Towards a fuzzy-based multi-classifier selection module for activity recognition applications

Henar Martín, Josué Iglesias, Jesús Cano, Ana M. Bernardos, José R. Casar
Universidad Politécnica de Madrid, Telecommunications School
Madrid, Spain
{hmartin, josue, jcano, abernardos, jramon}@grpss.ssr.upm.es

***Abstract*—Performing activity recognition using the information provided by the different sensors embedded in a smartphone face limitations due to the capabilities of those devices when the computations are carried out in the terminal. In this work a fuzzy inference module is implemented in order to decide which classifier is the most appropriate to be used at a specific moment regarding the application requirements and the device context characterized by its battery level, available memory and CPU load. The set of classifiers that is considered is composed of Decision Tables and Trees that have been trained using different number of sensors and features. In addition, some classifiers perform activity recognition regardless of the on-body device position and others rely on the previous recognition of that position to use a classifier that is trained with measurements gathered with the mobile placed on that specific position. The modules implemented show that an evaluation of the classifiers allows sorting them so the fuzzy inference module can choose periodically the one that best suits the device context and application requirements.**

Keywords - activity recognition, mobile technologies, context awareness, personal health applications, fuzzy inference.

I. INTRODUCTION

Activity recognition is a well-explored research field, e.g. developed on the need of supporting prevention, diagnosis and treatment of some healthcare problems [1]. The methods proposed to detect activity have evolved from diaries where the user had to write down an exhaustive registry of his/her daily activities, to automatic recognition systems. These automatic systems usually rely on the deployment of ad-hoc infrastructures (e.g. cameras, RFID tags, etc.) [2] or wearable sensors [3][4].

Nowadays, the penetration of smartphones and their use on a daily basis are growing spectacularly. This fact, together with the integration of new technologies and sensors in those devices, broadens the fields of applications developed relying on those devices. For this reason, some works are starting to focus on how to automatically recognize activity by using the sensors embedded in a smartphone. This approach overcomes the limitations of having to deal with external wearable sensors or infrastructures.

Most of the literature related to activity recognition based on wearable and mobile sensors presents systems which process the information in a centralized element, not considering the design of embeddable strategies for resource-constrained devices. In-device computing may substantially enhance the estimation response time for applications needing real-time continuous data. The design of sound embeddable techniques for activity recognition implies taking into account that they may be very costly in terms of resource consumption and that they will have to coexist with many other modules in the mobile device.

Thus, in this paper we analyze the cost of integrating a set of classifiers to detect user activity in a smartphone and propose a fuzzy method to optimize their use in real time applications by dynamically selecting the best algorithm capable of fulfilling the application's requirements in terms of accuracy and timing.

The paper is organized as follows. Section II focuses on reviewing some works related to performance measurement and use of fuzzy logic for resource management. Section III gives additional details on the problem. The architecture of our embeddable multi-classifier system for activity recognition is addressed in Section IV. Section V gathers implementation aspects driven to support the subsequent performance analysis of the individual classifiers and the fuzzy selection module. Section VI concludes the work.

II. RELATED WORK

Most of the works tackling with embeddable activity recognition aim at finding the best set of features and algorithms to differentiate the activity [5], but they do not analyze the impact of that computation process when the system must run inside the device. One of the works that do address this issue is the one from Lane et al. [6], which have designed BeWell, an application in which the embedded accelerometer, microphone and GPS receiver are used to recognize driving, stationary, running and walking activities, analyzing the RAM and CPU load of the application, as well as its impact on power consumption. Chu et al. [7] have developed Kobe: a tool that performs profiling and optimization of mobile embedded classifiers to achieve an optimal energy-latency-accuracy tradeoff.

