for stairsteps are taken into account for the analysis, reducing the total computational cost. Finally, this approach allows to deal with user variability. In fact, even if different users require different amount of time to complete a stairstep, the window will be appropriately re-sized to contain it without the need of calibration.



(a) Sliding windows using time

(b) Data-dependent sliding windows

Fig. 5: The comparison between the segmentation in windows of possible stairsteps obtained from the time-driven sliding windows (a) and the data-dependent sliding windows (b)

8 Representation and features standardization

In the representation phase, the dynamic window resulting from the segmentation phase is transformed into a vector of real values representing the information obtained by the device sensors contained in that particular time window. This mapping permits a natural application of standard machine learning methods to the task of recognizing whether the user is climbing stairs or not. The way data is represented is crucial for the effectiveness of a learning algorithm. Specifically, a good representation method is able to maintain the information which is really relevant for the task and to reduce the noise in the data as much as possible.

In the following we formally describe our choices for the representation module. In particular, let the frequency of sampling be fixed. Then, each window obtained by the segmentation step will consist of a sequence of n vectors,

$$\mathbf{v}_i = (x_i, y_i, z_i) \in \mathbb{R}^3, \quad i \in \{1, .., n\},$$

each one consisting of the standardized values of acceleration with respect to the three axes.

We also consider two additional computed values that, in our opinion, can carry precious information. Firstly, we consider the norm of the vector $\mathbf{v}_i$ which well represents the global amount of activity being performed and, secondly, the average value of horizontal accelerations, namely $\frac{x_i+y_i}{2}$ which combines the horizontal activities in the fixed coordinate system in a single value.

More formally, for any single standardized vector $\mathbf{v}_i$ of a dynamic window, a new vector $\mathbf{s}_i \in \mathbb{R}^5$ can be built as follows:

$$\mathbf{v}_i = (x_i, y_i, z_i) \mapsto \mathbf{s}_i = (x_i, y_i, z_i, \|\mathbf{v}_i\|_2, \frac{x_i + y_i}{2})$$

Now, the entire sequence of observations in a window can be represented in the matrix $\mathbf{S} \in \mathbb{R}^{5\times n}$ where the vectors $\mathbf{s}_i$ are accommodated in columns. Finally, the representation of the sequence corresponding to the entire dynamic window is obtained by applying a simple transformation $\Phi : \mathbb{R}^{5\times n} \to \mathbb{R}^{79}$, by evaluation of different statistical estimates over the rows of $\mathbf{S}$.

Specifically, the 79 dimensions of $\Phi(\mathbf{S})$ are created evaluating standard statistics (i.e., average, standard deviation, variance and difference between minimal and maximal values) which are very common in time series analysis or evaluating the ratio among the statistical features above and the correlations from different sources [31, 29].

Table 2 presents a detailed description of the entire set of computed features in order to represent a sequence of acceleration measures in a dynamic window. The first set of 20 features considers standard statistics computed over the rows $\mathbf{S}_j$ of the sequence matrix $\mathbf{S}$. The next group of 40 features considers the ratio of the same statistics computed on different rows. An additional group of 9 features (61-69) takes into account other important ratio statistics (among vertical and horizontal activities and the norm of the total activity), and two other features, the Magnitude Area (MA) and the Signal Magnitude Area (SMA) of $\{\mathbf{S}_1, \mathbf{S}_2, \mathbf{S}_3\}$. Finally, the last 10 features correspond to the value of the correlation between pairs of rows. For reason of computational complexity and energy consumption we didn't use the features in the Fast Fourier Transform (FFT) family.

In order to fairly compare our orientation-independent supporting method with other methods presented earlier in this paper, we used a similar representation mapping. In particular, exactly the same mapping is used for the Linear method. Concerning the

