Human Activity Recognition in Intelligent Home Environments: An Evolving Approach

Jose Antonio Iglesias [1] and **Plamen Angelov** [2] and **Agapito Ledezma** [3] and **Araceli Sanchis** [4]

Abstract. In this paper, we propose an automated approach to track and recognize daily activities. Any activity is represented in this research as a sequence of raw sensors data. These sequences are treated using statistical methods in order to discover activity patterns. However, as the way to perform an activity is usually not fixed but it changes and evolves, we propose an activity recognition method based on *Evolving Systems*.

1 Introduction

In recent years, significant work has been carried out for recognizing human activities; however, most of the models created for an activity do not change according to the moods and habits of the individual who performs that activity. In this research, we propose an adaptive approach for creating models of Activities of Daily Living (ADLs) and recognizing them from the sensor readings collected in an intelligent environment.

The creation of the model of an ADL from a sequence of sensor readings should consider the sequentiality of the readings and the influence of past experiences. However, it is difficult or in general impossible, to build a classifier that will have a full description of all possible ways to perform an activity because they are not static and new patterns may emerge as well as an old habit may be forgotten or stopped to be used. Thus, as the description of the performance of a particular activity itself may also evolve, we assume that each ADL is described by one or more fuzzy rules. A conventional system would not capture the new patterns that could appear in the sequence of sensor readings once the classifier is built. In addition, the information of the sensors readings collected from an intelligent home environment is quite large. For this reason, we need to cope with large amounts of data and process this information in real time because analyzing it in an offline mode would be impractical. In order to take into account these aspects, we propose an *Evolving Systems*-based approach that allows to create dynamic and evolving models of ADLs.

2 Our Approach

The input of the proposed method is a sequence of sensor readings with which our approach, as many other agent modeling methods, creates a library (the *Evolving ADL Library*, EALib) which contains the different expected models. However, the library we propose is not a pre-fixed one, but it is evolving, learning from the *new* sequences collected from the intelligent environment.

The proposed approach includes at each step the following two main actions: **1. Acquisition of the model of an ADL:** The sequences of sensor readings are analyzed and the corresponding models are created. **2. Evolving the Classifier and Sequence Classification:** This sub-action includes on-line learning and update of the classifier, changing the EALib. In addition, a sequence of sensor readings is classified into one of the ADLs previously analyzed.

2.1 Acquisition of the model of an ADL

In this step, a sequence of sensor readings collected while the ADL is done by a human, is treated considering that the human actions are usually influenced by past experiences. This aspect motivates the idea of automated sequence learning for activity classification. Thus, in order to get the most representative and relevant set of sensor readings subsequences from a sequence, we propose the use of a *trie* data structure [4]. This structure was also used (and explained) in [6].

The subsequences of sensor readings are stored in a *trie* in a way that all possible subsequences are accessible and explicitly represented. This *trie* describes a specific ADL and a node of the *trie* represents a sensor reading, and its *children* represent the sensor readings that follow it. Also, each node keeps track of the number of times a sensor reading has been inserted into it. As the dependencies of the sensor readings are relevant in the model of the ADL, the subsequence suffixes are also inserted.

Once the *trie* is created, the subsequences that characterize the ADL and its relevance are calculated by traversing the *trie*. For this purpose, frequency-based methods are used. In particular, the relevance of a subsequence is evaluated by its relative frequency or support [1]. In this case, the support of a subsequence is defined as the ratio of the number of times the subsequence has been inserted into the *trie* to the total number of subsequences of equal size inserted. After creating the model of an ADL, it is classified and used to update EALib.

2.2 Evolving the Classifier and Sequence Classification

This subsection describes how the model of an ADL is represented by the proposed classifier, called EvCDSsr - *Evolving Classifier based on Distributions of Subsequences of Sensor Readings*, and how this classifier is created in an evolving manner.

First, EvCDSsr converts the sequence of sensor readings into the corresponding distribution of subsequences on-line (as explained in the previous section). In order to classify an ADL, these distributions must be represented in a data space. For this reason, each distribution is considered as a data vector that defines a *point* that can be represented in the data space.

[1] Carlos III University, Madrid, Spain, email: jiglesia@inf.uc3m.es
[2] InfoLab21, Lancaster University, Lancaster, United Kingdom, email: p.angelov@lancaster.ac.uk
[3] Carlos III University, Madrid, Spain, email: ledezma@inf.uc3m.es
[4] Carlos III University, Madrid, Spain, email: masm@inf.uc3m.es

The data space in which these *points* can be represented should consist of n dimensions, where n is the number of the different subsequences obtained. For this reason, in EvCDSsr, the dimension of the data space is incrementally growing according to the different subsequences that are represented in it.

2.3 Structure of the classifier EvCDSsr

Once the corresponding data vector, which represent the distribution of a specific ADL, has been created from the sequence of sensor readings, it is processed by the classifier. The structure of this classifier includes:

1. **Classify the new sample** in a group represented by a prototype.
2. **Calculate the *potential* of the new data sample** to be a prototype.
3. **Update all the prototypes** considering the new data sample.
4. **Insert the new data sample as a new prototype** if needed.
5. **Remove existing prototypes** if needed.

