Evaluation of smartphone capabilities for efficient physical activity recognition

Nikola Jajac[1], Bratislav Predic[2] and Dragan Stojanovic[3]

Abstract – **This paper considers if mobile devices are capable to perform activity recognition locally in an efficient way. Within the paper an application for activity recognition based on accelerometer embedded in a mobile phone is described. An evaluation of the impact that the application has on device performance was performed. It was concluded that mobile devices can perform activity recognition in an efficient way, without significant decrease in mobile device performance.**

Keywords – **physical activity recognition, efficient accelerometer data analysis, performance evaluation, mobile device capabilities.**

I. Introduction

Activity recognition fits within the bigger framework of context awareness. Context-aware systems take into account the current state of the user, as well as her surroundings, enabling a mobile device and application to adapt in an appropriate manner.

Initially, context-aware systems used location as the only aspect that defined user context. With the development of low-cost and low-power sensors (such as accelerometers, gyroscopes, digital compasses, light sensors etc.) and their integration into modern mobile devices, in combination with advances in machine learning, it is possible to create a much richer model of user context.

Activity of a user represents an important aspect of a context, because it directly impacts her ability to interact with the mobile device and applications. Information about the user activity enriches the description of a user context and in that manner enables the system to better adapt its services and resources to user context, which can be performed even without any explicit action from the user. In this way the user can stay more focused on the task at hand.

A sensor whose data is probably the most useful in activity recognition is the acceleration sensor. The acceleration sensor is a sensor which measures acceleration along one, two or three axes. Since the acceleration sensor also detects acceleration due to gravity, it can be used for orientation detection, which is useful information for activity recognition.

The greatest possibilities for application of activity recognition systems lay in the healthcare domain. For example, such systems can be used for elderly care support or for long-term health/fitness monitoring [1]. Current methods for tracking activities, like paying a trained observer or relying on self-reporting are time and resource consuming tasks, and are error prone. An automatic system for recognizing activities could help reduce errors that arise from previously mentioned methods. Also, such system would enable its users to go about their daily routines, while the data collection and processing are done in the background, and do not interfere with current user activities.

Another possibility for application is in the social networking domain. Social networks have an important place in today's society. Existing communication services enable simple exchange of text, images, videos etc., while by using data from sensors, a much richer user context could be shared with friends in a more natural and, for the user, simpler way. Automatic activity recognition would enable users to share their current activity with their friends over a social network without interrupting the user in her activity, consequently moving the interaction between social networks users to a new level.

Activity recognition by using data from an acceleration sensor can be performed in two ways. The first one implies transfer of data from the acceleration sensor to a server, where all of the further processing is done. In this way processor power of a server is utilized and also device battery consumption is decreased, since all of the processing is transferred to the server. Disadvantage of this approach is the necessity to transfer data to the server, and since it is a centralized approach, question of scalability arises. The second approach implies that the whole activity recognition process is performed on the mobile device itself. In this case there is no data transfer to a server, the system is maximally scalable since the data from every device is processed on the device itself, so it makes no difference how many devices perform activity recognition simultaneously. On the other hand, there is a question whether mobile devices have enough resources to perform activity recognition seamlessly, or it would cause significant increase in battery consumption and processor load, up until a level when the user could not continue to use other functionalities of the mobile device in a way he is accustomed to.

This paper explores the possibility of activity recognition directly on a mobile device in real time. As a test platform Android operating system was used. The main reason for the selection of Android operating system was the fact that by September 2012. 500 million Android devices were activated [2], and that 1.3 million new devices are activated every day, which represents a huge base of potential users for an activity recognition system. As a part of this paper a demo application for activity recognition was developed and an evaluation of the impact that the application has on a mobile device

[1] Nikola Jajac is with the Faculty of Electronic Engineering at University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: nikola.jajac@elfak.ni.ac.rs.

[2] Bratislav Predic is with the Faculty of Electronic Engineering at University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia.