Features No.	Description
$1-5$	$Ave(\mathbf{S}_j)$, $j \in \{1,\dots,5\}$
$6-10$	$Std(\mathbf{S}_j)$, $j \in \{1,\dots,5\}$
$11-15$	$Var(\mathbf{S}_j)$, $j \in \{1,\dots,5\}$
$16-20$	$Max(\mathbf{S}_j) - Min(\mathbf{S}_j)$, $j \in \{1,\dots,5\}$
$21-30$	$\frac{Ave(\mathbf{S}_j)}{Ave(\mathbf{S}_h)}$, $j,h \in \{1,\dots,5\}, j > h$
$31-40$	$\frac{Std(\mathbf{S}_j)}{Std(\mathbf{S}_h)}$, $j,h \in \{1,\dots,5\}, j > h$
$41-50$	$\frac{Var(\mathbf{S}_j)}{Var(\mathbf{S}_h)}$, $j,h \in \{1,\dots,5\}, j > h$
$51-60$	$\frac{Max(\mathbf{S}_j)-Min(\mathbf{S}_j)}{Max(\mathbf{S}_h)-Min(\mathbf{S}_h)}$, $j,h \in \{1,\dots,5\}, j > h$
61	$\frac{Max(\mathbf{S}_3)}{Max(\mathbf{S}_5)}$
62	$\lvert\frac{Min(\mathbf{S}_3)}{Min(\mathbf{S}_5)}\rvert$
63	$\sqrt{Ave(\mathbf{S}_1)^2 + Ave(\mathbf{S}_2)^2 + Ave(\mathbf{S}_3)^2}$
64	$Ave(\lvert\mathbf{S}_1\rvert + \lvert\mathbf{S}_2\rvert + \lvert\mathbf{S}_3\rvert)$
65	$\frac{Std(s_1)^2}{Std(s_4)}$
66	$\frac{Std(s_2)^2}{Std(s_4)}$
67	$\frac{Std(s_3)^2}{Std(s_4)}$
68	$\frac{(Std(s_1)+Std(s_2))^2}{Std(s_3)}$
69	$\frac{(Std(s_1)+Std(s_2))^2}{Std(s_4)}$
$70-79$	$Corr(\mathbf{S}_j, \mathbf{S}_h)$, $j,h \in \{1,\dots,5\}, j > h$

Table 2: The 79 features extracted from dynamic windows

Mizell method, that returns two measurements vectors instead of one, we have considered the natural extension of the function Φ applied to the two vectors independently and concatenating the resulting vectors, thus obtaining a new vector of dimension $79 \times 2 = 158$.

Finally, it's well-known that the classification algorithms are badly influenced by features with different orders of magnitude. For this reason, we have rescaled all the features from the dynamic windows used to train the classifiers between $[-1, 1]$ using a linear transformation. These transformations (that are fixed and different for each feature) are applied to all the features before the classification takes place, in order to improve the classification performance.

9 Experiments and results

In this section, we present experimental results obtained considering classification performances, energy consumption and features evaluation.

9.1 Classification setting and results

Our initial experimental results were presented in [1]. Starting from the same dataset, we added a new set of raw information captured from device sensors for any of the three methods (Mizell, Linear and our method). This information was acquired through direct data collection from 6 different users using different smartphones. This data is recorded keeping the devices consistent with the body movement[5] (e.g., hand held, in a backpack or in a handbag) and without other constraints with respect to where or how to keep the device when the data was collected. All the methods presented in Section 8 have been used to compute a vector of features for each dynamic window contained in the whole dataset. Data has been then manually labeled in order to use it with the supervised learning algorithms. Our new dataset is composed by a total of 4128 windows with 1554 stairsteps from 12 people, ranging between 7 and 72 years old.

We use algorithms from three different families to tackle this task:

- *Decision Trees (DT)* generated using the C4.5 algorithm,
- *K-Nearest Neighbors (KNN)* and
- *Kernel Optimization of the Margin Distribution* (KOMD) that is a kernel machine (SVM-like) described in [3], with RBF as the kernel.

For the KOMD algorithm, we used our own implementation[6], while for the Decision Trees and the K-Nearest Neighbors we used the Weka [16] implementation.

Raw data was sampled at three different frequencies, namely 50Hz, 30Hz and 20Hz, with the aim of exploring the relation between performance and energy consumption, to find the best trade-off. Higher frequencies correspond to more accurate information given to the system, but, obviously, they also correspond to an higher energy consumption for the device.

In our application, we face with the binary classification task of stairstep recognition. The distribution of different labels in this case is very imbalanced since we have far more negative examples than positive ones. For this, we evaluated *Precision* and *Recall*, instead of accuracy, to obtain a more proper performance estimation and comparison of different experimental settings. Then, we used the F_β-*score* (with $\beta = 1.0$) to combine recall and precision in a single aggregated effectiveness score. The analytic formulas for these measures are:

$$\textit{Precision} \qquad \frac{TP}{TP+FP}$$

$$\textit{Recall} \qquad \frac{TP}{TP+FN}$$

$$F_1\textit{-score} \qquad \frac{2\ \textit{Precision} \cdot \textit{Recall}}{\textit{Precision} + \textit{Recall}}$$

[5] In this paper, we consider a device movement consistent with the body movement whenever the device movement is negligible relatively to the body's barycenter

[6] Our KOMD implementations in Python and R can be found at `https://github.com/jmikko/EasyMKL`

