Improving the Quality of Life for Elderly by Adapting to Each Specific User

Božidara Cvetković, Erik Dovgan, Boštjan Kaluža,
Mitja Luštrek, Matjaž Gams
Department of Intelligent Systems
Jožef Stefan Institute
Ljubljana, Slovenia
{boza.cvetkovic, erik.dovgan, bostjan.kaluza,
mitja.lustrek, matjaz.gams}@ijs.si

Violeta Mirchevska
Result
Ljubljana, Slovenia
violeta.mircevska@ijs.si

***Abstract*—This paper presents the Confidence system, which aims to prolong independent life of the elderly. It accomplishes this with a set of intelligent modules that recognize falls and general disabilities, and inform the emergency services when a hazardous situation is detected. This way the elderly do not require constant caregivers' monitoring and are not forced to leave their homes since they are confident that they will receive assistance when needed. The Confidence modules adapt to each particular user by initialization before the first use. The initialization requires recordings of several user activities. The results show that the accuracy of the system significantly increases if the modules are properly initialized.**

Keywords-Confidence, elderly care system, fall recognition, general disability recognition, adaptation to the end user, active learning

I. INTRODUCTION

The percentage of elderly people in the modern societies is rapidly increasing. Consequently, the request for caregiver assistance and its costs are also increasing. Nevertheless, the elderly people would prefer to live an independent life without a need for such assistance, but they raise a concern that nobody could help them in case of an accident or a sudden health problem. A solution would be to create a caregiver system that is less expensive and more efficient than a caregiver assistance, and enables an independent life for the elderly.

An efficient caregiver system has to monitor the elderly 24 hours a day and has to be non-intrusive to enable independent life at home [1]. In order to develop such systems, the main problems limiting the independent life of the elderly have to be identified. Such problems range from a large set of possible illnesses to the falls as a consequence of deteriorated motor abilities. An efficient caregiver system has to detect falls and discover various illnesses or general disabilities. When such states are discovered, the system has to report them to an emergency center, a caregiver institution or a hospital. Consequently, such a system does not completely replace the caregiver institution. Instead, it complements it and ensures that its limited resources are used effectively.

This paper presents the Confidence system, developed as part of the FP7 Confidence project [2], which aims to reduce the dependency of the elderly people on the caregiver institutions. It uses position and acceleration data from tags placed on the user's clothes in order to recognize falls and general disability. The recognition procedure is performed in three steps. Firstly, the input data has to be filtered. Next, the activity of the user has to be recognized. Finally, the falls are recognized using activity information, and general disabilities are recognized by finding deviations in the user's behavior. When events such as falls and general disabilities are recognized, an emergency center is informed that the user needs help.

In the real-world usage the Confidence system monitors only one person. Therefore, the system is adapted to the monitored person, which results in an increased accuracy. The adaptation is done during the system initialization and the system usage with active semi-supervised learning. In both stages the built-in classifiers for activity recognition are updated with the user-specific data.

This paper is organized as follows. Firstly, the proposed caregiver intelligent system is presented. Secondly, the initialization procedure is described. Thirdly, the active semi-supervised learning for improving the activity recognition classifiers is given. Fourthly, the proposed methods are tested and the results are presented. Finally, the paper concludes with ideas for future work.

II. SYSTEM ARCHITECTURE

The presented system recognizes falls and general disabilities using position and acceleration of tags attached to the user's clothes [3]. These input data are received from the Ubisense localization system [4] and acceleration system [5]. These systems process and send the data from four tags located on the user's ankles, chest and belt. When the system receives these data, it processes them, analyzes them, and produces an alarm if a fall is recognized, or a warning if a general disability is detected. These messages are sent to the user via a basic Portable device or an advanced Control panel on the computer screen. If the user does not cancel the alarm or warning, an emergency center is informed. The system modules and data flow are shown in Fig. 1 and described in the following sections.

