Three-layer Activity Recognition Combining Domain Knowledge and Meta-classification

Simon Kozina* Hristijan Gjoreski Matjaž Gams Mitja Luštrek

Department of Intelligent Systems, Jožef Stefan Institute, Ljubljana SI-1000, Slovenia

Received 24 Sep 2012; Accepted 18 Jan 2013; doi: 10.5405/jmbe.1321

Abstract

One of the essential tasks of healthcare and smart-living systems is to recognize the current activity of a particular user. Such activity recognition (AR) is demanding when only limited sensors are used, such as accelerometers. Given a small number of accelerometers, intelligent AR systems often use simple architectures, either general or specific for their AR. In this paper, a system for AR named TriLAR is presented. TriLAR has an AR-specific architecture consisting of three layers: (i) a bottom layer, where an arbitrary number of AR methods can be used to recognize the current activity; (ii) a middle layer, where the predictions from the bottom-layer methods are inputs for a hierarchical structure that combines domain knowledge and meta-classification; and (iii) a top layer, where a hidden Markov model is used to correct spurious transitions between the recognized activities from the middle layer. The middle layer has a hierarchical, three-level structure. First, a meta-classifier is used to make the initial separation between the most distinct activities. Second, domain knowledge in the form of rules is used to differentiate between the remaining activities, recognizing those of interest (i.e., static activities). Third, another meta-classifier deals with the remaining activities. In this way, each activity is recognized by the method best suited to it, leaving unrecognized activities to the next method. This architecture was tested on a dataset recorded using ten volunteers who acted out a complex, real-life scenario while wearing accelerometers placed on the chest, thigh, and ankle. The results show that TriLAR successfully recognized elementary activities using one or two sensors and significantly outperformed three standard, single-layer methods with all sensor placements.

Keywords: Activity recognition, Ambient intelligence, Intelligent healthcare, Machine learning, Meta-classification, Multi-layer activity recognition

1. Introduction

The world's population is aging rapidly, threatening to overwhelm society's capacity to take care of its elderly members. The percentage of persons aged 65 or over in developed countries is projected to rise from 7.5% in 2009 to 16% in 2050 [1]. This is driving the development of innovative healthcare and smart-living technologies to help the elderly live independently for longer and with minimal support from the working-age population [2,3].

To be used in a real-world setting, healthcare and smart-living systems must take into account the user's situation and context, making activity recognition (AR) an essential component of such systems [4,5]. AR requires a sensor system that observes the user and intelligent software that infers the user's activities from the sensor data [6,7].

The idea for the AR method proposed here was initiated and gradually developed in two European healthcare projects: Confidence [8] and CHIRON [9]. Even though AR was not the main goal in either of the projects, it eventually emerged as one of the most important components, being the foundation for further reasoning in the main tasks, including the detection of falls, the detection of unusual behavior, and the estimation of human energy expenditure [10-12]. The initial AR model developed in the Confidence project was a traditional, single-layer classification model that uses two types of sensor (accelerometers and location sensors) to achieve adequate performance [13]. In the CHIRON project, the AR model was upgraded to two layers, which improved performance.

The present study presents a three-layer architecture for AR called TriLAR. The TriLAR architecture consists of the following: a bottom layer (an arbitrary number of independent AR methods), a middle layer (hierarchical structure that aggregates the predictions from the bottom-layer methods), and a top layer (a hidden Markov model (HMM) that uses the temporal dependence of activities to remove spurious transitions between them). This architecture was tested on a dataset recorded using ten volunteers who acted out a complex, 90-minute scenario while wearing accelerometers placed on the

* Corresponding author: Simon Kozina
Tel: +386-1-4773195; Fax: +386-1-4773131
E-mail: simon.kozina@ijs.si

chest, thigh, and ankle. The results show that the TriLAR architecture can successfully recognize elementary activities using a limited number of sensors.

The rest of this paper is organized as follows. An overview of studies related to AR is presented in Section 2. In Section 3, the methods used in TriLAR are described. Section 4 describes the experimental setup, including the sensor equipment, the experimental data, and the methods setup. Section 5 includes the results and discussion. Finally, conclusions and directions for future work are given in Section 6.

2. Related works

AR approaches can be divided into those that use wearable and non-wearable sensors, respectively. The most common non-wearable approach is based on cameras [14]. Although this approach is physically less intrusive for the user compared to one based on wearable sensors, it suffers from problems such as low image resolution, target occlusion, and time-consuming processing. However, often the biggest issue is user privacy: the user has to accept the fact that a camera will record him or her.

The most exploited and probably the most mature approach to AR is using wearable accelerometers, which are both inexpensive and effective [15-17]. Wearable accelerometers are thus used for the TriLAR architecture. There are two common types of wearable-sensor approach for AR that have proven to be successful: those that use domain knowledge encoded with rules, and those that use machine learning (ML). Most researchers have used a single-layer architecture, i.e., implementing only one of the two approaches. However, in recent years some more advanced, multi-layer, hierarchical approaches have also been proposed [18], with some of them exploiting the temporal dependence between human activities [19,20].

The most traditional AR approach is that based on ML. This approach has a single-layer architecture and usually implements known classification methods, e.g., decision trees (DTs), support vector machines (SVM), *k*-nearest neighbor (*k*NN) algorithms, and naive Bayes (NB) classification. Examples include Kwapisz *et al*. [15], who used an accelerometer placed on the thigh and tested their single-layer approach by comparing the results of three classification methods on dynamic activities such as walking, running, and jogging. However, for the elderly, static activities are also of great importance, as these activities are the main indicators of any degradation in health (e.g., an increase in the time spent lying down). Ravi *et al*. [21] used an accelerometer on a mobile phone and tested their single-layer approach with five classification methods. The results showed that when a given person’s data was used for both training and testing, the accuracy was 90%, but when a different person’s data was used for the testing, the accuracy dropped to 65%. In the TriLAR architecture, such classification methods are used for the bottom layer and the evaluation is performed on data from different people, as the developed model is intended for use by people who were not involved in the training of the model.