Frequency 20Hz												
	Mizell			**Linear**			**Rotation method**			**Smoothed**		
	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$
DT	0.81	0.84	$0.825_{\pm 0.012}$	0.67	0.70	$0.685_{\pm 0.024}$	0.77	0.79	$0.780_{\pm 0.015}$	0.73	0.71	$0.720_{\pm 0.010}$
KNN	0.80	0.77	$0.785_{\pm 0.010}$	0.87	0.77	$0.817_{\pm 0.024}$	0.80	0.75	$0.774_{\pm 0.012}$	0.81	0.81	$0.810_{\pm 0.009}$
KOMD	0.83	0.84	$0.835_{\pm 0.011}$	0.83	0.83	$0.830_{\pm 0.018}$	0.84	0.85	$0.845_{\pm 0.008}$	**0.84**	**0.90**	$\mathbf{0.869_{\pm 0.007}}$

Frequency 30Hz												
	Mizell			**Linear**			**Rotation method**			**Smoothed**		
	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$
DT	0.82	0.86	$0.840_{\pm 0.012}$	0.80	0.78	$0.790_{\pm 0.024}$	0.82	0.73	$0.772_{\pm 0.015}$	0.73	0.71	$0.720_{\pm 0.010}$
KNN	0.83	0.84	$0.835_{\pm 0.010}$	0.88	0.61	$0.721_{\pm 0.024}$	0.90	0.78	$0.836_{\pm 0.011}$	0.87	0.85	$0.860_{\pm 0.009}$
KOMD	0.89	0.85	$0.870_{\pm 0.010}$	0.86	0.83	$0.845_{\pm 0.018}$	0.90	0.88	$0.890_{\pm 0.008}$	**0.91**	**0.91**	$\mathbf{0.910_{\pm 0.007}}$

Frequency 50Hz												
	Mizell			**Linear**			**Rotation method**			**Smoothed**		
	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$	Prec	Rec	$F_{1 \pm std}$
DT	0.70	0.77	$0.733_{\pm 0.012}$	0.82	0.89	$0.854_{\pm 0.023}$	0.81	0.75	$0.779_{\pm 0.015}$	0.86	0.83	$0.845_{\pm 0.009}$
KNN	0.82	0.85	$0.835_{\pm 0.009}$	0.81	0.84	$0.825_{\pm 0.024}$	0.90	0.78	$0.836_{\pm 0.011}$	0.82	0.85	$0.835_{\pm 0.009}$
KOMD	0.85	0.88	$0.865_{\pm 0.010}$	0.79	0.86	$0.824_{\pm 0.018}$	0.91	0.90	$0.905_{\pm 0.008}$	**0.92**	**0.92**	$\mathbf{0.920_{\pm 0.006}}$

Table 3: Experimental results of Precision (Prec), Recall (Rec) and $F_{1 \pm std}$ score for the algorithms and methods used with frequency of sampling of **20**Hz, **30**Hz and **50**Hz. The result highlighted has the largest value of $\boldsymbol{F_1}$

where TP stands for True Positives, FP as False Positives and FN as False Negatives.

Starting from the original dataset with manually assigned labels for each dynamic window, the experiments are performed using a user-independent nested stratified 10-Fold cross validation. Firstly, the dataset has been randomly divided in 10 partitions (each partition with the same distribution of stairsteps, i.e. stratified). The best parameters are selected for each algorithm using the classical 10-Fold cross validation among the windows contained in 9 out of the 10 partitions (i.e., the training set). Finally, the machine learning algorithms are trained to generate a model using the best parameters and the classification statistics are evaluated among the windows contained in the remaining partition, used as test set. This procedure is repeated selecting all the 10 partitions as test set and evaluating the average of the performance. The obtained results over the test set are summarized in Table 3.

These results show that smoothing of data improves the proposed method for the support of the orientation-independence. This technique, when combined with the KOMD classification algorithm, obtains the best performances against any other combination of methods and algorithms. The results are satisfactory considering the high difficulty of the classification task and given the small training sample obtained on each single dynamic window. We are also interested in finding the best trade-off between energy consumption and performances. On this respect, it is possible to see that the best compromise is obtained at 30Hz, as we can notice that the difference in performances between 50Hz and 30Hz is not statistically significant considering the standard deviation. Moreover, our method with KOMD shows an higher F_1 score at 30Hz than any other classifier at 50Hz.

9.2 Energy consumption results

When dealing with smartphone devices and applications, energy consumption is one of the key aspects to consider. One of the most important aspect for mobile devices users is battery lifetime, meaning that it is not possible to develop an application that wastes a lot of energy and asks the user to recharge smartphone more than once a day. This kind of situation usually takes the user to remove the application.