Regarding the use of fuzzy-based techniques for dynamic resource selection, it is proven that such techniques are suitable in representing quality measurements by means of natural linguistic rules over imprecise variables and are also quite effective for making real time decisions using incomplete information [8]. In [9], a Fuzzy Inference Scheme module deployed in WSN nodes is used to assess the degree of trust on the data each node offers (QoI^{out}) when aggregating information. Three input variables are evaluated: *(i) completeness* (fraction of nodes responding the aggregation query), *(ii) consistency* (standard deviation on

aggregated data received) and *(iii)* QoI^{in} (quality of information of each aggregated data source). This QoI^{out} score is used to decide whether to employ or not the information offered. Taking into account this model, improvements up to around 20% can be achieved in the accuracy of aggregated data. In the same line, although just a simulation is presented, PalCom project researchers proposed a fuzzy-based process where services *QoS* (Quality of Service) is modeled taking into consideration service intrinsic characteristics and those related to the devices where they are executed (e.g. battery lifetime, processing load and signal strength) [10]. A rule-based fuzzy expert system is used to figure out a 'potential value' scoring each available service, used as criteria to optimize the service selection.

III. PROBLEM STATEMENT

This work focuses on *(i)* integrating and analyzing in practice the performance of a set of embeddable classifiers for activity recognition and *(ii)* developing a module that chooses the best activity classifier to be used, taking into consideration the requirements of the application that is going to use the activity information and the availability of resources in the mobile device (*device context*). The application relies on a set of classifiers based on Decision Trees and Tables, previously analyzed in [11], which have proven to be light enough to be integrated in mobile devices, to recognize the following activities: *slow*, *normal* and *rush walking*, *running*, *standing* and *sitting*. These activities have a direct translation into energy expenditure [12] and are common in standard daily settings (office, home, commuting, etc.).

Previous works on mobile activity estimation [5] have shown that knowing the body position in which a mobile is being carried may enhance the results achieved in user activity classification. In our proposal of light classifiers, we have included the possibility of activating a first stage of body position estimation, capable of delivering information about where the user is carrying the mobile device: in the hand (texting or talking), in the front or back trouser pockets, in the shirt or jacket pockets, in a short or long strap bag, in a backpack, in an armband or in a waist case. Naturally, considering this stage implies increasing the computation time and cost.

In addition, as nowadays devices integrate a large set of sensors, the classification can be carried out profiting from the information gathered by all the available sensors, or by a selected group of them. Accelerometers are the sensors most commonly embedded in different devices (not only phones), so it has been chosen to reduce computational load of the classification. Finally, the set of parameters that can be computed to get significant values that allow differentiating the activities can be composed of time or frequency-domain features, having the latter higher computational cost.

This large amount of sensors, features and algorithms results in a big set of possible classifiers, with different accuracies, latencies and costs. Figure 1 summarizes the characteristics used to determine which is the most convenient classifier regarding the specific context of the mobile device at the moment when the classification must be carried out (a significant part of the list of classifiers is shown in Table I). In an off-line phase, the characteristics of each available classifier must be evaluated in order to be used as classifier selection process input. The features used are the *trained accuracy*, *response delay*, *memory size*, and *complexity*. The requirements of the application related to *accuracy* and *response delay* are also considered. Once the application is running the device context is measured, which is characterized by its *battery level*, *available memory* and *CPU load*. On these data, a rule-based selection module based on fuzzy inference is then used to choose the classifier.

IV. ARCHITECTURE'S DETAILS OF AN EMBEDDABLE MULTI-CLASSIFIER SYSTEM FOR ACTIVITY RECOGNITION

In our approach, an application willing to receive activity data exposes its needs in terms of accuracy and response delay (these parameters are introduced as discrete values, e.g. *low*, *medium* or *high*). Then, a fuzzy selector analyzes the device state and chooses the best classifier to fulfill the consumer application requests. To do so, a previous off-line characterization on the performance of the available algorithms has to be done. Figure 1 details the different modules needed to support the whole process: *(i)* multipurpose modules, performing data acquisition and device monitoring; *(ii)* the classifier evaluation module, for the off-line characterization and *(iii)* the on-line fuzzy selection module. Following there is a detailed description of each module's logic.