Another common approach for accelerometer-based AR is based on manually created rules. These rules are usually based on features that are calculated from sensor orientations and accelerations. Wu *et al*. [16] presented an approach in which decision rules are used to recognize activities. Even though the rules are only one of several components in their approach, they showed that rules can successfully contribute to AR. Another implementation of such rules was presented by Lai *et al*. [17]. The authors used six accelerometers, placed on the neck, waist, left wrist, right wrist, left thigh, and right thigh, respectively. The reported accuracy was almost perfect (99.5%), but the number of sensors is excessive for everyday use. In the TriLAR architecture, domain rules are one of several methods used for the bottom and middle layers. The TriLAR architecture is shown to achieve adequate performance using only one or two sensors.

Hierarchical approaches implementing an ensemble of classifiers to recognize a user's activity have been a popular research topic in recent years. Banos *et al*. [18] showed that using traditional aggregation techniques such as majority vote [22,23] is not sufficient for a highly accurate AR system. They thus presented hierarchical weighted classification (HWC), which combines majority vote and weighted hierarchical aggregation. In their implementation, at the first level each sensor makes decision about the recognized activity using binary classifiers. At the next level, a weighted majority vote scheme aggregates the decision in order to make the final decision. In the approach proposed here, each bottom-layer method determines the user's activity using all the information available (from all sensors). The middle layer of the TriLAR architecture then aggregates the outputs of the bottom layer using meta-classification techniques and domain rules.

Finally, because human activities have certain natural regularities and temporal dependence (smoothness), e.g., people do not abruptly switch back and forth between lying and cycling, the history of recent activities can aid the recognition of the current activity. A common way to address this problem, and consequently reduce spurious activity transitions, is by using HMMs [24]. Lester *et al*. [20] showed that incorporating HMMs significantly improves the recognition of activities. In the TriLAR architecture, an HMM is used in the top layer, after the aggregation in the middle layer.

3. Methods

An overview of an AR system using the TriLAR architecture is shown in Fig. 1. Data from wearable sensors is preprocessed and fed to the bottom layer, and the user’s activity is returned by the top layer.

In the bottom layer of the TriLAR architecture, three AR methods are used to recognize the current activity of a user. The first method uses domain knowledge encoded with rules (rule-based AR), the second method uses trained classification models (binary classification), and the third method uses a similarity metric (distance) between the activities in order to recognize the current activity. The methods are fundamentally different, so that each has an advantage in different situations.

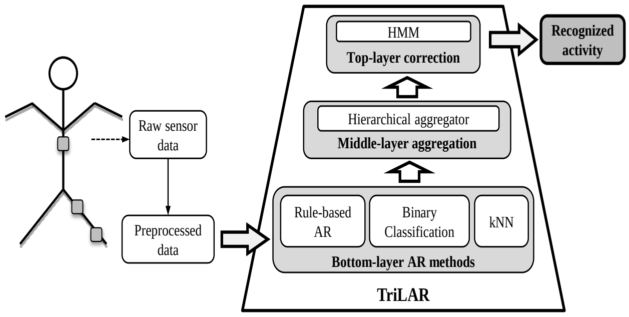


Figure 1. System overview.

In the middle layer, a hierarchical scheme aggregates the predictions from the bottom-layer classification methods and makes a joint decision about the user's activity. Because traditional aggregation techniques, such as majority vote, did not achieve adequate results, a hierarchical, three-level, aggregation structure was designed that incorporates both ML (meta-classification) and domain knowledge (rules). In the top layer, an HMM incorporates the temporal component of the human activity, and corrects the final decision about the recognized activity. Each of the layers, as well as the preprocessing, is explained in more detail in the subsections that follow.

3.1 Data preprocessing

The sensor equipment used in this study consists of three 3-axis accelerometers placed on the chest, thigh, and ankle, respectively. The acceleration in each direction is the sum of the acceleration due to gravity and the acceleration due to the movement of the sensor.

The first step in the preprocessing phase is sensor data synchronization. This is necessary when multiple sensors are used, since the data from the sensors is not all received at the same time.

Once the sensor measurements are synchronized, further preprocessing is performed using band- and low-pass filters, respectively. The band-pass filter has two goals: (1) to eliminate the low-frequency acceleration (gravity) that captures information about the orientation of the sensor with respect to the ground and (2) to eliminate the high-frequency signal components generated by non-human motion and high-frequency noise, thus preserving the medium-frequency signal components generated by dynamic human motion. The band-pass-filtered data is used for the extraction of features relevant for dynamic activities, such as walking, running, and cycling. The low-pass filter is used to eliminate most of the signals generated by dynamic human motion, preserving the low-frequency component, i.e., gravity [25,26]. The low-pass-filtered data thus contains sensor orientation information, which is relevant for the recognition of static activities (postures), such as lying, sitting, standing, and kneeling.

Finally, an overlapping sliding-window technique is applied. A window of fixed size (width) moves across the stream of data, advancing by half its length in each step. The data within each window is used in the AR described in the next section.

3.2 TriLAR architecture

3.2.1 Bottom-layer methods

In the TriLAR's bottom layer, three methods are implemented, each representing a distinct approach to AR. The first method is based on rules that recognize the posture using domain knowledge. In the second method, a set of binary classifiers is trained. Each classifier is trained to distinguish only one activity. As this increases the specificity of the classifiers, the accuracy should also increase. The last method uses *k*NNs. As the majority of datasets are annotated manually, the data can contain spurious annotations. The *k*NNs should reduce the influence of such data. In general, the bottom layer can consist of an arbitrary number of AR methods.

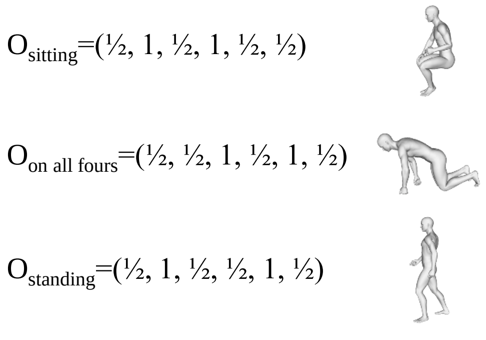
Domain-knowledge rules use the low-pass-filtered sensor orientations as the input, whereas the binary classifiers and *k*NNs use a longer feature vector as the input. The feature vector also contains low-pass-filtered features that measure the posture of the body. Additionally, it contains band-pass-filtered features that represent: (1) the motion shape, (2) the motion variance, (3) the motion energy, and (4) the motion periodicity [25]. The feature vector consists of a total of 60 features per sensor.