A deep analysis of each step of the pipeline is a fundamental requirement during design and development. In our particular case, we studied how resources are used and which are the computationally expensive tasks. Furthermore, we wanted to understand how data frequency acquisition from the accelerometer and the rotation sensor affects energy consumption and in which measure, how much energy the data-smoothing step requires and how well the data segmentation technique performs with respect to the *sliding window* technique.

To measure the energy consumption, we used the Monsoon Power Monitor[7]. Differently form other approaches like background application running on the smartphone (see [25, 28]) that could introduce an unknown and not fixed overhead that could affect measured data, the Monsoon Power Monitor is an external device that directly measures the "requested" energy from the smartphone to the battery. In this way, no overhead is introduced. Thanks to the software for re-

[7] `http://www.msoon.com/LabEquipment/PowerMonitor/`

mote data reading, it is possible to have information about "Energy Consumption", "Average Power" and "Average current", "Expected Battery Life", etc. These information are extremely important, since it is possible to understand how much energy is consumed by the smartphone and the application, for how much time, which are the tasks that are more expensive, etc.

Tests were performed using a Samsung Galaxy i9250 with a 1750mAh battery. Even if this smartphone is not up to date, this is not a real problem since we do not want to consider absolute values of energy consumption, but we aim at comparing energy consumption of different methods to determine which is the less expensive.

Figure 6 reports the energy consumption measurements. Here and in the following discussion we just consider the "Consumed Energy" value, which is the energy the smartphone is requesting to the battery when acquiring data from the sensors (accelerometer and rotation sensor) and running one of the methods we consider in this paper. Each experiment lasted two minutes and we repeated each test three times for each method. We tested our method using the target frequency of 30Hz.

The first evidence we can note from the results is that the Mizell method is the less consuming one. This is not surprising, since this method gets data from the accelerometer sensor only, thus requiring less energy. On the other side, this method shows a significantly poorer performance (see Table 3).

Comparing our method, which uses both the rotation sensor and the accelerometer, and the linear method, which uses the rotation sensor and the linear sensor, we can note that the usage of the accelerometer is less demanding than the usage of the linear sensor. These results, along with the ones presented in Table 3, demonstrate that our solution is able to improve both the accuracy in terms of stairstep recognition and the energy consumption with respect to the linear solution. Moreover, when adding the data smoothing phase, the performance in accuracy are further improved without affecting the energy consumption (only about +0.26%).

Another key aspect of our pipeline is the use of data-driven windows that, differently from standard time-based windows, use the data pattern to determine how to segment data coming from sensors. To compare the two methods in terms of energy consumption, we tested both of them acquiring data at 30Hz and using KOMD as the classification algorithm. For time-based windows, a time length of 500ms and overlapping at 50% were used. Final results are reported in Figure 7.

As we can see, data-driven segmentation allows to save more energy (more than 15%) with respect to the time-based sliding windows. This is because in the time-based approach every windows is classified, while, with the proposed method, only the ones that contains the correct pattern are taken into consideration. Moreover, the time filter let us further reduce the number of analyzed windows, and cannot be applied to standard windows. Clearly, less windows analyzed by the classifier means less energy consumption.

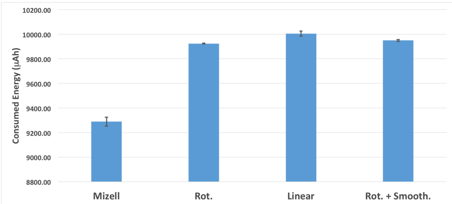
9.3 Irrelevant features

A further analisys has been performed to investigate which features are more useful for our specific task. The motivation for this is three-fold. On one hand we wanted to understand what kind of information is sufficient to recognize stairsteps. On the other hand, removing useless features allows a classifier to learn more quickly, that is, it needs fewer examples to obtain the same performance in classification [27]. Finally, having fewer features to compute can be beneficial from the point of view of energy consumption.

Feature Selection (FS) is a very popular dimensionality reduction technique. FS achieves dimensionality reduction by removing the irrelevant and redundant features and is especially useful for applications where the original features are important for model interpretation and knowledge extraction. In particular, we applied a Multiple Kernel Learning (MKL) approach for feature selection, called EasyMKL [2]. Basically, in the MKL paradigm the kernel used by the classifier (KOMD) is learned from data as a positive linear combination of a set of base kernels. The base kernels in our case are obtained from single features. The weight, and hence the importance, of a feature is given by the value obtained in the linear combination returned by EasyMKL. Given in input the whole dataset, EasyMKL returns an order of the features (from the best to the worst) for our specific classification task. Experiments have shown that it is possible to achieve a performance that is very similar to the one obtained using the whole set of 79 features (F_1-*score* $0.905_{\pm 0.009}$) by keeping only the first 49 most important features. Specifically, the following features: 69, 31, 13, 48, 49, 17, 47, 24, 76, 1, 27, 73, 7, 64, 35, 39, 47, 72, 53, 67, 59, 66, 55, 33, 45, 25, 37, 23, 21 and 57 have been discarded (the indexes refer to the Table 2 and they are sorted in order of importance).