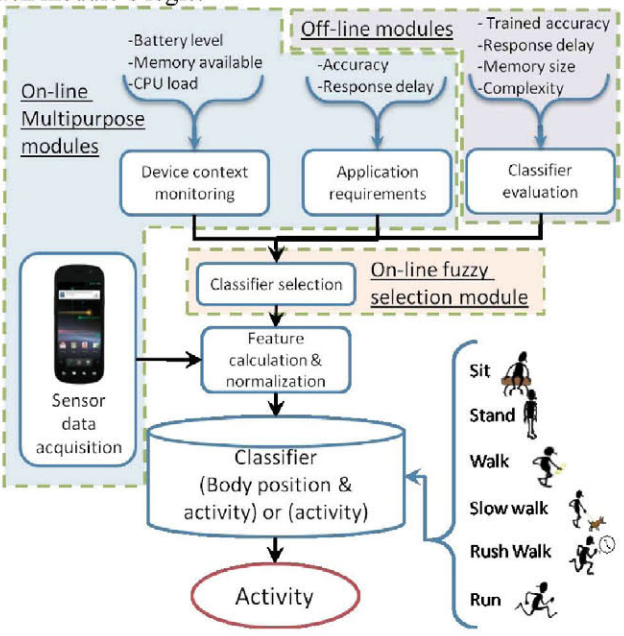

Figure 1. Application modules related to activity classification.

A. Multipurpose modules: Sensor Data Acquisition and Device Context Monitoring modules

An Android enabled mobile device equipped with accelerometer, gyroscope, magnetometer, light and proximity sensors is used (Google Nexus S,). In this way, the Android API facilitates gathering the measurements of these

sensors at a sampling rate that is high enough to recognize the different activities. Regarding sensor data acquisition, the frequency used to gather them is the one labeled as 'normal' in the API, chosen among the ones provided by Android (i.e. normal, fastest, game and user interface), which provides a data rate of around 6 Hz (in this device) for the accelerometer sensor which has proven to be enough to get several data samples from each step during the movement of the user, since the average longest time per step for an individual is not beyond 1.15 s/step (experimentally determined).

The device context is also obtained using the Android tools to gather the information related to the battery level, memory available and CPU load of the different applications running in the device at that specific moment. This information, together with the application requirements, helps selecting the most suitable classifier.

B. *Off-line stage: Classifier evaluation module*

As it has been introduced in the problem statement section (Section III) the set of classifiers considered for this application can be divided in different groups regarding their different characteristics. The most important difference is that some classifiers compute classification regardless of the position in which the mobile is being carried, and others perform the classification in a two stage process: first, a Decision Tree classifier is considered to determine the position of the mobile, and second, the appropriate classifier trained with measurements gathered carrying the mobile on that specific position is used.

Once that first classification is made, the classifiers are subdivided in both branches regarding other conditions that affect resource consumption:

Number of sensors: we have trained *(i)* classifiers taking into consideration accelerometer (using the estimation of gravity and linear acceleration provided by Android tools), gyroscope, magnetometer sensors and the computation of orientation (provided by Android tools based on accelerometer as well as magnetometer measurements) and *(ii)* classifiers that only use the accelerometer. This factor is a tradeoff between accuracy, computational load and size of the classifier..

Sliding windows used to compute the attributes: two possible approaches have been considered when calculating the features: first of all, windows 3 seconds long (18 samples in the accelerometer case) have been chosen, since that length guarantees that the signals gathered from several steps are stored in the same window. Then, the measurements gathered using windows without overlap as well windows with 50 % overlap are used to carry out feature computation. The former case feeds the activity classification stage every 3 seconds and the latter every 1.5 seconds, so the second approach suits better applications with stricter delay constraints. Obviously, the length of the windows and the overlapping percentage could be modified to fulfill the specific application requirements.

Type of features: time-domain features (mean, variance, zero crossing rate, 75th percentile, interquartile, correlation) and frequency-domain features (FFT energy, frequency entropy and power spectrum centroid), together with the signal energy are considered. Due to the fact that frequency-domain features are demanding in computational terms, classifiers are trained with the best set of features among all time and frequency-domain features as well as only with the best of all mean and variance features. This selection is carried out estimating the features the most correlated ones with the class attribute and the most uncorrelated ones among them (WEKA's CfsSubsetEval method [13]).

In this case, Decision Tables and Trees (C4.5) have been used, due to the fact that their computational weight is small and can be easily integrated in a mobile device. The training of the classifiers has been done using WEKA [13] (a tool that provides the implementation of many data mining algorithms). A subset of the list of classifiers that we have considered is available in Table I.