3.2.1.2 Rule-based activity recognition

Rule-based activity recognition (R-BAR) is used for detecting static activities, such as standing, lying, and sitting. Dynamic activities, like walking or running, are merged with their static equivalent, standing. R-BAR uses the orientation of the sensors to recognize posture. The orientation of a sensor *j* is computed with Eq. (1), where *i* is one of the axes (x, y, or z). The orientation is simultaneously normalized to the [0,1] interval.

$$\phi_{j,i} = \left(\arccos\left(\frac{a_i}{\sqrt{a_x^2 + a_y^2 + a_z^2}}\right) + 1\right) \cdot \frac{1}{2} \qquad (1)$$

The values computed in this way form an orientation vector $O = (\phi_{j,x}, \phi_{j,y}, \phi_{j,z})^{|number\ of\ sensors|}$, which is then matched with the set of rules defined by a domain expert as the typical orientations of the sensors for each activity. Figure 2 shows example orientations for three activities (sitting, on all fours, and standing) when using chest and thigh sensors. The structure of the rules in Fig. 2 is $O_{activity} = (\phi_{chest,x}, \phi_{chest,y}, \phi_{chest,z}, \phi_{thigh,x}, \phi_{thigh,y}, \phi_{thigh,z})$. For every orientation measurement in vector *O*,



$O_{sitting}$=(½, 1, ½, 1, ½, ½)

$O_{on\ all\ fours}$=(½, ½, 1, ½, 1, ½)

$O_{standing}$=(½, 1, ½, ½, 1, ½)

Figure 2. R-BAR rules for sitting, on all fours, and standing.

an error is computed using Eq. (2), where *d* is the absolute difference between the value defined in the rules and the actual measurement. A higher absolute difference *d* denotes a higher difference between the actual and the typical sensor orientation, resulting in a larger value of the error *e*.

$$e = \begin{cases} \dfrac{d^4}{0.25^3}; & 0 \le d < 0.25 \\ 3d - 0.5; & 0.25 \le d < 0.5 \\ 1 & 0.5 \le d \end{cases} \tag{2}$$

The error values form an error vector whose size is the same as that of the orientation vector. These components are summed up in order to obtain the overall error of an activity. The error values for each activity are passed to the next layer, i.e., hierarchical aggregation.

3.2.1.3 Binary classification

This method uses as many classifiers as there are activities to be recognized. An arbitrary ML method can be used to train the classifiers. In our experiments, the Random Forest method, implemented using the Weka ML tool, was empirically found to be the most successful [27]. Each classifier is trained to separate a single activity from other activities.

Each of the binary classifiers takes the feature vector described at the beginning of Section 3.2.1 as an input, and outputs the classification probability for the current activity. The probabilities of all the classifiers are merged into a vector of probabilities and are then passed to the next layer, i.e., hierarchical aggregation.

3.2.1.4 k-nearest neighbor method

The binary classifiers rely on a correct annotation of the training data. As the data annotation was performed manually during the experiments, it may contain errors. For example, the start of a new activity is often annotated either too soon or too late, since it is difficult to determine the correct ending and starting points between two activities. In order to reduce the influence of these mistakes and to improve the overall classification accuracy, an instance-based ML method, *k*NN, implemented in Weka is used [27]. *k*NN is instance-based (lazy) because it does not use the training data to do any generalization, but instead keeps all the training instances in memory.

For each instance to be classified, *k*-nearest instances from the training set are retrieved. The instances labeled with each activity are then counted, forming a frequency vector. The frequency vector is normalized so that each component represents the probability information of an instance to be classified to a given activity. The resulting probability vector is passed to the next layer.

3.2.2 Middle-layer aggregation

In the middle layer of the TriLAR architecture, the outputs from the bottom layer are aggregated and the activities are recognized. Figure 3 shows the proposed hierarchical structure for the middle-layer aggregation of the bottom-layer predictions. The aggregation uses three levels: meta-classification A, R-BAR, and meta-classification B. The lowest two either recognize the final activity or recognize it as belonging to a group and pass the decision to a higher level. The structure of this layer was designed to combine domain knowledge with ML to achieve better results. Domain knowledge incorporated into the R-BAR can distinguish between static activities (i.e., postures) and meta-classifiers A and B use the principle of multiple knowledge [28] to improve the recognition of dynamic activities.

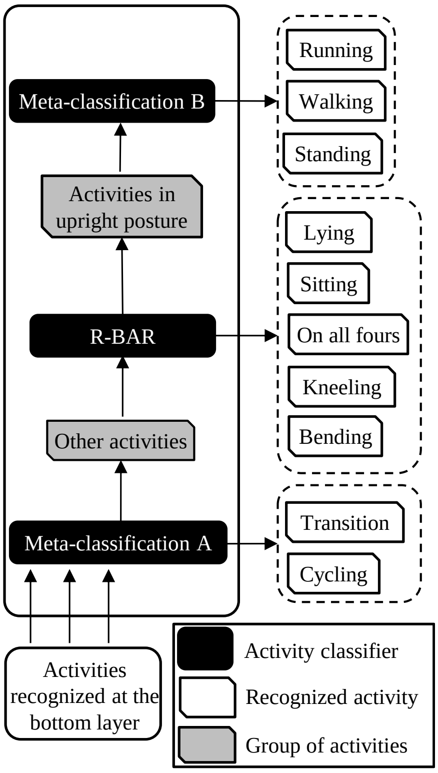


Figure 3. Middle-layer aggregation procedure.