[3] Dragan Stojanovic is with the Faculty of Electronic Engineering at University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia.

performance was performed, to determine whether a typical mobile device can perform activity recognition in an efficient way. The rest of the paper is organized as follows: section 2 provides an overview of related work on activity recognition, with special regard to a paper which preceded this one and whose results were used in the development of the application for activity recognition. Section 3 describes the developed application for activity recognition on Android mobile devices. Section 4 presents the evaluation of impact that the developed application has on device performance. Finally, section 5 gives the conclusions about the paper.

II. Related work

In recent years there has been a lot of research related to recognizing activities from accelerometer data. In [3] authors used data from 5 biaxial accelerometers worn simultaneously on different parts of the body. Used accelerometers could detect acceleration up to ±10G. Accelerometers were mounted onto hoarder boards and firmly attached to different body parts. Data was collected from 20 subjects performing various everyday tasks without researcher supervision. The following features were computed on sliding windows of accelerometer data: mean, energy, frequency-domain entropy and correlation. A number of classifiers were trained and tested with the calculated data, where decision trees showed the best result, recognizing activities with an accuracy of 84%.

Ravi et al. in [4] attempted to perform activity recognition using a single triaxial accelerometer worn near the pelvic region. Data was collected by 2 subjects performing 8 different activities. Similarly to [3] the features were computed using the sliding window technique. Four features were extracted: mean, standard deviation, energy and correlation. Extracted features were used to train and test 5 base-level classifiers, and in addition to that, 5 meta-level classifiers. Authors concluded that meta-level classifiers in general outperform base-level classifiers and that plurality voting, which combines multiple base-level classifiers, shows the best results. The authors also showed that out of the used features, energy is the least significant one, and that there is no significant change in accuracy when this feature is not calculated.

Kwapisz et al. in [5] tried to recognize activities by using data from a single acceleration sensor, but they used data from an acceleration sensor embedded into a standard mobile phone. These accelerometers typically detect acceleration up to ±2G along three axes. Their research methodology follows the one in [3, 4]. The authors collected data from 29 subjects, extracted 6 basic features and tested 3 classifiers, where multilayer perceptrons showed the best result, recognizing activities with an accuracy of 91.7%. The authors showed that activity recognition can be performed successfully by using acceleration data from a mobile phone.

Work presented in paper [6], which preceded this one, focuses on activity recognition by using an acceleration sensor embedded into a standard mobile phone. The approach for recognizing activities follows the one used in papers [3-5]. By using a specifically designed mobile application data from the acceleration sensor was collected while performing 8 different activities: standing, sitting, walking, running, walking up stairs, walking down stairs, driving a bicycle and doing pushups. Data was collected by a single test user. For calculating features of the signal from the acceleration sensor the *FeatureExtraction* library was developed. The library was developed in the Java programming language, so it could be used on desktop computers and also on mobile devices (primarily Android operating system was considered). One of the main goals in development was the flexibility of the library, and so the library allows: adding of features for extraction, defining of a sensor data source, and defining of components which use the feature extraction results.

The basic classes of the library are shown in Fig. 1. The sensor data source is defined by inheriting the *DataSource* class. Features are added by inheriting the *Feature* class, and the components that use the feature extractions results are defined by implementing the *FeatureExtractionListener* interface.