In order to be able to choose the best classifier that fulfills the application requirements at a specific moment, an evaluation of the characteristics of these classifiers must be done. The parameters considered to discriminate them are:

Trained accuracy: to estimate the accuracy that could be expected from a classifier, the one obtained using offline gathered data processed with WEKA in a computer is used.

Response delay: apart from the intrinsic delay due to the windows considered to store sensor measurements, the time necessary to compute features and perform classification is calculated to differentiate slower classifiers from faster ones.

Memory size: the classifiers trained using WEKA have been translated to java objects and classes to carry out classification in the mobile. Their size is considered for comparison.

Complexity: this parameter models the computation complexity of the classifier regarding the number of features that must be computed and their type, since frequency-domain features are more computational demanding than time-domain features.

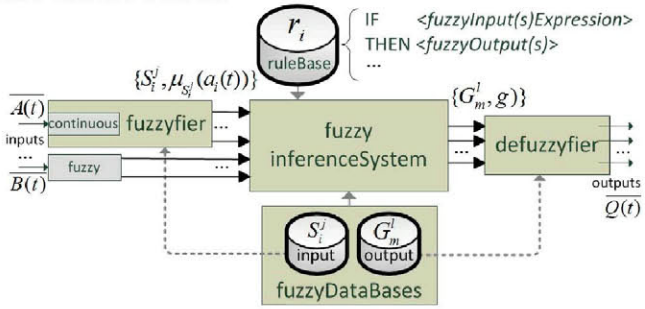

Figure 2. Fuzzy Inference System (FIS) schema offering a quality score $\overline{Q(t)}$ for each particular classifier feature (m), according device context $\overline{A(t)}$ and application requirements $\overline{B(t)}$.

C. *On-line stage: Fuzzy selector*

1) *Fuzzy-based classifier quality computation module*

As a first step in our classifier selector, several quality measurements associated to the desired classifiers have to be performed. These calculations are implemented by means of a FIS (Fuzzy Inference System; see Figure 2). FIS performs a nonlinear mapping between inputs and outputs. The inputs are crisp values that are transformed into fuzzy sets by a

fuzzification process. Input and output fuzzy sets (i.e. fuzzy database) are combined using several fuzzy rules to produce fuzzy conclusions, finally converted into crisp outputs by means of a *defuzzification* mechanism.

Two kinds of inputs are considered in our particular FIS: *(i)* $\overline{A(t)} = [a_1(t), a_2(t), \dots, a_A(t)]$ represents the set of device context random variables describing the available resources of the mobile device at time *t*; *(ii)* the set of application requirements regarding activity recognition process at time *t* are defined by $\overline{B(t)} = [b_1(t), b_2(t), \dots, b_B(t)]$. FIS output $\overline{Q(t)} = [q_1(t), q_2(t), \dots, q_M(t)]$ offers a score indicating the desired quality level, at time *t*, for a particular classifier feature *m* according the corresponding inputs ($m \in \{1,2,\dots,M\}$).

Each input variable $a_i(t)$, is transformed into one or more fuzzy sets by the *fuzzification* process (note that $b_j(t)$ are already fuzzy values). S_i^j, with $i \in \{1,2,\dots,A\}$ and $j \in \{1,2,\dots,L_i\}$, represents the j^{th} fuzzy set associated to variable $a_i(t)$. Each fuzzy set is defined by a membership function $\mu_{S_i^j}(a_i(t))$ that maps $a_i(t)$ to [0,1]. Then, for each input variable $a_i(t)$, the *fuzzification* process obtains L_i pairs $\{S_i^j, \mu_{S_i^j}(a_i(t))\}$. Equivalently, potential $q_m(t)$ crisp outputs are mapped into W_m fuzzy sets G_m^l (with $l \in \{1,2,\dots,W_m\}$ and $m \in \{1,2,\dots,M\}$). G_m^l represents the l^{th} fuzzy set associated to output variable $q_m(t)$.

A set of rules r_i with $i \in \{1,2,\dots,N_R\}$ is then used to combine input fuzzy sets and mapping them into output fuzzy sets. Each rule r_i has the form '**IF** *<antecedent>* **THEN** *<consequent>*'. The *antecedent* is an association of an input fuzzy variable (a_i or b_j) with a fuzzy set S_i^j (several antecedents linked with fuzzy logic operators can be also considered); the *consequent* is an output fuzzy set G_m^l. So each rule offers a pair $\{G_m^l, g\}$, where g represents the membership degree [0,1] for fuzzy set G_m^l. After combining the set of G_m^l pairs, the *defuzzification* process offers a quality score for each classifier feature *m* according to the current inputs.