On the first level, meta-classifier A is trained to distinguish between cycling, transitions (such as standing up or sitting down), and other activities. All other activities can be associated with distinct postures and can therefore be efficiently separated with R-BAR (cycling would sometimes be classified as sitting, bending, standing, walking, or even kneeling by R-BAR). The same applies to the transitions. A feature vector of meta-classifier A is defined as follows:

$$x_A = \left\langle \omega_{1,A} \cdot x_{kNN}^{(1)} + \omega_{2,A} \cdot x_{BC}^{(1)}, \omega_{1,A} \cdot x_{kNN}^{(2)} + \omega_{2,A} \cdot x_{BC}^{(2)}, \omega_{1,A} \cdot x_{kNN}^{(3)} + \omega_{2,A} \cdot x_{BC}^{(3)} \right\rangle \tag{3}$$

where $\omega_{1,A}$ and $\omega_{2,A}$ ($\omega_{1,A} + \omega_{2,A} = 1$) are the confidence factors in the binary and *k*NN classifiers' outputs, determined experimentally, $x_{kNN}^{(i)}$ is the *k*NN's probability for the i-th activity (cycling, transition, other activities) and $x_{BC}^{(i)}$ is the binary classifier's probability for the i-th activity. Meta-classifier A is trained using the Random Forest algorithm [29]. If a feature vector x_A is classified as cycling or transition, the aggregation is completed. If it is classified as other activities, the second level of the aggregation is applied to the bottom-layer outputs.

On the second level, the R-BAR is used to distinguish between body postures using the domain knowledge. An empirical analysis of the rules showed that they are the most suitable for distinguishing static activities (postures). The procedure described in Section 3.2.1.2 is applied to instances that were classified as other activities on the first level of the aggregation structure. Six postures are recognized on this level: lying, kneeling, sitting, bending, on all fours, and upright posture. In the case of the upright posture, the third level of the aggregation procedure is used to further distinguish between the activities associated with this position; otherwise, the aggregation procedure is completed.

On the last level, the instances that were classified as an upright posture are further classified by meta-classifier B into: standing, walking, and running. Like with the bottom level, the components of feature vector x_B are weighted sums of the probabilities returned by the *k*NN and binary classifiers for the aforementioned three activities. The main difference is that the weights ($\omega_{1,B}$ and $\omega_{2,B}$) are not necessarily equal to the ones on the first level. The classifier used is the Random Forest algorithm.

When the aggregation is completed on the first, second, or third level, the recognized activity is passed to the top layer of TriLAR for the removal of spurious transitions.

3.2.3 Top-layer correction

The sequence of the activities' output by the hierarchical aggregation is sometimes characterized by spurious transitions between different activities. This problem is usually caused by the previous two layers of the AR architecture failing to take into account the continuity of the human activity. They classify each data sample in isolation and assume that there is no connection with the previous and following data samples.

A common way to address this problem, and consequently to reduce spurious activity transitions, is to model the temporal dependence of the activities using HMMs [24]. An HMM observes the Markov property that the current system state is only dependent on the previous state of the system. The model consists of a number of hidden states and the associated transition probabilities between these hidden states. The hidden states emit events with certain emission probabilities, and these events are observed by an outside observer [30]. Our hidden states are the unknown true activities. The observed states are the activities aggregated by the middle layer. The Baum-Welch algorithm [31] is used to find the transition and emission probabilities of the HMM, and the Viterbi algorithm [32] is used to generate the most likely sequence of hidden states given an observation sequence of events. The output of the method was used to correct the aggregated middle-layer prediction and output the final decision about the user's activity.

4. Experimental setup

4.1 Sensor equipment

The sensor equipment used in this study consists of three accelerometers placed on the chest, thigh, and ankle, respectively. These placements were chosen as a trade-off between the intrusiveness with respect to the user and the achieved AR performance from preliminary tests [13]. For each sensor placement, a custom-made body strap was used. These straps were made of elastic material with Velcro at the ends, which meant they could be adapted to different types of user.

After analyzing the various available commercial accelerometers, the Shimmer sensor platform [33] was chosen. It has a reasonable battery life and compact size, is completely wireless, and has the option to reprogram the sensor based on the user's needs and situation. The chosen platform has a 3-axis accelerometer, uses Bluetooth communication, and has 2 GB of storage, which is enough to store 3 months of sensor data when the frequency of data acquisition is 50 Hz. This frequency was also used to record all the experimental data described in the next section.

4.2 Experimental data

A complex, 90-minute, test scenario was designed in cooperation with a medical expert to capture the real-life conditions of a person's behavior, although it was recorded in a laboratory. The scenario was acted out by ten volunteers. They were asked to attach the sensors to their ankle, thigh, and chest, respectively, following the instructions from the expert.

The scenario was divided into three groups, each containing several sub-scenarios. The first group was exercise activities. The following three sub-scenarios were recorded: walking on a treadmill with a one-percent inclination at 4 km/h and 6 km/h, running on a treadmill with a one-percent inclination at 8 km/h, and cycling on a stationary bicycle at 65 RPM with the difficulty set to 80 W for the first six minutes and 160 W for the remaining six minutes. In the second group, elementary activities and transitions between the activities were recorded. The sequence of activities performed in these sub-scenarios was predefined and volunteers were asked to follow them. In the third group, everyday-life activities were recorded. The sequence of activities was not predefined and the volunteers were asked to mimic their normal, everyday-life behavior when executing activities such as cooking, reading, typing, washing dishes, and scrubbing the floor.

Altogether, ten sub-scenarios were recorded, resulting in 140 recordings, as some sub-scenarios were repeated multiple times, yielding a total of approximately 1,000,000 raw data samples per volunteer. These raw data samples were transformed into approximately 7,000 feature vectors per volunteer. The scenario included ten elementary activities (percentage of instances per class given in brackets): standing (16%), sitting (11%), lying (22%), on all fours (10%), kneeling (6%), bending (standing leaning) (3%), walking (15%), running (5%), cycling (10%), and transition (going down and standing up) (2%). These activities were selected because they are the most common elementary, everyday-life activities. A more detailed breakdown is presented in Table 1.

4.3 Methods setup

The preliminary tests [34] showed that a 2-second window size for the sliding window is a reasonable trade-off between the duration of the activities and the recognition delay. In some cases, longer windows yielded higher recognition

Table 1. Activity scenario.