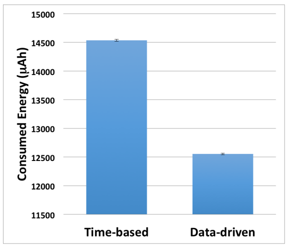
10 Users Tests

In order to evaluate the persuasive power of *ClimbTheWorld*, we conducted a test with real users, at the end of



Energy consumption (μAh)			
Mizell	**Linear**	**Rotation method**	**Smoothing and rotation method**
$9289_{\pm 36}$	$10003_{\pm 21}$	$9923_{\pm 3}$	$9949_{\pm 8}$

Fig. 6: Energy consumption of the four different methods



	Segmentation approach	
	Time-based window	**Data-driven window**
Energy consumption (μAh)	$14535_{\pm 17}$	$12554_{\pm 14}$

Fig. 7: Energy consumption comparison between time-based and data-driven windows

which we provided a small questionnaire to understand their impression of the game.

The test involved 13 participants, 8 females and 5 males, aged between 24 and 30. All the participants are students from the Bachelor and Master degrees in Computer Science at the University of Padua, Italy. No participants were involved in the development process, and they did not have any knowledge about the project before the experiment.

The experiment lasted 9 days, and each participant used his/her own smartphone. We created a new Facebook account for each participant in order to preserve their privacy (no need to use the personal account to ask friendship to someone they did not know) and to provide them full access to all functionality of the game. We did not ask participants to change their daily routine, but only to use freely the application whenever they could and wanted.

Participants were divided into two different groups of 7 and 6 users, respectively. Each participant was randomly assigned to one of the two groups. Both the groups played the same game with all the game modes, we used these two groups to randomize our test. Since we provide two different game modes, singleplayer and multiplayer, we asked one group to use the singleplayer mode first and the multiplayer mode afterwards, while the second group started with the multiplayer modality.

The complete organization and task order of the two groups is provided in Table 4. The first two days we asked participants to use the "Step counter" mode to record a baseline on the number of stairsteps made without using a *serious game*. Then, the first group played two days in singleplayer mode and the following three days in multiplayer mode, and vice versa for the second group. On last two days both groups concluded the experiment using only the "Step counter" modality.

At the end of the experiment, each participant completed a 5-point Likert questionnaire, based on the *IBM Computer Usability Satisfaction Questionnaires* [21], with possible answers ranging from "Strongly disagree" to "Strongly agree". The questionnaire was divided into two different main sections: the first one was used to get a description of the participants taking part to the experiment, e.g., what they think about physical activity and being physically active, which kind of game players they are (single or multiplayer), etc. The second part of the questionnaire contained questions about *ClimbTheWorld*, the different game modes and their experience during the days.

Moreover, during the experiment, a background logger was used to trace user activity with the game. We recorded activities performed with the game and, in particular, the number of stairsteps participants made during each day and with each game mode. In this way, we could understand how the number of stairsteps made changed during the experiment related to the game mode each participants played.

The first part of the questionnaire showed that 53% of participants do not play any sport, while the others 47% play an individual sport, and 77% of participants have positive feelings about being physically active. Moreover, about 61% does not frequently use elevators or escalators, but prefer to take stairs in order to be more active. Finally, 54% of them think they do not need any form of external stimulus to keep

Day	Group A	Group B
1	Stairstep counter	Stairstep counter
2	Stairstep counter	Stairstep counter
3	Singleplayer	Multiplayer (*Social Climb*)
4	Singleplayer	Multiplayer (*Social Challenge*)
5	Multiplayer (*Social Climb*)	Multiplayer (*Team vs. Team*)
6	Multiplayer (*Social Challenge*)	Singleplayer
7	Multiplayer (*Team vs. Team*)	Singleplayer
8	Stairstep counter	Stairstep counter
9	Stairstep counter	Stairstep counter

Table 4: Game mode order for each of the two groups.

them active. From the obtained answers, we can say that participants represent a difficult test case since they already prefer using stairs and have an active life, meaning that, they do not need a serious game to increase physical activity. Considering questions and answers about their relation with video games, they defined themselves mainly as "casual players" (69%), that play almost alone (61.5%) or with another player in the same room (46.2%). Finally, only 23% of participants frequently play with mobile games.