At present, three device context random variables are considered ($A = 3$) *batteryLevel* $a_1(t)$, *freeMemory* $a_2(t)$ and *cpuLoad* $a_3(t)$, all of them representing percentages ($a_1(t), a_2(t), a_3(t) \rightarrow [0,100]$). The application requirements are defined by $b_1(t) = \{low, medium, high\}$ (*requiredAccuracy*) and $b_2(t) = \{fast, medium, slow\}$ (*requiredResponseTime*) (then, $B = 2$). Four outputs regarding the quality of four classifier features are considered in the FIS ($M = 4$): *accuracyQuality* $q_1(t)$, *responseTimeQuality* $q_2(t)$, *complexityQuality* $q_3(t)$ and *fileSizeQuality* $q_4(t)$ ($q_m(t) \rightarrow [0,1]$). The fuzzy database is formed by $\sum_{\forall i} L_i = 12$ input fuzzy sets and $\sum_{\forall m} W_m = 8$ output fuzzy sets. Eleven rules ($N_R = 11$) have been set by experts in order to control the fuzzy inference process (e.g. '**IF** *batteryLevel=excellent* **OR** *freeMem=excellent* **THEN** *fileSizeQuality=high*'). Antecedents expressions have been combined using Zadeh fuzzy logic *AND* and *OR* operators [14] and consequents are obtained applying Mamdani implication [15]. Crisp outputs are obtained using COG (Center Of Gravity) as *defuzzification* method [16].

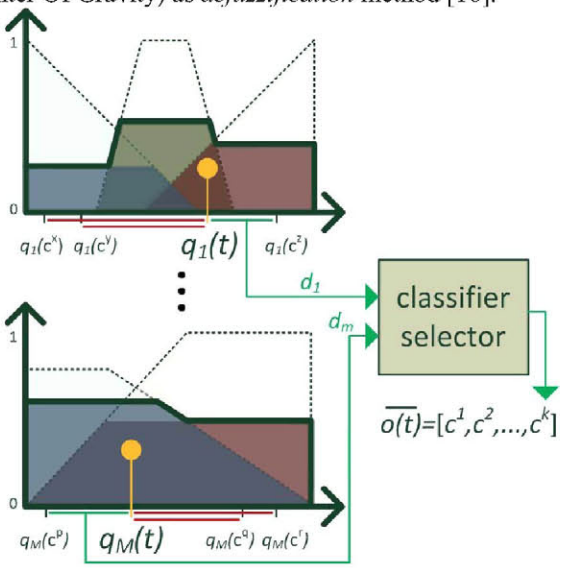

Figure 3. Distance-based classifier selector according classifiers modeling ($q_m(c^k)$) and desired classifier score ($q_m(t)$) for several features (*m*).

2) Distance-based classifier choice method

The above presented fuzzy-based module periodically offers the set of quality scores that best match the application inputs regarding several classification features (i.e. *accuracyQuality*, *responseTimeQuality*, *complexityQuality* and *fileSizeQuality*). Based on the classifiers modeling presented in Subsection IV.B, a distance-based method has been developed in order to calculate the best classifier fulfilling these scores (1)-(4) (see Figure 3). Equation (1) is used to calculate the distance between *(i)* the quality associated to each classifier c^k for a particular feature *m* (i.e. $q_m(c^k)$; remember that this quality has been previously calculated, see Subsection IV.B) and *(ii)* the quality score obtained in the FIS phase ($q_m(t)$) for a particular classifier feature *m*.

$$d_m^k(t) = \|q_m(t) - q_m(c^k)\| \quad (1)$$

Distances in (1) are used to calculate $c^*(t)$: the most adapted classifier having into account the application inputs (i.e. device context and application requirements). This can be easily obtained calculating:

$$d_m(t) = \min_k \left(d_m^k(t)\right) \quad (2)$$

$$m^*(t) = \arg\min_m \left(d_m(t)\right) \quad (3)$$

$$c^*(t) = \arg\min_k \left(d_{m^*(t)}^k(t)\right) \quad (4)$$

It has to be noted that this selection method may generate more than just one candidate classifier. Then, final output can be expressed as a vector $\overline{o(t)} = [c^1, c^2, \dots, c^K]$ indicating if each classifier is recommended ($c^k = 1$) or not ($c^k = 0$) for the given inputs (context and requirements).