Group of activity scenarios (percentage of instances per group)	Recorded activities (percentage of instances per class per group)
Exercise (25%)	walking (10%), running (5%), cycling (10%)
Elementary activities and transitions between them (50%)	lying (17%), sitting (8%), standing (8%), bending (3%), kneeling (6%), on all fours (2%), transition (2%), walking (4%)
Everyday-life activities (25%) (e.g. reading, typing, cooking, washing dishes, cleaning)	lying (5%), sitting (5%), standing (8%), on all fours (4%), kneeling (1%), walking (2%)

accuracies, but some short activities were missed if the length was more than 2 seconds.

The parameters used for the SVM (implemented as SMO in Weka), DTs (implemented as J48 in Weka), NB, and Random Forest were all default ones, as described in Weka's API [35]. The *k*NN (implemented as LinearNNSearch in Weka) parameters were set to default, except for the *k* value, which was set to 81, as proposed by Maier *et al.* [36]. Two other values of *k* ($k = 1$, $k = 201$) were also tested, but there was no significant difference in the final accuracy. For the middle-layer aggregation, four confidence factors were used. These factors were determined experimentally and set to $\omega_{1,A} = \omega_{1,B} = 0.25$ and $\omega_{2,A} = \omega_{2,B} = 0.75$.

For the top layer, the implementation of the HMM consisted of two phases. The Baum-Welch method was parameterized by the following: $N = 10$ internal hidden states (equal to the number of activities due to the Viterbi assumption), the initial state transition probability $a_{ij} = 1/10$; the initial state probability $p_i = 1/10$, and the output symbol distribution in state $b_j(k) = 1$ if $k = j$, otherwise $b_j(k) = 0$. Learning was performed with the Jahmm implementation [37] of the Baum-Welch method with 70 iterations on the training data. The size of the observation-activities sequence used by the Viterbi algorithm was experimentally set to 200.

5. Results and discussion

To test the proposed TriLAR architecture, the dataset described in Subsection 4.2 was used. The evaluation technique for the ML methods, the ones that require training a model, was leave-one-person-out cross validation. This technique constructs the training model on the data from all the people except one. The remaining person is used to evaluate the accuracy of the trained model. This procedure was repeated for each person (10 times) and the average performance was measured. This evaluation procedure was used because using a given person's data for both training and testing would give overly optimistic results if the intended use of the model is to classify the activities of previously unseen people.

Table 2 shows the AR accuracy and standard deviation achieved by the TriLAR architecture for each sensor placement. In order to show the improvements due to each layer of the TriLAR architecture, the accuracies and standard deviations achieved by the methods in each layer are presented, i.e., the bottom layer (R-BAR, *k*NN, binary classification); the middle layer (hierarchical aggregation); and the top layer (HMM correction). The results show that for each sensor placement, the middle and the top layers improve the basic accuracy of each of the bottom-layer methods. Additionally, the sensor placement analysis shows that the ankle and thigh sensor placements are the best performing for one-sensor AR systems in which each activity is equally important, achieving accuracies of 85.72% and 85.96%, respectively. Chest and ankle sensors are the best-performing sensor combination for two-sensor AR systems, improving over the best single-sensor placement by 8.1 percentage points (p.p.). The three-sensor placement (chest, ankle, thigh) achieves a 98.03% accuracy, which is 3.96 p.p. better than that of the best two-sensor placement. However, by increasing the number of sensors, the intrusiveness with respect to the user increases as well. Therefore, the two-sensor AR system is a reasonable trade-off between the number of sensors and AR performance. In addition, the performance of the TriLAR architecture was compared to three commonly used single-layer classification methods: SVM, DTs, and NB. The comparison between the TriLAR final accuracy and the single-layer methods shows that the TriLAR architecture outperforms the single-layer methods for any sensor placement. For instance, the TriLAR accuracy

Table 2. Classification accuracy (in percent) for all methods in TriLAR and for each sensor placement.

	TriLAR							
	Bottom layer			Middle layer	Top layer	Single-layer methods		
Sensor placement	R-BAR	kNN	Binary	Aggregation	HMM - Final	SVM	DT	NB
Ankle	76.65 ± 10.77	81.14 ± 4.24	83.73 ± 4.97	84.90 ± 2.68	85.72 ± 2.71	82.87 ± 4.47	78.76 ± 8.05	76.86 ± 3.69
Chest	70.42 ± 27.66	64.40 ± 8.40	73.14 ± 8.43	76.56 ± 6.67	79.43 ± 6.94	67.15 ± 9.10	68.40 ± 7.34	64.16 ± 11.09
Thigh	82.10 ± 9.45	80.59 ± 8.66	83.95 ± 11.02	84.44 ± 10.80	85.96 ± 10.10	81.73 ± 10.74	79.75 ± 9.23	78.88 ± 8.54
Chest and ankle	88.86 ± 5.98	83.20 ± 6.75	92.49 ± 2.93	92.96 ± 2.84	94.06 ± 2.51	90.93 ± 3.92	88.76 ± 3.93	88.64 ± 3.75
Chest and thigh	90.92 ± 9.40	77.08 ± 8.26	91.06 ± 3.78	91.93 ± 4.07	92.79 ± 3.74	88.41 ± 4.82	90.04 ± 3.68	87.41 ± 6.81
Thigh and ankle	92.22 ± 6.84	85.30 ± 6.56	91.85 ± 4.29	92.93 ± 3.77	93.67 ± 3.30	91.84 ± 3.52	89.11 ± 3.59	86.96 ± 3.97
Chest, thigh, and ankle	97.26 ± 1.29	89.29 ± 4.33	97.05 ± 0.58	97.69 ± 0.70	98.03 ± 0.62	95.05 ± 2.26	95.04 ± 1.79	93.52 ± 2.48

achieved by the best-performing two- sensor placement (chest and ankle) is 94.06%, which is 4.2 p.p. better than that of the best-performing single-layer method, i.e., SVM. In total, the TriLAR architecture outperformed the single-layer classifiers by an average of 4 p.p. In addition, tests of the statistical significance were performed. Because of the small number of folds (10) and because the individual samples (folds) are paired (the same person's data for each placement), the paired Student's t-test with a significance level of 5% was used. The results show that the TriLAR performance is statistically significantly better than each of the single-layer methods.