Each step of this evolving classification method is based in [2] and [5]. Initially, to classify a new data sample, we compare it with all the prototypes stored in EALib. This comparison is done using cosine distance and the smallest distance determines the closest similarity.

As in [2], a prototype is a data sample (the model of an ADL represented by a distribution of subsequences of sensor readings) that groups several samples which represent a certain way to perform an ADL. The classifier is initialized with the first data sample, which is stored in the EALib. Then, each data sample is classified into one of the prototypes defined in the classifier. Finally, based on the *potential* of the new data sample to become a prototype, it could form a new prototype or replace an existing one.

In [3] the potential is calculated using the euclidean distance and in [2] it is calculated using the cosine distance. Cosine distance has the advantage that it tolerates different samples to have different number of attributes (in this case, an attribute is the support value of a subsequence of sensor readings). Therefore, EvCDSsr uses the cosine distance to measure the similarity between two samples.

Once the potential of the new data sample has been calculated, all the existing prototypes in the EALib are updated considering this new data sample. It is done because the *density* of the data space surrounding certain data sample changes with the insertion of each new data sample. This operation is done really fast and it requires very little memory space because of the use of recursive equations.

The proposed evolving classifier, EvCDSsr, does not require pretraning and it can start '*from scratch*' (without prototypes in the library) in a similar manner as *eClass* evolving fuzzy rule-based classifier proposed in [3]. The potential of each new data sample is calculated and the potential of the other prototypes is updated. However, it is remarkable that the potential is calculated *recursively*, so there is no need to store all the samples in the library.

After adding a new prototype, we check whether any of the already existing prototypes in the EALib are described *well* by the newly added prototype [2]. By *well* we mean that the value of the membership function that describes the closeness to the prototype is a Gaussian bell function chosen due to its generalization capabilities.

EvCDSsr faces an important challenge in the human activity recognition: to evolve the created classifier according to the new sequences of sensor readings collected in the intelligent environment.

3 Experimental Setup and Results

In order to evaluate EvCDSsr, we use a dataset with the sensor readings activated by a person while s/he is doing a specific ADL. Thus, each sequence of sensor readings is labeled. The dataset used in this research (created by the *CASAS Smart Home project* [7]) represents sensor readings collected in a smart apartment testbed. The data represents 24 participants performing the following five ADLs in the apartment: *Make a phone call, Wash hands, Cook, Eat* and *Clean*.

For evaluating the performance of EvCDSsr, we compare it with incremental and non-incremental classifiers: Naive Bayes, k-Nearest Neighbor (incremental and non-incremental), and C5.0. In general, the results show that the proposed classifier works well in this kind of environments when the subsequence length is around 3 (its classification rate is 94,2 %). However, EvCDSsr is suitable in the proposed environment because it does not need to store the entire data stream in the memory and disregards any sample after being used. In addition, EvCDSsr is computationally simple and efficient as it is recursive and one-pass (each sample is proceeded once at the time of its arrival). In fact, because the number of attributes is very large in a real environment, the proposed approach EvCDSsr is the best working alternative.

4 Conclusions

In this paper we propose a generic approach (EvCDSsr) to model and classify sequences of sensor readings which represent a certain ADL. The underlying assumption in this approach is that the information collected in an intelligent home environment when a specific ADL is performed, can be transformed into a sequence of ordered sensor readings. The model of the ADL is represented by a distribution of relevant sub-sequences. However, as the way to perform an activity by an individual is not fixed but rather it changes and evolves, we propose a classifier able to keep up to date the models of certain ADLs. The proposed evolving classifier EvCDSsr is based on *Evolving Systems* and it is one pass, non-iterative, recursive and it has the potential to be used in an interactive mode; therefore, it is computationally very efficient and fast.

The test results with a real dataset demonstrate that EvCDSsr performs as well as other well established off-line classifiers in terms of correct classification on validation data. However, the proposed classifier is able to adapt dynamically to new data.

REFERENCES

[1] R. Agrawal and R. Srikant, 'Mining sequential patterns', in *International Conference on Data Engineering*, pp. 3–14, Taipei, Taiwan, (1995).

[2] P. Angelov and X. Zhou, 'Evolving fuzzy rule-based classifiers from data streams', *IEEE Transactions on Fuzzy Systems: Special issue on Evolving Fuzzy Systems*, **16**(6), 1462–1475, (2008).

[3] P. Angelov, X. Zhou, and F. Klawonn, 'Evolving fuzzy rule-based classifiers', in *Proceedings of the IEEE Symposium on Computational Intelligence in Image and Signal Processing*, pp. 220–225, (2007).

[4] E. Fredkin, 'Trie memory', *Comm. A.C.M.*, **3**(9), 490–499, (1960).

[5] J. A. Iglesias, P. Angelov, A. Ledezma, and A. Sanchis, 'Modelling evolving user behaviours', in *IEEE Workshop on Evolving and Self-Developing Intelligent Systems, ESDIS*, pp. 16–23, (2009).

[6] J. A. Iglesias, A. Ledezma, and A. Sanchís, 'Sequence classification using statistical pattern recognition.', in *Proceedings of Intelligent Data Analysis*, volume 4723 of *LNCS*, pp. 207–218. Springer, (2007).

[7] School of Electrical Engineering and Computer Science Washington State University. CASAS Smart Home Project (http://ailab.wsu.edu/casas/), 2010.