V. SYSTEM VALIDATION

In this section the results of the evaluation and the election of the classifiers are explained.

TABLE I. SUBSET OF THE CLASSIFIERS CONSIDERED

Classifiers	*Type of classifier*	*Sensors*	*Windows*	*Type of features*
1	Position Tree	All	None	Instant values
2	Activity Tree	All	Without overlap	Best set of all
3	Activity Table	All	With overlap	Best set mean variance
4	Activity Tree	Acc	With overlap	Best set of all
5	Activity Table	Acc	Without overlap	Best set mean variance
6	Activity Table	All	With overlap	Best of all
7	Activity Tree	All	With overlap	Best set mean variance
8	Activity Tree	Acc	Without overlap	Best set of all
9	Activity Table	Acc	With overlap	Best set mean variance

TABLE II. RESULTS OF THE EVALUATION OF THE MOST SIGNIFICANT CLASSIFIERS

Classifiers	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***
Accuracy (%)	93	93	78	89	76	88.5	98	97.5	83.5
Size (kB)	857	58	154	104	13	17	4	5	7
Response time (ms)	0.25	64.7	7.8	15.3	3.1	6.8	37.3	28	2.2
Complexity	1	0	1	0.25	1	1	0.5	0.25	1

The results of the evaluation of classifiers that take into consideration the position correspond to the ones trained with the data that uses most frequency-domain features, since those are considered the worst case due to their computational requirements.

A. Classifier evaluation

The evaluation of each classifier feature has been carried out as follows (Table II contains the obtained results):

Trained accuracy: this accuracy is obtained by testing the classifiers using leave-one-subject-out method (training the classifier with the data gathered from $N-1$ subjects and testing it on the data corresponding to the last user) using a database composed of 16 subjects who performed the above mentioned activities. The results achieved vary between 76% and 99% accuracy.

Response delay: the Traceview Android tool allows storing when each thread and method started and stopped together with a summary of what happened inside the methods. The delays corresponding to the methods that carry out the computation of the features and the classification are added to calculate the specific time that each classifier needs to perform those operations. The measurements are repeated 20 times; the results are averaged. In addition, the delay measured when carrying out position classification is added to the activity classifiers that need that estimation previous to activity classification. Finally, it is also considered windows with or without overlap used to compute the features that add 3 seconds between classifications in the former case and 1.5 seconds in the latter one. The processing times associated to the computation of the features and the classification vary between 2 and 265 ms. These differences are mainly due to the computation of the features when frequency-domain ones are present, since the classification stage is very fast in the Tree as well as in the Decision Table cases.

Memory size: the values of this parameter are the sizes of the class files of each Decision Table and the size of the binary file that stores the java object that describes the Tree and are used to carry out classification in the mobile. Regarding Table II, it can be seen that the values vary between 4 and 857 kB. The differences are due to the amount of rules in the case of the Decision Table and Leaves in the case of the Tree. Larger values correspond to the cases in which the classifier carries out the recognition regardless of the position of the mobile, since the training has been done using the data from all the positions and more information is needed to distinguish the activities as they are sensed quite differently depending on where the mobile is placed.

Complexity: the complexity of the different classifiers has been chosen to be a discrete parameter that varies between 0 and 1, characterizing 0 the most complex classifiers and 1 the simplest ones. The specific values that have been chosen follow these rules:

$$\begin{cases} 0 & N_{feat} \geq 20 \\ 0.25 & N_{feat} \geq 10 \; and \; Freq_{feat} \neq 0 \\ 0.5 & N_{feat} \geq 10 \; and \; and \; Freq_{feat} = 0 \\ 0.75 & N_{feat} < 10 \; and \; Freq_{feat} \neq 0 \\ 1 & N_{feat} < 10 \; and \; Freq_{feat} = 0 \end{cases} \quad (5)$$

Where N_{feat} is the number of features and $Freq_{feat}$ is the number of frequency-domain features that are computed. The rules remark the fact that that type of features demand higher computational capabilities.