Classification accuracy can sometimes be misleading as it averages over all the activities. This is a problem, especially when the test set is not balanced, i.e., some activities are represented by a small number of examples [38]. Therefore, in Table 3, the TriLAR recall, precision, and F-measure (F1 score) for each activity and each sensor placement are presented. Because the F-measure combines the precision and the recall (harmonic mean), it is used as a reference metric for easier discussion and explanation of the results. Additionally, a set of confusion matrices is shown in Fig. 4. Each confusion matrix is presented with a density plot: the darker the color, the higher the accuracy. Actual activities are placed horizontally and the predicted ones are placed vertically (the activity numbers correspond to the numbers in Table 3). The correct predictions are shown on the diagonal of each matrix. All the other positions denote misclassified examples.

The results show that the ankle and thigh sensor placements, compared to the chest one, are better for almost any activity. The difference is most evident in the kneeling, sitting, and cycling activities. The rationale behind this is that the chest sensor has similar orientations and accelerations during kneeling, sitting, and standing and during walking and cycling on a stationary bike. The confusion matrix [TriLAR × C] in Fig. 4 representing the chest sensor confirms this explanation and shows the mutual misclassification between these activities (activities no. 6, 2, and 3; and 8 and 10). For the same reasons, the ankle and thigh sensors mutually misclassify: kneeling and on all fours; and standing and bending (shown in [TriLAR × A] and [TriLAR × T] matrices in Fig. 4, activities no. 6 and 5; and 3 and 4). For two-sensor placements, the difference in the accuracies is minimal (1.2 p.p.). The best performing is the chest and ankle placement, which achieves lower accuracies only for the sitting and standing activities. The reason for this is the similarity in the sensor orientations during these two activities. This is also confirmed by the [TriLAR × A + T] confusion matrix shown in Fig. 4, activities no. 2 and 3. The chest and thigh placement is better in recognizing sitting, but has a lower accuracy for kneeling and standing activities (shown in [TriLAR × T + C] matrix, activities no. 6 and 3). The reasons are the same as above, i.e., the similarity in sensor orientations. The thigh and ankle placement has a lower accuracy in kneeling and all fours for the same reasons (shown in [TriLAR × A + T] matrix, activities no. 6 and 5). To summarize, for two-sensor

Table 3. TriLAR classification recall or true-positive rate (in percent) for all activities for each sensor placement.

			Sensor body placements						
		Evaluation metric	Ankle	Chest	Thigh	Chest and ankle	Chest and thigh	Thigh and ankle	Chest, thigh, and ankle
Static activities	1. Lying	Recall	94.6	97.4	92.9	98.3	99.4	99.6	99.7
		Precision	96.1	96.0	96.1	98.9	98.6	99.0	99.1
		F-measure	95.3	96.7	94.5	98.6	99.0	99.3	99.4
	2. Sitting	Recall	80.8	40.7	81.2	77.8	97.5	97.6	98.5
		Precision	82.6	57.6	73.7	83.1	90.9	96.0	97.7
		F-measure	81.7	47.7	77.3	80.4	94.1	96.8	98.1
	3. Standing	Recall	78.7	73.6	80.3	86.0	87.4	90.1	97.3
		Precision	75.3	57.0	72.6	82.2	80.7	90.0	97.4
		F-measure	77.0	64.2	76.3	84.1	83.9	90.0	97.3
	4. Bending	Recall	56.3	59.5	66.2	93.2	84.8	75.3	96.0
		Precision	60.4	61.0	57.4	95.5	83.1	73.6	97.2
		F-measure	58.3	60.2	61.5	94.3	83.9	74.4	96.6
	5. All fours	Recall	53.5	80.0	71.5	98.7	93.0	78.7	99.1
		Precision	58.6	76.3	74.7	94.7	94.9	81.6	98.7
		F-measure	55.9	78.1	73.1	96.7	93.9	80.1	98.9
	6. Kneeling	Recall	63.2	31.2	46.1	98.8	63.0	84.5	98.8
		Precision	54.1	36.7	54.5	99.1	79.8	81.0	99.1
		F-measure	58.3	33.7	49.9	98.9	70.4	82.7	98.9
Dynamic activities	7. Transition	Recall	66.4	77.8	65.5	75.2	69.2	68.2	80.6
		Precision	72.8	77.2	70.7	77.1	72.9	75.5	84.7
		F-measure	69.5	77.5	68.0	76.1	71.0	71.7	82.6
	8. Walking	Recall	97.2	95.2	97.4	97.3	97.3	97.7	98.3
		Precision	96.2	93.4	95.8	96.3	96.3	97.1	97.4
		F-measure	96.7	94.3	96.6	96.8	96.8	97.4	97.8
	9. Running	Recall	96.8	97.3	97.3	97.1	97.1	97.1	97.6
		Precision	98.9	98.9	99.9	98.6	99.6	99.8	99.8
		F-measure	97.8	98.1	98.6	97.8	98.3	98.4	98.7
	10. Cycling	Recall	99.8	83.9	99.1	99.8	99.7	99.8	99.9
		Precision	99.9	86.4	99.7	100.0	99.7	99.9	100.0
		F-measure	99.8	85.1	99.4	99.9	99.7	99.8	99.9