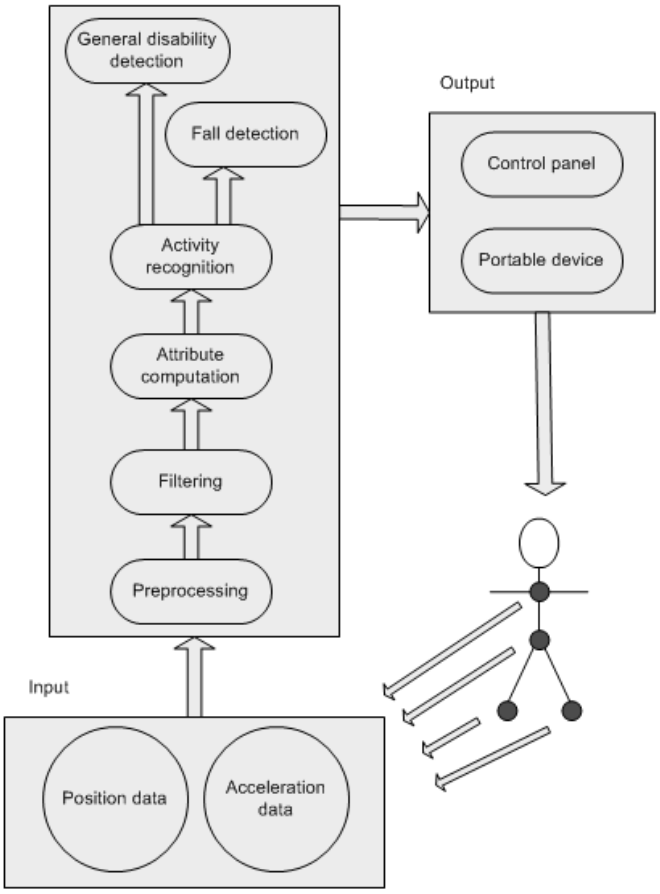

Figure 1. System modules and data flow.

A. *Preprocessing*

The Preprocessing module receives the input data, i.e., position and acceleration data from the tags. The data from individual tags are not synchronized and neither are acceleration data synchronized with position data. Besides, the frequency of incoming data is not constant even for each single input system. Consequently, the input data has to be synchronized and the information about all the tags is combined into single snapshots. A snapshot stores the information about all tags, i.e., their positions and accelerations, at one moment in time. When a snapshot is created, it is passed to the Filtering module.

B. *Filtering*

The Filtering module is used to increase redundancy since the real input data may be inaccurate or even missing, e.g., a missing position of a tag. In order to bypass these shortcomings, the tags' data is filtered and smoothed using six methods. An example of such a method is the anatomic filter that uses anatomic constraints, e.g., the distance between the belt tag and the chest tag that is constant, in order to correct the positions of the tags. In addition, the data of missing tags are estimated using specific heuristic procedures. The output is a snapshot with no missing data and with positions and accelerations that are filtered and smoothed. A snapshot is then passed to the Attribute computation module.

C. *Attribute computation*

The Attribute computation module calculates a set of attributes related to the human body and available tags, e.g., the speed of each tag and the distances between the tags, which are used by the successive modules. When the attributes are calculated, they are added to the snapshot that is then passed to the Activity recognition module.

D. *Activity recognition*

The Activity recognition module recognizes the current user's activity. Examples of the activities are standing, sitting, lying, walking, standing up etc. In order to determine the current activity, two modules are used, namely Random forest classifier [6] and expert-knowledge rules [7]. Their classifications are combined into the final classification using heuristics. The final classification is smoothed with a Hidden Markov Model [8], which eliminates infeasible activity transitions, e.g., from sitting to standing without standing up in between. Afterwards, the assigned activity is added to the snapshot, which is passed to the two main modules: Fall detection and General disability detection.

E. *Fall detection*

The Fall detection module [9] recognizes falls and other potentially dangerous situations, and produces alarms. An example of such a situation occurs when the user lies immovable on the floor for a prolonged time. In order to recognize alarms, two methods are used, namely expert-knowledge rules and two classifiers trained by machine learning algorithms C4.5 [10] and SVM [11]. The two classifications are fused using heuristics.

F. *General disability detection*

The General disability detection module recognizes general disabilities and issues a warning. An example of a general disability is limping. A general disability is recognized by collecting a set of statistics about the user behavior, mainly walking statistics, and comparing them to the past statistics of the same person in order to recognize unusual or changed behavior. If the behavior change is significant, it may indicate the development of an illness/general disability. The changes are recognized using the LOF [12] algorithm.

G. *Control panel and Portable device*

When an alarm or a warning is produced, it is sent to the user, relatives and afterwards to an emergency center, a caregiver center or a hospital. The modules for sending alarms and warnings are implemented in the Control panel and Portable device user interfaces. Both interfaces also enable cancelling an alarm before it is forwarded. The Portable device has a simplified interface that does not enable any other communication with the user. On the other hand, the Control panel has an advanced interface that also shows positions of the tags in the current room, a detailed explanation of the alarms and warnings, a video of the current situation in the room etc. In addition, it displays the history of alarms and warnings upon request. An important issue is that the user can define the protocol of sending warnings and alarms thus adapting communication according to his/her wishes.