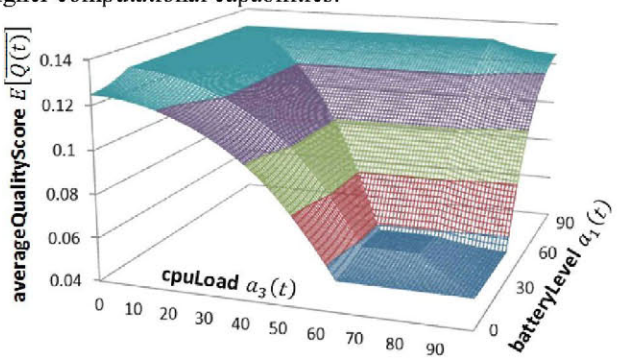

Figure 4. Average quality score calculation based on *cpuLoad* and *batteryLevel*.

B. Fuzzy classifier selection module

As an example to check the correctness of this approach, the average quality score of the fuzzy classifier selector ($E[\overline{Q(t)}] = 1/M \sum_{\forall m} q_m(t)$) has been examined setting some input variables to a fixed value (*freeMemory* $a_2(t) = 80$, *requiredAccuracy* $b_1(t) = medium$ and *required ResponseTime* $b_2(t) = medium$) and evaluating the behavior of the rest of variables (*batteryLevel* $a_1(t)$ and

cpuLoad $a_3(t)$). The associated results are presented in Figure 4, having obtained similar results when evaluating the behavior of other variables.

Random inputs simulation

A simulation generating random inputs has been performed in order to test the behavior of the fuzzy-based classifier selector module.

TABLE III. (A) RELATIVE FREQUENCY OF CLASSIFIERS SELECTION (B) CLASSIFIERS ORDERED ACCORDING EQUATION (6)

A	**C5**	**C6**	**C21**	**C22**	**C29**	**C19**	**C4**	**C10**	...	**C32**	**C12**	**C16**	**C17**
	25.1	24.8	21.7	21.7	16.0	10.4	9.3	6.5		0	0	0	0
B	**C5**	**C6**	**C10**	**C9**	**C21**	**C19**	**C4**	**C22**	...	**C32**	**C12**	**C16**	**C17**
	78.9	77.1	73.7	72.4	67.6	65.1	65.0	64.68		52.8	51.8	51.7	51.4

Within this random environment, the relative frequency each classifier has been selected by the fuzzy-distance module has been calculated (Table III.A). It can be noted that these values are aligned with those presented in Table III.B that represents, as shown in equation (6), the mean distance between classifier score ($q_m(c^k)$) and the midpoint of the range of the classification feature m (MAX_{q_m}), for each available feature *m*. This alignment shows that, for a random inputs simulation and for our particular fuzzy databases and fuzzy rules, our classifier selector *system* (FIS score computation plus distance-based selection) tent to offer those classifiers presenting average characteristics.

$$h(c^k) = \frac{1}{M}\sum_{\forall m} \left\| q_m(c^k) - (MAX_{q_m}/2) \right\| \qquad (6)$$

VI. CONCLUSIONS

This work has analyzed the cost of integrating a set of classifiers used to detect user activity in a smartphone by means of a fuzzy-based method optimizing their use in real time applications. The procedure used to evaluate the characteristics of each classifier in order to be able to choose the most suitable one has proven to be sufficient to sort out the classifiers regarding those characteristics. The results show that the time taken to perform classification is negligible with respect to the time necessary to compute the features when frequency-domain ones are considered. In addition, the size of the classifiers is obviously related to the number of rules or leaves which depends on the consideration of the position in which the mobile is being carried, since the movements differ significantly depending on where the mobile is placed. In addition, the computational load (time and size) is reduced when only the accelerometer is used, since the number of features is also reduced.

Regarding the fuzzy classifier selection, the approach presented in this work has to be considered just as a first step towards a comprehensive integration of this kind of technologies. Next steps have to consider automatic generation of the fuzzy rules used to score classifiers quality (even fuzzy membership functions automatic tuning may be also considered), e.g. using neural networks.