Analyzing participants experience with *ClimbTheWorld*, about 70% of participants preferred to play in single mode, while only 38.5% preferred to play with his/her friends. Moreover, 92.2% of participants liked the *Solo Climb* mode and no one said he/she would not play again with it. These data are in accordance with participants description about preferring singleplayer more than multiplayer: the multiplayer mode obtained less appreciation. In particular, within the multiplayer modes, the most preferred one was the *Social Climb* mode, since all the participants used it at least one time and 92.3% of them ranked it positively. The second preference was *Social Challenge*, played by 77% of participants and 80% of them liked it. Finally, *Team vs. Team* mode was played by 61.5% of participants and 63.6% would play again with it. This rank can be explained by the fact that this mode, that should be the most challenging and engaging one, has the drawback that it is difficult to set up, since it is necessary to find at least 4 users, active in the same interval of time, to be able to start the game (and this could take time that not all users are happy to wait for).

The second step of our analysis has focused on the data acquired with the logger. Data logger registered how many stairsteps participants climbed during the experimental period, and which game mode they used. We compared answers provided with the questionnaire with objective data, and performances of participants depending on the game mode used. Figure 8 shows the number of stairsteps climbed by the participants along the days of our experiment. Together with Table 4, the figure also shows the number of stairsteps climbed by participants depending on the game mode used: the number of stairsteps climbed using the *serious game* (both in singleplayer or multiplayer) is higher than the number when using simply the stairstep counter. In particular, singleplayer mode increased the average amount of about 61%, while multiplayer of about 64%. This means that the game is able to engage the users and is effective in incentivizing people in taking stairs. This result is particularly important since the test groups were made by people that think that they do not need to be incentivized to be physically active.

Moreover, we can note that, when used, the *Team vs. Team* game mode allows to reach the highest number of stairsteps climbed. This probably comes from the fact that this game mode combines both collaboration and competition among users, a combination that is able to engage participants and create high motivation. On the other side, the big difference between the two groups also shows the limitation of this game mode, since the setup phase is longer than other game modes and could reduce users interest.

There is another important difference in the behavior of the two groups. The second group, the one that used the *Team vs. Team* game mode, approximately doubled the number of stairsteps climbed when using the simple counter in the last two days, while the first group lowered the number of stairsteps climbed when not using the game in the last two days with respect to the first two days of the experiment. This means that the *Team vs. Team* game mode is not always accepted by the users, due to the initial setup phase, but, if used, is able to achieve good results in persuading people to change their behavior, and this result remains also in absence of the game. On the other side, singleplayer games were able to engage both groups, showing how an easy entry setup of the game makes it more engaging.

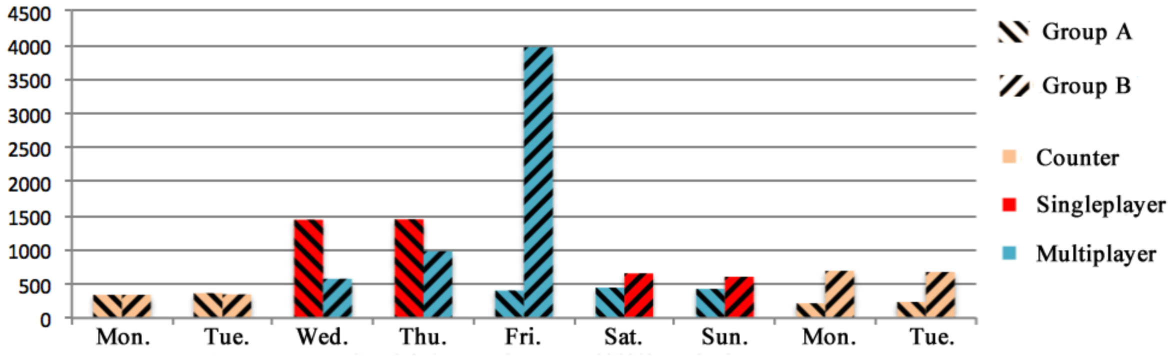


Fig. 8: Number of stairsteps made by both groups each day of the test.

10.1 Limitations of the current study

Although the user study represents the users in a realistic situation, it also has some limits. The main limitation is represented by the restricted data collection period. But, the collected data show an increment in the number of climbed stairstep even in this short time interval, so the result is encouraging.

We have deployed our study on a real user context. Moreover, the study is not affected by problems related to privacy, i.e., that the users could be afraid to ask other people to play the game together, since we provide each user with a test Facebook account and we do not give any information about the association between Facebook accounts and real users to the participants. All the users came from an homogeneous group, they all are students of Computer Science. Clearly, a user study with a more heterogeneous group of real friends would provide further insight.