H. Initialization of the modules

Since the system is used only by one user, the majority of the modules have the potential to work better if they are adjusted to that user either initially or during usage. For example, when the current activity is calculated, it is more appropriate to use user-specific machine learning classifier than a general classifier. Therefore, an appropriate initialization procedure is crucial for achieving high accuracy of the system. The implemented initialization procedure is described in details in the following section.

III. System initialization

A. Initialization procedure

The purpose of the initialization procedure is to automatically adjust the parameters that are unique for each individual user and improve the accuracy and precision of the Confidence system. The procedure is introduced through a user-friendly interface Initialization Wizard, containing up to 11 steps. These steps can be divided into four sets according to their content. These sets are: (1) system connection and initialization of the modules, (2) basic user information, (3) activity recording, and (4) lying locations. A diagram representing the initialization procedure is shown in Fig. 2. During this procedure the system adjusts the machine learning classifier for activity recognition module by scaling values of certain attributes, calculates user specific expert rules for activity recognition module, creates personal machine learning classifier, and stores the data about the room. The following sections describe the initialization steps in details.

1) System connection and initialization of modules

First, the position and acceleration systems have to be installed and connected to the Confidence system. Second, the following modules in the Confidence system have to be initialized: (1) Preprocessing, where the raw data from the sensors are collected and prepared for further processing, (2) Filtering, where the data is smoothed and corrected with six filters, (3) Attribute computation, where all significant attributes for expert rule generator described in Section B and activity recognition machine learning classifiers are calculated. The acceleration data is not used in the adaptation process. Therefore, the acceleration system can be disabled during the initialization.

2) Basic information

In this step, the user provides information about him/her and his/her requirements. The system collects the following basic information. Firstly, the user has to identify himself/herself with a name or with an anonymous code. Secondly, the user height is stored for classifier adjustment. Thirdly, the user has to determine the time in the day when the collected statistics and possibly warnings are shown on the Portable device or on the User screen [13] if the user does not use the Portable device. Finally, the duration of the activity recordings has to be defined according to the user's ability to perform basic activities such as standing, lying and sitting. The activity recording is described in the following section. The default duration of a recording is 30 seconds. The user can choose to perform one activity for up to ten minutes.

3) Activity recording

The activity recording procedure consists of four to seven steps, depending on the vitality of the user. The user should be able to perform the basic activities: standing, sitting and lying. Additionally, more advanced activities are optional and can be recorded after the basic activities. These activities are sitting on the ground and being on all fours. The recorded data is used in order to improve activity recognition rules and the machine learning classifier thus increasing the accuracy of the entire system.

4) Lying locations

This step of the initialization wizard stores the lying locations for the current room where the system is installed. Lying locations are all locations where the user is allowed to lie (e.g., bed) or to perform activities that could be considered hazardous by the system (e.g., exercise and yoga).

When the initialization procedure is completed, three files are created. The first file describes the general information about the user; the second file contains the new rules created as described in Section B and used in the Activity recognition module, while the third file contains the data for the personal machine learning classifier for the Activity recognition module. This module is used for active semi-supervised learning as explained in Section IV.

B. Rule adaptation

The rule engine for activity recognition was originally created by the domain expert using the knowledge stored in a decision tree classifier and domain knowledge [7]. An example of a rule in the rule engine is as follows: "IF coordinate$Z$(chest) < 0.3 m and velocity(chest) ≈ 0 THEN lying". However, this is a general rule not specialized for a specific user (body dimensions) and a specific room (e.g., low chair). The rules in the rule engine for activity recognition need to be adjusted to suite each particular user (e.g., to user's height and movement characteristics) as well as the particular system localization hardware in a specific room (e.g., adjustments to perform optimally given the hardware's noise level).

The adaptation of the rule engine encompasses only the adjustment of the limits in the conditions of its rules. In the example above, this would be the values 0.3 m (the z coordinate of the chest) and 0 (the velocity of the chest). The form of the rules stays unchanged as defined originally by the domain expert. Activity recordings obtained with the initialization wizard represent training data for rule engine adaptation. We considered two approaches for the adaptation of the limits in the rule engine: (1) a genetic algorithms and (2) an approach which computes information gain for each attribute included in a rule in order to determine the most suitable rule condition limit. Due to time constraints, the second approach was used for the adaptation of the rule engine during system initialization.