Finally, a real application environment feeding the fuzzy classification module with the required inputs (device context and application requirements) will be also considered in future developments.

ACKNOWLEDGMENT

Henar Martín acknowledges the Spanish Ministry of Education for her grant. This work has been supported by the Spanish Ministry for Science and Innovation under grant TIN2011-28620-C02-02 and IPT2011-1052-390000 and the Government of Madrid under grant S2009/TIC-1485 (CONTEXTS).

REFERENCES

[1] A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, P. Havinga, "Activity Recognition Using Inertial Sensing for Healthcare, Wellbeing and Sports Applications: A Survey," Proc. of the 23rd Conference Architecture of Computing Systems (ARCS 2010), Hannover, Germany. IEEE, pp. 1-10, February 22-25, 2010.

[2] M. Philipose, KP. Fishkin, M. Perkowitz, DJ. Patterson, D. Fox, H. Kautz, D. Hahnel, "Inferring activities from interactions with objects," IEEE Pervasive Computing, Vol. 3, No 4, pp. 50-57, 2004.

[3] D.M. Karantonis, M.R. Narayanan, et al., "Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring," IEEE Trans. on Information Technology in Biomedicine, Vol.10, No 1, pp.156-167, Jan. 2006.

[4] L. Bao, S. S. Intille, "Activity recognition from User-Annotated Acceleration Data", Proc. of Pervasive Computing 2004, Vol. LNCS 3001, Vienna, Austria. Springer, pp. 1-17, April 21-23, 2004.

[5] L. Sun, D. Zhang, B. Li, B. Guo, S. Li, "Activity Recognition on an Accelerometer Embedded Mobile Phone with Varying Positions and Orientations". Ubiquitous Intelligence and Computing, Vol. LNCS 6406, Springer, pp. 548-562, 2010.

[6] N. D. Lane, T. Choudhury, A. Campbell, M. Mohammod, M. Lin, X. Yang, A. Doryab, H. Lu, S. Ali and E. Berke, "BeWell: A Smartphone Application to Monitor, Model and Promote Wellbeing," 5th Int. ICST Conference on Pervasive Computing Technologies for Healthcare, Dublin, 23-26 May 2011.

[7] D. Chu, N. D. Lane, et al., "Balancing Energy, Latency and Accuracy for Mobile Sensor Data Classification", Proc. of ACM Sensys 2011, Seattle, WA, USA. ACM, November 1-4, 2011.

[8] G. Gerla. "Effectiveness and Multivalued Logics". Journal of Symbolic Logic. Vol. 71 (1), pp. 137 - 162, 2006.

[9] E. De Cristofaro, J. Bohli, D. Westhoff,"FAIR: Fuzzy-based Aggregation providing In-network Resilience for real-time Wireless Sensor Networks". Proc. of the 2nd ACM conference on Wireless network security, pp. 253-260, 2009.

[10] G. Prochart, R. Weiss, R. Schmid, G. Kaefer,"Fuzzy-based Support for Service Composition in Mobile Ad Hoc Networks," IEEE Int. Conference on Pervasive Services, pp.379-384, 15-20, July 2007.

[11] H. Martín, A.M. Bernardos, J. Iglesias, J.R. Casar, "Activity recognition using smartphone embedded sensors relying on device position" Internal Report, Data processing and simulation group, UPM, Oct. 2011.

[12] J. Iglesias, A. Bernardos, P. Tarrío, J. R. Casar, H. Martín "Design and validation of a light inference system to support embedded context reasoning," Personal and Ubiquitous Comp., pp. 1-17, 2011.

[13] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, "The WEKA Data Mining Software: An Update," SIGKDD Explorations, Vol. 11, No. 1, 2009

[14] L.A. Zadeh, "Fuzzy logic," IEEE Comput., vol. 21, pp. 89–91, 1988.

[15] E.H. Mamdani, "Application of Fuzzy Logic to Approximate Reasoning Using Linguistic Synthesis," IEEE Trans. on Computers, Vol. 26, no. 12, pp. 1182-1191, December 1977.

[16] S. Roychowdhury, W. Pedrycz,"A survey of defuzzification strategies,"Int. Journal of Intelligent Systems,Vol.16, pp. 679-695, 2001.