Given the results achieved from the tests we performed, we can claim that *ClimbTheWorld* is really effective in incentivizing people in taking stairs, and both singleplayer and multiplayer modes are engaging and appreciated by users.

11 Conclusions and Future works

In this paper, we have presented a method for *real-time* stairsteps recognition and counting that is independent from the orientation of the smartphone. This algorithm does not impose constraints concerning the walking speed of the users as it autonomously adjusts the size of the window for its analysis of data. Additionally, a smoothing preprocessing of data makes the performance improve further as it removes the technical noise that affects the smartphone sensors.

We reported several experiments showing that we are able to obtain a very accurate recognition, comparable or even better than the one obtained with more time-demanding methods. In particular, using a state-of-the-art kernel method (KOMD) in synergy with our orientation -invariant preprocessing method, a data-driven window segmentation and the smoothing of the raw data, we achieved the best performances against all the others combinations of methods and algorithms, with a precision of 92% and a recall of 92%, using a sampling rate of 50Hz.

The purpose of this work was to study a classification algorithm inside a real-time smartphone application (namely *ClimbTheWorld*) whose aim is to persuade people to use stairs instead of elevators or escalators. An aspect that has been mandatory to take into account during the development of the application was the control of energy consumption. Our experiments have shown that the new proposed method outperforms the native Android solution in terms of energy saving. The best trade-off between energy saving and precision is reached using KOMD classification algorithm combined with our method for data acquisition, using a sampling rate of 30Hz (in this case, the obtained precision is 91%, recall is 91%).

The *serious game* was tested during an experiment with real users, divided into two groups. The obtained results have shown that the game is able to engage users and to persuade them to change their behavior. In particular, we noted that the *Team vs. Team* game mode is capable to achieve better persistency results, that is collaboration and competitions tend to motivate users to continue their activity during the time.

Although the performance of our method is not much affected by the smartphone orientation, we noticed a performance degradation when the user is carrying the smartphone on his/her trouser pocket (i.e., not consistently with the body).

Future analysis will be dedicated to solve this particular issue. A possible solution could be to exploit additional information, like the proximity or light sensors, to recognize this case. Another possibility could be to analyze the user movement to find out when he/she puts the smartphone on his/her trouser pocket.

Acknowledgements

The authors would like to thank Nicola Beghin, Silvia Segato and Mattia Bazzega for their contribute to the implementation of *ClimbTheWorld*. Moreover, the authors would like to thank the users who have partecipated to the user test, and those who helped to collect stairstep data.