C. Machine learning adaptation

The machine learning module contains the *Default classifier* for activity recognition that is created using position data of several users. Similarly to the rules module, the *Default classifier* may not be suitable for the current user since his/her height might be different with respect to the previously tested

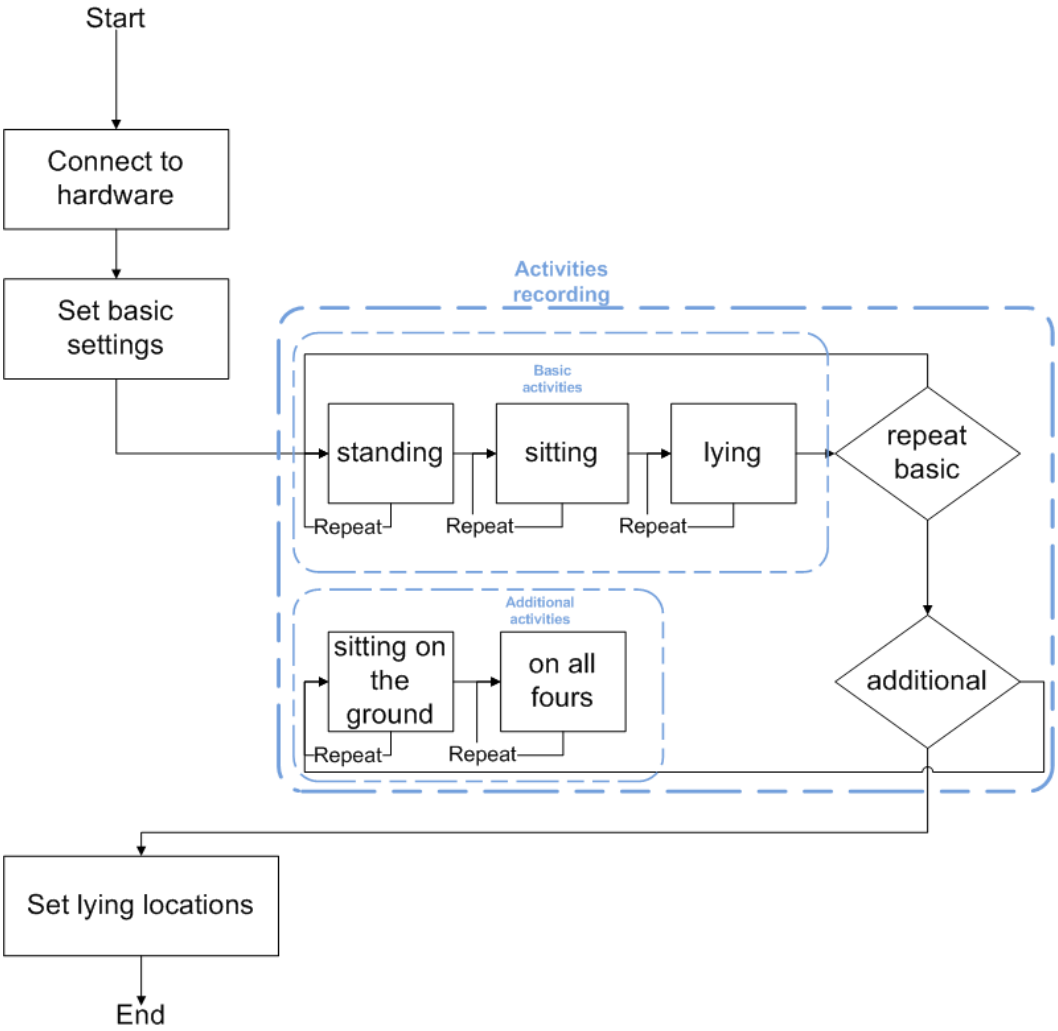

Figure 2. Individual steps within all four sets of the initialization procedure.

users. Consequently, this classifier has to be adapted during the initialization procedure as follows. The initial data has to be scaled by taking into consideration the height of the current user that is stored during the initialization procedure (Section IV.B.1). However, such minor adjustments of the *Default classifier* do not guarantee a high accuracy of the activity recognition module. Consequently, a more advanced procedure for the adaptation to the user has been designed as described in the following section.

In addition to the *Default classifier*, the activity recognition uses the *Person classifier*. This classifier is created by taking into account the data about the activities that are recorded during the initialization procedure as described in Section A.3.