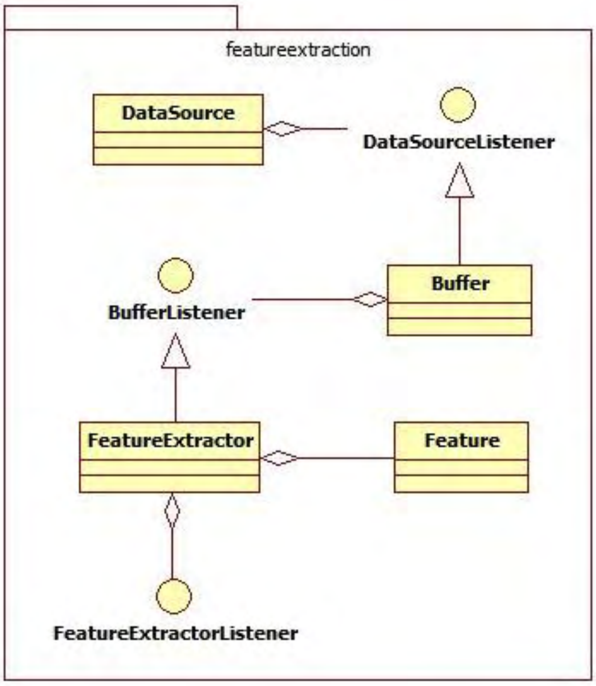


Fig. 1. *FeatureExtraction* library class diagram

Within the paper, the files with recorded data from the acceleration sensor were used as a data source. The library was used to calculate the following features: mean, standard deviation, inter-axis correlation, acceleration vector intensity mean, energy and entropy. Feature extraction results were written into new files together with the name of the activity represented by the source data. Activity recognition was formulated as a classification problem in which classes correspond to activities and attributes correspond to features. The resulting files were used to train and test three classifiers available in the *WEKA Machine Learning Algorithms Toolkit* [7]. Tested classifiers were: C4.5 decision tree, Naïve Bayes and K-nearest neighbors. All three classifiers achieved excellent results in activity recognition, with more than 99% of successfully classified instances.

III. Mobile application for activity recognition

By using results from the paper [6] a mobile application for activity recognition directly on a mobile device in real time was developed. *FeatureExtraction* library was used for feature extraction. In this case the data source is the acceleration sensor itself. Data from the acceleration sensor is read and directly passed to the *FeatureExtraction* library. Diagram of the main application classes and their connections with the *FeatureExtraction* library is shown in Fig. 2.

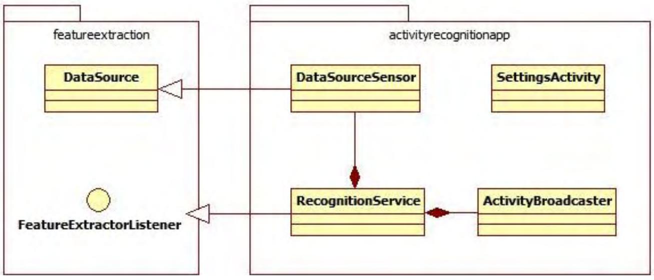
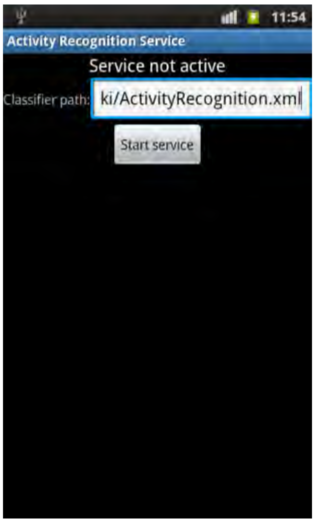


Fig. 2. The class diagram of mobile application for activity recognition

Application for activity recognition consists of two components. The first component is an Android service [8], which performs the task of activity recognition. An Android service is an application component which enables performing of long-running tasks in the background, and as such is ideal for implementation of a system for activity recognition. The service itself has no user interface. The second component is an Android activity [9], which implements a simple user interface and it is used as a front-end for service control. Within the activity a user can start and stop the service, and also define a path to the file with the decision tree definition. The activity user interface is shown in Fig. 3.



Service not active

Classifier path: ki/ActivityRecognition.xml

Fig. 3. Android activity for activity recognition service management

Since all three classifiers from paper [6] showed excellent results in activity recognition, for the implementation on a mobile device, C4.5 decision tree was selected, because it requires the least amount of computation in the classification phase. Specifically, decision tree from the paper [6] was used. To increase the application flexibility, decision tree definition is not coded in the application itself, but in a separate file. In this way, it is simple to change the decision tree definition without any changes to the application itself.