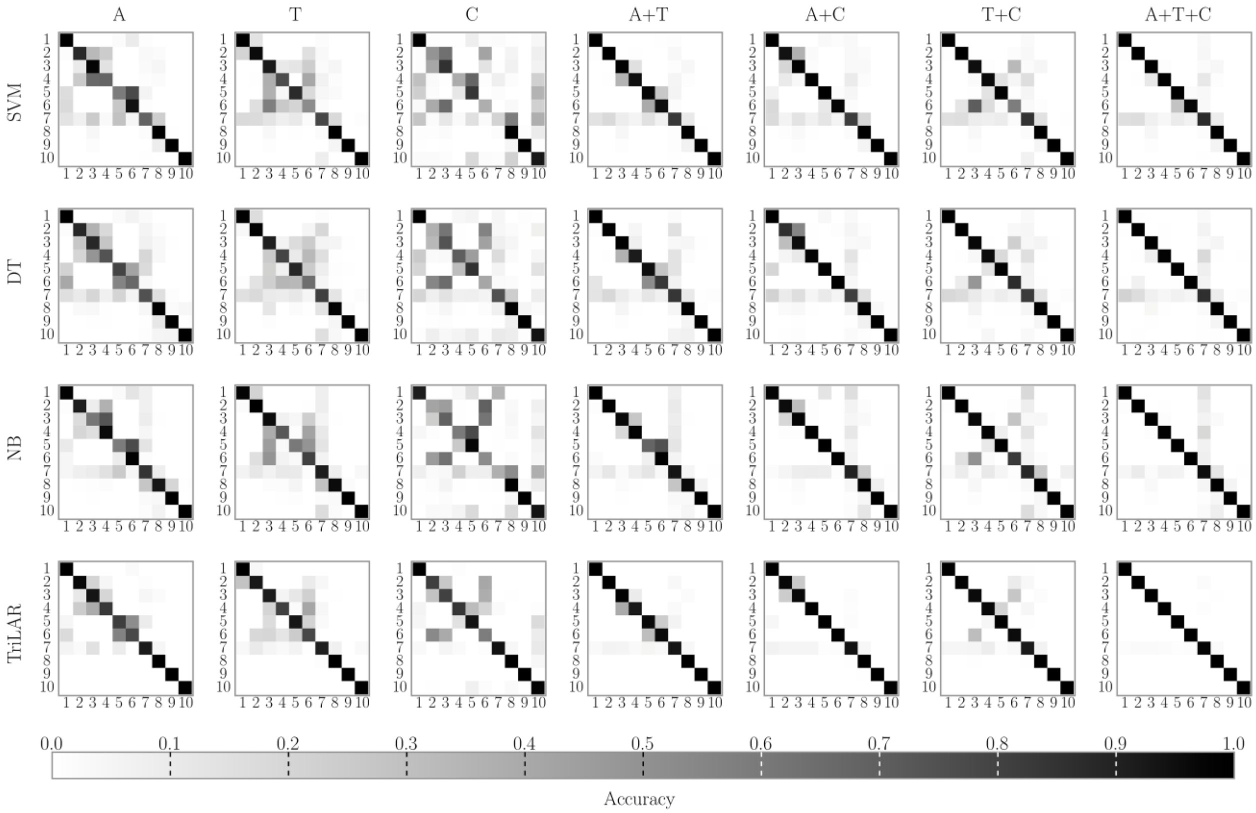


Figure 4. Confusion matrices for each sensor placement and ML algorithm. *Activity legend:* 1-ying, 2-sitting, 3-standing, 4-bending, 5-on all fours, 6-kneeling, 7-transition, 8-walking, 9-running, 10-cycling. *Sensor legend*: A-ankle, T-thigh, C-chest.

placements, the results show that no sensor placement clearly outperforms the others. Therefore, given the analysis in Table 3 and Fig. 4, the most appropriate sensor placement can be selected for each use case with respect to the recognition accuracy of the individual activities and the user's comfort.

The three-sensor placement performance for different activities shown in Table 3 and Fig. 4 ([TriLAR × A + T + C] matrix) shows that three sensors are enough to develop an almost 100% accurate AR system.

6. Conclusion

This study presented an architecture for AR called TriLAR. It has a three-layer design: a bottom layer (arbitrary number of independent AR classifiers), a middle layer (meta-classification and domain knowledge to aggregate the predictions from the bottom-layer components), and a top layer (hidden Markov model that uses the temporal dependence of the activities to remove spurious transitions between them). The architecture incorporates methods proven to be successful for AR and combines them in a way that allows each component to perform the task most suitable for it.

The TriLAR architecture is a general AR concept that can be used with different activities and data acquired from different types of sensor. It can be used with other wearable sensors (gyroscopes, magnetometers, location sensors) and also possibly with non-wearable sensors (e.g., pressure sensors, sound, and cameras). To do so, the rule-based method would require the most changes, and the ML-based bottom-layer methods (*k*NN and binary classifiers) would require new features. Meta classifiers in the middle layer and the top layer could remain unchanged. Furthermore, the bottom layer of the TriLAR architecture can be extended or even completely replaced with an arbitrary number of AR methods.

The performance of the TriLAR architecture with all possible combinations of three accelerometer placements was tested using a complex, 90-minute scenario. The results show that the accuracy for the three-layer architecture improves with each layer for all the sensor placements. Additionally, the TriLAR was shown to significantly outperform three commonly used, single-layer methods, namely DTs, NB, and SVM. Finally, it was shown that by using an advanced layered architecture, such as TriLAR, it is possible to successfully recognize elementary, everyday-life activities using a small number of sensors, i.e., one or two. The best practical solution may be the ankle and chest or thigh and chest sensor placement, but the most appropriate placement should be evaluated for each individual case.

Acknowledgements

This work was partly supported by the Slovenian Research Agency under the Research Programme P2-0209 Artificial Intelligence and Intelligent Systems and partly by the Confidence project-the European Community's Framework Programme FP7/2007-2013 under grant agreement No. 214986, and partly by the CHIRON project-ARTEMIS Joint Undertaking, under grant agreement No. 2009-1-100228. The authors would like to thank Boštjan Kaluža for his helpful suggestions and discussions.

References

[1] United Nations 2009, World population ageing, Report.

[2] A. Bourouis, M. Feham and A. Bouchachia, "A new architecture of a ubiquitous health monitoring system: a prototype of cloud mobile health monitoring system," *The Comput. Res. Repository*, 2012.

[3] M. Lustrek, B. Kaluza, B. Cvetkovic, E. Dovgan, H. Gjoreski, V. Mirchevska and M. Gams, "Confidence: ubiquitous care system to support independent living," *Eur. Conf. Artif. Intell. (demo track)*, 1013-1014, 2012.