IV. Active Semi-Supervised Learning

In order to increase the accuracy of the machine learning classifier for activity recognition, a combination of the active learning method and semi-supervised learning method has been implemented. The idea of active learning is to improve the accuracy by choosing the data for the training set. The active learning method updates the existing classifier, namely the *Default classifier*, by taking into account the data that has not been included in the existing classifier (e.g., recent data gained after the classifier was created). The implemented method is based on the stream-based selective sampling method. The presented system receives the data from the tags and creates instances in real-time. The instance contains calculated attributes describing distances between tags, velocity of tags and angles between tags. The activities of the instances are obtained with the *Default* and *Person* classifiers. In addition, the confidences of such classifications are also obtained. Since the classifiers may classify an activity into different classes, a new *Meta classifier* is created in order to select the most suitable classification. Afterwards, if the confidence of the classification is sufficient, the instance is used by the active learning method to update the existing *Default classifier*. A similar method to ours was used for video annotation [14]. Their method uses two complementary classifiers for the classification and the decision on the more suitable classifier is based on an effectiveness measure. If the effectiveness measure

is higher than certain value, the video has to be labeled manually.

A. Activity recognition classifiers and Meta classifier

This section describes the three classifiers, namely the *Default*, *Person*, and *Meta classifier*. The first two are classifiers for activity recognition while the last classifier chooses which of the first two should be trusted.

1) Default classifier

The *Default classifier* is a generic classifier. It contains activity data from recordings of three persons. The activities that are recognized by this classifier are lying, standing, sitting, going down, standing up, falling, sitting on the ground, and on all fours. The classifier was built using the Random Forest algorithm. Afterwards, it was tested with leave-one-person-out cross-validation. The achieved accuracy was 86%. By examining the results we have learned that certain physical characteristics of a person can be significant for classifier accuracy. One of these characteristics is the height of the person. In order to test how the user's height affects the accuracy, an additional person with a different height compared to the already tested persons has been tested and the accuracy has decreased to 73%. In order to overcome this shortcoming, the active semi-supervised learning method has been implemented that updates and adapts this classifier which resulted in increased accuracy (see Section V for the results).

2) Person classifier

The *Person classifier* is user specific classifier that contains only three basic activities: lying, sitting and standing. Amount of the data representing each activity depends on the previously chosen time for recording the individual activity. Since the dataset is relatively small, each recorded instance is multiplied four times before the classifier is built with the Random Forest algorithm. In order to define the *Meta classifier* described in the following section, a *Person classifier* was build and afterwards tested with a 30 minutes recording with labeled data of the current user. The achieved accuracy was 69%. The accuracy was low since the recording contains all eight activities while the *Person classifier* can classify only three of them.

3) Meta classifier

The *Meta classifier* is used to decide which of the previously mentioned classifiers is more appropriate for the classification of the current instance. It was built on two recordings with labeled data of the current user. The duration of each recording is 30 minutes. The attributes of the classifier are calculations that reflect statistical relation between the classifiers and attributes that include the classifications of the *Default* and *Person* classifiers. The actually used attributes were selected manually from three different sets of attributes after an extensive testing of several *Meta classifiers*. Each *Meta classifier* was tested with the ten-fold cross-validation. The tested sets of attributes can be seen in Table 1 while the accuracy of the tested *Meta classifiers* can be seen in Table 2. The results show that the most efficient set of attributes is the combination of the first and third set of attributes. The algorithm with the highest accuracy is Random Forest algorithm.

B. Classifier adaptation

The adaptation procedure consists of four steps: 1) *Default classifier* scaling, 2) *Person classifier* creation 3) usage of the *Meta classifier* to label the instances and 4) *Default classifier* update if the confidence in the activity label is high. In case the fourth step is positive, the system updates the learning data for the *Default classifier* thus building a new classifier. The first two steps are done during the initialization procedure (Section III.A). The last two steps are implemented as the active semi-supervised learning that is done during the normal usage of the system. The entire process of the active semi-supervised learning can be seen in Fig. 3.