Feature extraction result from the *FeatureExtraction* library is returned to the activity recognition service which performs classification (recognition) of activities by using the externally defined decision tree. The service notifies all of the interested applications about the recognized activity, by using the standardized Android broadcasting mechanism. In this way any application can register to receive information about recognized activities and further process that information in an appropriate way.

Developed application was tested with data from the acceleration sensor in real time, in order to determine application performance when dealing with real life data. The test user performed a subset of activities tested in [6]. Each activity was performed for a specific period of time, with an active application for activity recognition. The results from the application were recorded, and application success in recognizing specific activities was calculated. Table I shows that the results are similar or worse than the ones achieved with recorded data in [6], depending on the activity.

Sitting and standing are two activities which are most easily distinguished from any other activity and were recognized with high accuracy. Even though the signal from the acceleration sensor for walking, walking down stairs and walking up stairs is very similar, walking was also recognized with high accuracy. Walking down and up stairs was recognized with lower accuracy than in [6]. Most of the misclassified walking down stairs instances were classified as walking and walking up stairs, and most of the misclassified walking up stairs instances were classified as walking and walking down stairs, which was expected given the fact the acceleration signal is similar for these activities. Our future research will try to improve recognition and distinction of walking and walking up/down stairs activities by taking into account a location of the mobile user detected by appropriate indoor localization system based on WiFi, Bluetooth or dead reckoning and using data from other available sensors.

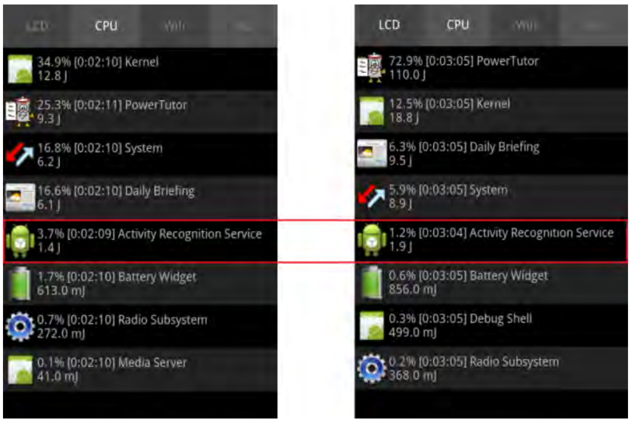
Table I
Application performance in recognizing activities

Sitting	96%
Standing	100%
Walking	97.6%
Walking down stairs	71.4%
Walking up stairs	54.1%

IV. Evaluation of application impact on device performance

Since the activity recognition service is a background process, which is supposed to be transparent for the user, it is very important to perform an analysis of energy consumption and processor load generated by the activity recognition service. Since the basic function of a standard mobile phone is not activity recognition, processor load generated by the service must not be large, so that the performance of other application wouldn't be decreased. Also, battery consumption must not be large, so that the device autonomy wouldn't be decreased significantly. With lower power consumption it would be possible to monitor activities for prolonged periods of time, without the need to recharge the battery. Lower consumption and impact on the performance would lead to a higher degree of service acceptability from a larger number of users. As a test device for the evaluation of application impact on device performance *Samsung I9001 Galaxy S Plus* was used which runs on Android operating system version 2.3.5.

The Fig. 4 shows that, if power consumption generated only by the processor is considered, with an active application for battery consumption measurement *(PowerTutor)*, the service contributes to battery consumption with 3.7%. If power consumption generated by the display is taken into consideration as well, the service part in battery consumption drops to just 1.2%. Since the Android operating system kernel participates in battery consumption with 34.9% when battery consumption generated only by the processor is considered, it can be concluded that the service for activity recognition does not increase battery consumption significantly.