References

1. Aiolli, F., Ciman, M., Donini, M., Gaggi, O.: Climbtheworld: Real-time stairstep counting to increase physical activity. In: Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (Mobiquitous 14), pp. 218–227. London, UK (2014)
2. Aiolli, F., Donini, M.: EasyMKL: a scalable multiple kernel learning algorithm. Neurocomputing pp. 1–10 (2015)
3. Aiolli, F., Martino, G.D.S., Sperduti, A.: A kernel method for the optimization of the margin distribution. In: Proceedings of the 18th International Conference on Artificial Neural Networks (ICANN), pp. 305–314 (2008)
4. Anjum, A., Ilyas, M.: Activity recognition using smartphone sensors. In: Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC 2013), pp. 914–919 (2013)
5. Apple Inc.: Health. http://www.apple.com/ios/whats-new/health/ (2015)
6. Ayabe, M., Aoki, J., Ishii, K., Takayama, K., Tanaka, H.: Pedometer accuracy during stair climbing and bench stepping exercises. Journal of sports science & medicine **7**(2), 249 (2008)
7. Bielik, P., Tomlein, M., Krátky, P., Mitrík, v., Barla, M., Bieliková, M.: Move2play: An innovative approach to encouraging people to be more physically active. In: Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium, IHI '12, pp. 61–70 (2012)
8. Bloom, L., Eardley, R., Geelhoed, E., Manahan, M., Ranganathan, P.: Investigating the relationship between battery life and user acceptance of dynamic, energy-aware interfaces on handhelds. In: Proceedings of the International Conference on Human Computer Interaction with Mobile Devices & Services, pp. 13–24 (2004)
9. Brajdic, A., Harle, R.: Walk detection and step counting on unconstrained smartphones. In: Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13), pp. 225–234 (2013)
10. Ciman, M., Gaggi, O.: Exploiting users natural competitiveness to promote physical activity. In: EAI International Conference on Games fOr WELL-being (2016)
11. Consolvo, S., McDonald, D.W., Toscos, T., Chen, M.Y., Froehlich, J., Harrison, B., Klasnja, P., LaMarca, A., LeGrand, L., Libby, R., Smith, I., Landay, J.A.: Activity sensing in the wild: A field trial of ubifit garden. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '08, pp. 1797–1806 (2008)
12. Corusen LLC: Accupedo. http://www.accupedo.com/ (2015)
13. Do, T.M., Loke, S.W., Liu, F.: Healthylife: An activity recognition system with smartphone using logic-based stream reasoning. In: Proceedings of the International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services (MOBIQUITOUS), pp. 188–199. Springer (2013)
14. Fitbit Inc.: Fitbit. http://www.fitbit.com/ (2015)
15. Fogg, B.J.: A behavior model for persuasive design. In: Proceedings of the 4th International Conference on Persuasive Technology, Persuasive '09, pp. 1–7. ACM, New York, NY, USA (2009)
16. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)
17. Jawbone: Jawbone Up. https://jawbone.com/ (2015)
18. Junker, H., Amft, O., Lukowicz, P., Tröster, G.: Gesture spotting with body-worn inertial sensors to detect user activities. Pattern Recognition **41**(6), 2010–2024 (2008)
19. Lane, N.D., Mohammod, M., Lin, M., Yang, X., Lu, H., Ali, S., Doryab, A., Berke, E., Choudhury, T., Campbell, A.T.: Bewell: A smartphone application to monitor, model and promote wellbeing. In: Proceedings of the International ICST Conference on Pervasive Computing Technologies for Healthcare, pp. 23–26 (2011)
20. Lester, J., Choudhury, T., Borriello, G.: A practical approach to recognizing physical activities. In: Proceedings of the 4th International Conference on Pervasive Computing, pp. 1–16 (2006)
21. Lewis, J.R.: Ibm computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use. International Journal on Human-Computer Interaction **7**(1), 57–78 (1995)
22. Lin, M., Lane, N.D., Mohammod, M., Yang, X., Lu, H., Cardone, G., Ali, S., Doryab, A., Berke, E., Campbell, A.T., Choudhury, T.: Bewell+: Multi-dimensional wellbeing monitoring with community-guided user feedback and energy optimization. In: Proceedings of the Conference on Wireless Health, WH '12, pp. 10:1–10:8 (2012)
23. Maurer, U., Smailagic, A., Siewiorek, D.P., Deisher, M.: Activity recognition and monitoring using multiple sensors on different body positions. In: 2006 International Workshop on Wearable and Implantable Body Sensor Networks (BSN 2006), 3-5 April 2006, Cambridge, Massachusetts, USA, pp. 113–116 (2006). DOI 10.1109/BSN.2006.6
24. Misfit: Flash. http://misfit.com/products/flash (2015)
25. Mittal, R., Kansal, A., Chandra, R.: Empowering developers to estimate app energy consumption. In: Proceedings of the 18th annual International Conference on Mobile Computing and Networking (MobiCom '12), pp. 317–328 (2012)
26. Mizell, D.: Using gravity to estimate accelerometer orientation. In: Proceedings of the 7th IEEE International Symposium on Wearable Computers (ISWC'03), p. 252 (2003)
27. Ng, A.Y.: On Feature selection: Learning with Exponentially many Irrelevant Features Training Examples. In: Proceedings of the 15th Interntional Conference on Machine Learning, pp. 404–412 (1998)
28. Pathak, A., Hu, Y.C., Zhang, M.: Where is the energy spent inside my app?: Fine grained energy accounting on smartphones with eprof. In: Proceedings of the 7th

ACM European Conference on Computer Systems (EuroSys '12), pp. 29–42 (2012)
29. Pirttikangas, S., Fujinami, K., Nakajima, T.: Feature selection and activity recognition from wearable sensors. In: H. Youn, M. Kim, H. Morikawa (eds.) Ubiquitous Computing Systems, *Lecture Notes in Computer Science*, vol. 4239, pp. 516–527. Springer Berlin Heidelberg (2006)
30. Shoaib, M., Scholten, H., Havinga, P.: Towards physical activity recognition using smartphone sensors. In: Proceedings of the 10th International Conference on Ubiquitous Intelligence and Computing, pp. 80–87 (2013)
31. Siirtola, P., Röning, J.: Recognizing human activities user-independently on smartphones based on accelerometer data. International Journal of Interactive Multimedia and Artificial Intelligence **1**(5), 38–45 (2012)
32. Under Armour ConnectedFitness: Endomondo. https://www.endomondo.com/ (2015)
33. World Health Organization: Physical Inactivity: A Global Public Health Problem www.who.int/dietphysicalactivity/factsheet_inactivity
34. Wu, W., Dasgupta, S., Ramirez, E.E., Peterson, C., Norman, G.J.: Classification accuracies of physical activities using smartphone motion sensors. Journal of medical Internet research **14**(5), e130 (2012)