1) Default classifier scaling

The height of the person is a parameter that significantly affects the classifier accuracy. Consequently, the *Default classifier* may not be efficient. Nevertheless, the data used to build this classifier, which are correlated to the height of the tested users, can be easily scaled to the current user's height using the equation (1) where we calculate the ratio between the height of the current user (h_P) and the average height of people (h_A) whose data is contained in the *Default classifier* and multiply it with the old value of the attribute (*attribute_old*). Consequently, the accuracy of the updated *Default classifier* increases as presented in Section V.

$$attribute\_new = \frac{h_P}{h_A} \cdot attribute\_old \quad (1)$$

2) Person classifier creation

The purpose of the second step is to capture the data of the basic activities of the user and create a new classifier for activity classification. However, this classifier is built considering only the data of the three basic activities while the *Default classifier* is built using the data of all activities. Consequently, the *Meta classifier* in general should use the *Person classifier* to classify three basic activities with higher confidence and accuracy, and the *Default classifier* to classify the other activities.

1) Meta classifier labeling

The *Meta classifier* labeling is done as follows. Each instance that the system receives is classified with the *Default* and the *Person classifier*. In addition, both classifiers return the confidence values of the predicted classes. The attributes of the learning data for the *Meta classifier* are the predictions and confidences of the two classifiers, statistical attributes and attributes calculated with logical functions. The output of the *Meta classifier* is the index of the classifier that is more appropriate for the classification of the current instance.

2) Default classifier update

The update of the *Default classifier* occurs only if the confidence of the classification that is chosen by the *Meta classifier* and the confidence of the *Meta classifier* are sufficiently high. More precisely, both parameters have to be 100% confident in order to use the current instance for the adaptation. Such instances are added to the training set of the *Default classifier* and after a certain time interval a new *Default classifier* is built. Consequently, the *Default classifier* is

adapted to the current user during the classifier update procedure.

The classifier update using active learning procedure stops, when the *Meta classifier* chooses only the *Default classifier* during the last 30 minutes. When active leaning stops, only the *Default classifier* is used for further activity classification.

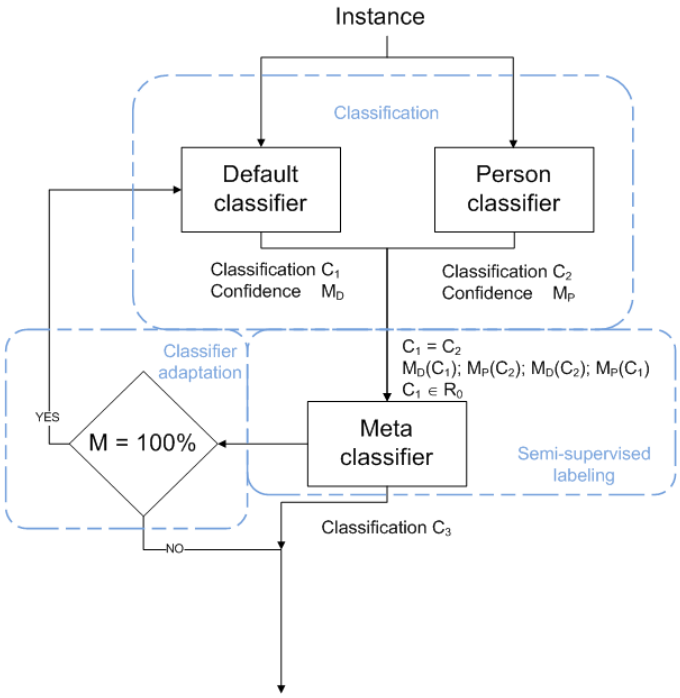

Figure 3. Active semi-supervised learning procedure assembled of classification, semi-supervised labeling and *Default classifier* adaptation.

TABLE I. ATTRIBUTE SETS

Set	Attributes	
	Attribute	***Label/Equation***
1	*Default classifier* classification	C_1
	Person classifier classification	C_2
	Default classifier confidence in C_1	$M_D(C_1)$
	Person classifier confidence in C_2	$M_P(C_1)$
	Is C_1 a basic class	$C_1 \in R_0$
	Are the classes equal	$C_1 = C_2$
2	z coordinate of all tags	/
	Distance in height betwwen neck and average height of ankles	/
	Distance in height betwwen neck and belt	/
3	*Default classifier* confidence in C_2	$M_D(C_2)$
	Person classifier confidence in C_1	$M_P(C_1)$

TABLE II. TESTED ALGORITHMS

Algorithm	Attribute set combination				
	Snapshot + 1	***1***	***2***	***1+2***	***1+3***
SMO	86.6%	92.9%	88.9%	87.8%	88.3%
C4.5	96.8%	95.4%	96.1%	96.6%	95.9%
Random Forest	90.9%	95.9%	96.6%	96.9%	**97.4%**
Naïve Bayes	61.0%	75.7%	70.1%	68.8%	82.3%
AdaBoost	88.6%	84.8%	84.6%	84.6%	79.0%
Bagging	96.9%	94.7%	95.8%	96.2%	95.8%