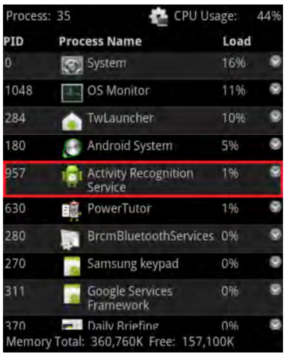


34.9% [0:02:10] Kernel
12.8 J
25.3% [0:02:11] PowerTutor
9.3 J
16.8% [0:02:10] System
6.2 J
16.6% [0:02:10] Daily Briefing
6.1 J
3.7% [0:02:09] Activity Recognition Service
1.4 J
1.7% [0:02:10] Battery Widget
613.0 mJ
0.7% [0:02:10] Radio Subsystem
272.0 mJ
0.1% [0:02:10] Media Server
41.0 mJ

72.9% [0:03:05] PowerTutor
110.0 J
12.5% [0:03:05] Kernel
18.8 J
6.3% [0:03:05] Daily Briefing
9.5 J
5.9% [0:03:05] System
8.9 J
1.2% [0:03:04] Activity Recognition Service
1.9 J
0.6% [0:03:05] Battery Widget
856.0 mJ
0.3% [0:03:05] Debug Shell
499.0 mJ
0.2% [0:03:05] Radio Subsystem
368.0 mJ

Fig. 4. Energy consumption by the activity recognition service

The Fig. 5 shows that, with an active application for processor load measurement (*OS Monitor*), the service for activity recognition participates in processor load with 1-2%. Since the participation in processor load is relatively small, it can be concluded that the service will not have a significant impact on other applications performance.



Process: 35 CPU Usage: 44%

PID	Process Name	Load
0	System	16%
1048	OS Monitor	11%
284	TwLauncher	10%
180	Android System	5%
957	Activity Recognition Service	1%
630	PowerTutor	1%
280	BrcmBluetoothServices	0%
270	Samsung keypad	0%
311	Google Services Framework	0%
370	Daily Briefing	0%

Memory Total: 360,760K Free: 157,100K

Fig. 5. Processor load by the activity recognition service

V. Conclusion

In this paper, an overview of related work on activity recognition was presented, as well as some fundamentals in the development of modern context-aware services. A mobile application for activity recognition performed directly on a mobile device in real time was described. Following that an evaluation of impact that the application has on device performance was performed. It was concluded that the developed application does not increase processor load, or battery consumption significantly. From the aforementioned it can be concluded that mobile devices with a built-in acceleration sensor can perform activity recognition locally in an efficient way, without significant decrease in performance.

Acknowledgement

This paper was realized as a part of the project "Studying climate change and its influence on the environment: impacts, adaptation and mitigation" (43007) financed by the Ministry of Education, Science and Technological Development of the Republic of Serbia within the framework of integrated and interdisciplinary research for the period 2011-2014.

References

[1] J. Lester, T. Choudhury, G. Booriello, “A Practical Approach to Recognizing Physical Activities“, Proceedings of the 4th International Conference on Pervasive Computing, 2006, pp. 1-16.

[2] Wikipedia, Android (operating system), http://en.wikipedia.org/wiki/Android_(operating_system), 10.04.2013.

[3] L. Bao, S. S. Intille, “Activity recognition from user-annotated acceleration data“, Proceedings of the 2nd International Conference on Pervasive Computing, 2004, pp. 1-17.

[4] N. Ravi, N. Dandekar, P. Mysore, M. L. Littman, “Activity recognition from accelerometer data“, Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence (IAAI-05), 2005, pp. 1541-1546.

[5] Jennifer R. Kwapisz, Gary M. Weiss, Samuel A. Moore, “Activity Recognition using Cell Phone Accelerometers”, ACM SIGKDD Explorations Newsletter, vol. 12, issue 2, pp. 74-82.

[6] Nikola Jajac, Bratislav Predic, Dragan Stojanovic, “User activity detection using smartphones with acceleration sensor”, Proceedings of the 56. conference ETRAN (in Serbian), Zlatibor, 11-14. June 2012.

[7] I. H. Witte, E. Frank, “Data Mining: Practical Machine Learning Tools and Techniques”, 3rd ed. Morgan Kaufmann, 2011.

[8] Android Developers, Service, http://developer.android.com/reference/android/app/Service.html, 10.04.2013.

[9] Android Developers, Activity, http://developer.android.com/reference/android/app/Activity.html, 10.04.2013.