[4] D. A. Gregory, K. D. Anind, J. B. Peter, D. Nigel, S. Mark and S. Pete, "Towards a better understanding of context and context-awareness," *Proc. 1st Int. Symp. Handheld Ubiquitous Comput.*, 304-307, 1999.

[5] N. Vyas, J. Farringdon, D. Andre and J. I. Stivoric, "Machine learning and sensor fusion for estimating continuous energy expenditure," *Proc. Innovative Appl. Artif. Intell. Conf.*, 1613-1620, 2012.

[6] H. Gjoreski, M. Luštrek and M. Gams, "Accelerometer placement for posture recognition and fall detection," *Proc. Intell. Environ. Conf.*, 47-54, 2011.

[7] PAMSys-The physical activity monitoring system, 2012. Available: http://www.biosensics.com/pamsys.html

[8] The European FP7 Confidence project, 2012. Available: http:// www.confidence-eu.org

[9] CHIRON project (Cyclic and person-centric health management: Integrated approach for home, mobile and clinical environments), 2012. Available: http:// www.chiron-project.eu/

[10] H. Gjoreski, M. Luštrek and M. Gams, "Context-based fall detection using inertial and location sensors," *Proc. Int. Joint Conf. Ambient Intell.*, 1-16, 2012.

[11] E. Dovgan, M. Luštrek, B. Pogorelc, A. Gradišek, H. Burger and M. Gams, "Intelligent elderly-care prototype for fall and disease detection," *Zdravstveni Vestnik*, 80: 824-831, 2011.

[12] B. Kaluža and M. Gams, "Analysis of daily-living dynamics," *J. Ambient Intell. Smart Environ.*, 4: 403-413, 2012.

[13] H. Gjoreski, "Adaptive human activity recognition and fall detection using wearable sensors," *Msc Thesis, Jozef Stefan Int. Postgraduate School*, 2011.

[14] G. Sukthankar and K. Sycara, "A cost minimization approach to human behavior recognition," *Proc. Int. Conf. Autonom. Agents*, 1067-1074, 2005.

[15] J. R. Kwapisz, G. M. Weiss and S. A. Moore, "Activity recognition using cell phone accelerometers," *SIGKDD Explor. Newsl.*, 12: 74-82, 2010.

[16] H. Wu, E. D. Lemaire and N. Baddour, "Activity change-of-state identification using a Blackberry smartphone," *J. Med. Biol. Eng.*, 32: 265-272, 2012.

[17] C. Lai, Y. M. Huang, J. H. Park and H. C. Chao, "Adaptive body posture analysis for elderly-falling detection with multisensors," *IEEE Intell. Syst.*, 25: 20-30, 2010.

[18] O. Banos, M. Damas, H. Pomares, F. Rojas, B. Delgado-Marquez and O. Valenzuela, "Human activity recognition based on a sensor weighting hierarchical classifier," *Soft Comput.*, 17: 333-343, 2013.

[19] J. Lester, T. Choudhury and G. Borriello, "A practical approach to recognizing physical activities," *Lecture Notes in Comput. Sci.*, 3968: 1-16, 2006.

[20] J. Lester, T. Choudhury, N. Kern, G. Borriello and B. Hannaford, "A hybrid discriminative/generative approach for modeling human activities," *Proc. Inter. Joint Conf. Artif. Intell.*, 766-772, 2005.

[21] N. Ravi, N. Dandekar, P. Mysore and M. L. Littman, "Activity recognition from accelerometer data," *Proc. 17th Conf. Innovative Appl. Artif. Intell.*, 3: 1541-1546, 2005.

[22] P. Zappi, T. Stiefmeier, E. Farella, D. Roggen, L. Benini and G. Troster, "Activity recognition from on-body sensors by classifier fusion: sensor scalability and robustness," *Proc. Int. Conf. Intell Sens., Sens. Networks Inf.*, 281-286, 2007.

[23] T. G. Dietterich, "Ensemble methods in machine learning," *Proc. 1st Int. Workshop Multiple Classifier Syst.*, 1-15, 2000.

[24] B. Kaluža, "Reducing spurious activity transitions in a sequence of movement," *Proc. 18th Int. Electrotechnical Comput. Sci. Conf.*, 163-166, 2009.

[25] E. M. Tapia, "Using machine learning for real-time activity recognition and estimation of energy expenditure," *Ph.D. Thesis, Massachusetts Institute of Technology*, 2008.

[26] J. Hausmann and K. Wac, "Activity level estimator on a commercial mobile phone: a feasibility study," *Proc. Int. Workshop Front. Act. Recognition using Pervasive Sens.*, 42-47, 2011.

[27] I. Witten and E. Frank, *Data mining: practical machine learning tools and techniques (Third edition)*, San Francisco: Morgan Kaufmann, 2011.

[28] M. Gams, *Weak intelligence: through the principle and paradox of multiple knowledge*, New York: Nova Science Publishers, 2001.

[29] L. Breiman, "Random forests," *Mach. Learn.*, 45: 5-32, 2001.

[30] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, 77: 257-286, 1989.

[31] L. E. Baum, T. Petrie, G. Soules and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Stat.*, 41: 164-171, 1970.

[32] J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, 13: 260-269, 1967.

[33] Shimmer sensor platform. Available: http://www.shimmer -research.com/

[34] M. Žbogar, H. Gjoreski, S. Kozina and M. Luštrek, "Improving accelerometer based activity recognition," *Proc. 15th Int. Multiconf. Inf. Soc.*, 167-170, 2012.

[35] Weka application programming interface (API), 2012. Available: http://weka.sourceforge.net/doc/

[36] M. Maier, M. Hein and U. von Luxburg, "Optimal construction of k-nearest neighbor graphs for identifying noisy clusters," *Theor. Comput. Sci.*, 410: 1749-1764, 2009.

[37] Jahmm (Java implementation of HMM related algorithms), 2012. Available: http://www.run.montefiore.ulg.ac.be/~francois/ software/jahmm

[38] T. L. M. van Kasteren, H. Alemdar and C. Ersoy, "Effective performance metrics for evaluating activity recognition methods," *Proc. Int. Conf. Archit. Comput. Syst.*, 301-310, 2011.