TABLE III. ACCURACY OF THE CLASSIFIERS FOR EACH ACTIVITY

Activity	Classifiers			
	Default classifier not scaled	***Default classifier scaled***	***Person classifier***	***Default classifier after learning***
Standing	95.5%	98.1%	99.8%	98.4%
Sitting	35.9%	41.7%	100%	97.3%
Lying	81.6%	75.3%	98.3%	93.3%
Sitting on the ground	28.8%	52.0%	0%	84.5%
On all fours	100%	98.0%	0%	71.7%
Going down	52.0%	54.7%	0%	45.3%
Standing up	56.7%	58.6%	0%	74.5%
Falling	3.6%	9.1%	0%	11.0%
Classifier accuracy	**73.0%**	**79.0%**	**69.0%**	**84.5%**

V. EXPERIMENTAL RESULTS

This section describes the experiment of the classifier adaptation procedure consisting of the initialization procedure and active semi-supervised learning. The *Default classifier* used for the experiment has accuracy of 73% when the current user is tested. Such low accuracy is achieved since the current user is shorter than the users whose data was used to build the *Default classifier*. More precisely, low accuracy is achieved during the classification of the basic activities, namely sitting (35.9% accuracy) and lying (81.64% accuracy). The low accuracy of sitting is caused by mistakenly classifying sitting as sitting on the ground. The accuracy of the classification for each activity can be seen in Table 3.

The first step of the adaptation consists of scaling the *Default classifier* to the current user height. This was done by scaling the attributes related to the user height using the equation (1), where the height was reduced from 176 cm to 160 cm. The scaling increased the accuracy of the *Default classifier* by six percents thus the final accuracy was 79%. By scaling previously described attributes we have increased the accuracy of the classification of almost all activities except lying and on all fours, where the accuracy has decreased. The reason is that the height of the person does not affect certain activities like falling.

	falling	lying	sitting	standing	on all fours	sitting on the ground	going down	standing up
falling	6	8	0	14	14	0	5	12
lying	0	557	0	0	40	0	0	0
sitting	2	0	328	0	0	0	5	2
standing	0	0	0	1279	0	0	6	15
on all fours	2	4	7	0	81	13	0	6
sitting on the ground	0	19	0	0	1	168	0	11
going down	8	25	2	9	2	0	38	0
standing up	2	6	5	11	10	1	0	102

Figure 4. Confusion matrix of the *Default classifier* after the active semi-supervised learning was applied.

The *Person classifier* used in this experiment is described in Section IV.A.2. We have used 30 seconds of each basic activity to build the classifier. Its accuracy is 69%. The accuracy of classification of the basic activities is seen in Table 3. This classifier and the scaled *Default classifier* were used by the active semi-supervised learning process. The test of this process was done by using three 60 minutes recordings of unlabeled data of the current user. Those data were balanced to the percentages of the average activities in daily living that are shown in Fig. 5. Each recording was used twice. During the active semi-supervised learning, each instance with 100% confidence was added to the learning set of the *Default classifier* four times. This classifier was rebuilt every five minutes. The result was increased accuracy of the *Default classifier* from 79% to 84-85%.

Accuracy of the individual activities has increased (e.g., sitting from 35.9% to 97.3%) except those activities that could be easily misclassified as lying. Even the recognition of the falling activity has increased even though it is the shortest activity to classify. The confusion matrix of the final classifier with accuracy 84.5% (Fig. 4) reveals the misclassification problem. The error is a result of scaling the attributes in lying instances. Exclusion of these instances from scaling should be done in further research.

VI. CONCLUSION

This paper describes the adaptation part of the Confidence system. The main module of Confidence is the activity recognition module that uses three classifiers to recognize current activity of the user, and all three classifiers are adjusted to each particular user. The initialization procedure includes the scaling of the activity recognition classifiers and their update using active semi-supervised learning. With the usage of the initialization procedure, the accuracy of the activity recognition module in the experiments increased from 73% to 84%. To achieve this improvement, several reasonably novel approaches were designed. Overall, the Confidence system is now in the extensive testing phase by several users and independent reviewers. The results so far, including a live demonstration at the ICT Digitally Driven 2010 Event in Brussels were better than promised in the project proposal.

In the future work, certain activities will be excluded from the scaling process, new attributes will be added to the activity recognition classifiers, e.g., the ratio of different activities in the data, history of the classification for *Default* and *Person classifier*, and more complex statistics. Besides, the aging of the data will be added to the learning data by removing the oldest data from the classifiers. Future work also includes extensive testing of the method with various end users.

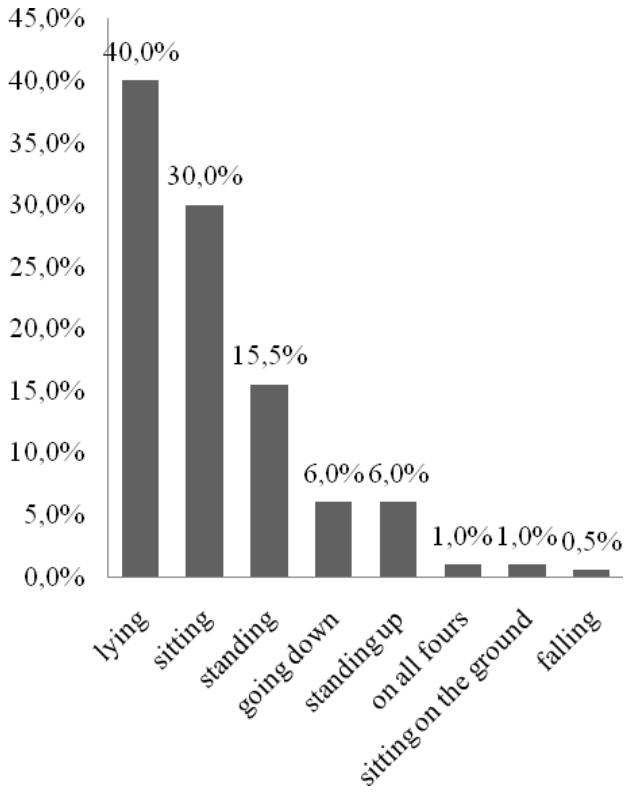

Figure 5. Percentage of activities during the day.

ACKNOWLEDGMENTS

This work was partly supported by the ARRS under the Research Programme P2-0209, partly from the EU FP7/2007–2013 under grant agreement No. 214986, and partly by the European Union, European Social Found.

REFERENCES

[1] R. Means, S. Richards, and R. Smith, Community Care: Policy and Practice. Basingstoke, Hampshire: Palgrave MacMillan, 2008.

[2] Confidence. http://www.confidence-eu.org, 2010-09-01.

[3] B. Kaluža, V. Mirchevska, E. Dovgan, M. Luštrek, and M. Gams, "An agent-based approach to care in independent living", Proc. of International Joint Conference on Ambient Intelligence 2010, in press.

[4] Ubisense. http://www.ubisense.net, 2008-09-15.

[5] Fraunhofer. http://www.fraunhofer.de, 2010-09-03.

[6] M. Lustrek and B. Kaluza, "Fall detection and activity recognition with machine learning," Informatica, vol. 33, no. 2, pp. 197–204, 2009.

[7] V. Mirchevska, M. Luštrek, and M. Gams, "Combining machine learning and expert knowledge for classifying human posture", Proc. of International Electrotechnical and Computer Science Conference 2009, pp. 183–186, 2009.

[8] L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," Proc. of the IEEE, vol. 77, no. 2, pp. 257–286, 1989.

[9] V. Mirchevska, B. Kaluža, M. Luštrek, and M. Gams, "Real-time alarm model adaptation based on user feedback", Proc. of Ubiquitous Data Mining workshop in conjunction with ECAI 2010, pp. 39–44, 2010.

[10] J. R. Quinlan, C4.5: Programs for Machine Learning. San Mateo, CA: Morgan Kaufmann Publishers, 1993.

[11] N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge: Cambridge University Press, 2000.

[12] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: identifying density-based local outliers," Proc. of the 2000 ACM SIGMOD international conference on Management of data, pp. 93–104, 2000.

[13] B. Cvetković, V. Mirčevska, E. Dovgan, B. Kaluža, M. Luštrek, and M Gams, "User manual – User screen," Technical Report IJS-DP 10559, 2010.

[14] Y. Song, X. Hua, L. Dai, and M. Wang, "Semi-automatic video annotation based on active learning with multiple complementary predictors," Proc. of the 7th ACM SIGMM international multimedia conference, pp. 97–104, 2005.