

On-line ADL Recognition with Prior Knowledge

Jonas Ullberg and Silvia Coradeschi and Federico Pecora¹

Abstract. This paper addresses the problem of recognizing activities of daily living. The novelty lies in the use of an existing knowledge base (ConceptNet) to introduce prior knowledge into the system in order to reduce the amount of learning required to deploy the system in a real environment. The use of household objects is central in the recognition of activities that are being performed, and we attach semantic meaning to both the objects and activities that are being recognized. The paper describes a framework which is specifically geared towards realizing activity recognition systems which leverage prior knowledge. A preliminary implementation of a neural network based recognition system built on this framework is shown, and the added value of prior knowledge is evaluated through the use of various data sets.

1 INTRODUCTION

The ability to recognize activities of daily living (ADLs) is pivotal for developing intelligent home environments since it enables them to provide proactive contextualized services that help the resident concretely and in a timely fashion. For instance, if an intelligent home environment is aware of the fact that the resident is taking a shower, it could turn off the TV in order to save electricity. Many approaches to the problem of recognizing ADLs have been studied. However, these approaches have focused mainly on the challenging pattern recognition aspects of the problem, while disregarding some key practical aspects of the proposed solutions. While systems described throughout the literature often have a high rate of success and can in some cases recognize both concurrent and interleaved activities as well as fine grained activities [12], this efficacy inevitably comes at a cost. Specifically, the proposed systems tend to require extensive model training with significant amounts of annotated data before being deployed in real environments. Furthermore, the training data used in the process is difficult to reuse for training models in other settings. In addition, the performance of these systems is often evaluated on the entire data set at once, which says little about how well such systems would be able to detect the activities in a timely fashion.

In this paper we draw inspiration from previous attempts and try to overcome these weaknesses by shifting the focus slightly from the pattern recognition aspects of the problem to a more practical one that focuses on recognizing basic ADLs in a generic way that can be more easily accepted by the end-user.

The main goal of our work is to develop a system that provides two key features that have not gained much attention in the activity recognition literature. Specifically, we want our system to; (1) be flexible enough to be deployed in many different settings without any prior training, and (2) allow modular enhancements to accommodate functionalities that make existing systems adaptable and precise

(e.g., through the use of learning).

We accomplish this by: (a) developing a stream-based framework, inspired by Digital Signal Processing (DSP) techniques which leverages prior knowledge and is able to accommodate both event-based and continuous input, (b) accepting a coarser level of detail on the recognized activities, and (c) representing as output the estimated probabilities of performed activities rather than the best match. The latter point is particularly beneficial since it allows us to represent uncertainty in the output of the activity recognition process.

The activity recognition system presented in this paper takes as input objects that are being used in the environment and uses information retrieved from the freely available ConceptNet [8] common-sense knowledge base (KB) to detect activities as they occur. ConceptNet contains knowledge about objects and their respective uses which we use to create rough estimates about which activity is being performed at any given time. The claim is that in so doing we can reduce the dependency of our system on training, thus allowing a non-trained, out-of-the-box deployment of the system which already provides coarse ADL recognition capabilities.

This paper is organized as follows. Section 2 describes related work in the field of activity recognition which is either relevant or complementary to ours. Section 3 gives an overview of the architecture of our system and the parts of ConceptNet that are relevant to our work. Section 4 describes some preliminary experimental results. Finally, Section 5 presents our conclusions and lists some possible future work.

2 RELATED WORK

Prior approaches to the problem of recognizing activities of daily living can roughly be categorized by how the input data is processed. A common solution is the *data-driven* approach in which models of human behavior are learned from large sets of sensory data recorded over time. These models often employ Hidden Markov Models (HMM) or variations thereof to process the data. A notable example is described in [12], in which activities are being recognized with the help of an RFID-based glove and a series of probabilistic graphical models. Also, [10] shows how location data and Relational Markov Networks (RMR) can be used for this purpose. More recently, [4] has described the use of Switching Hidden Semi Markov Models (SHSMM) to recognize activities. Activity recognition systems which employ Neural Networks are more uncommon but a few exist, such as the one described in [7], where human motion patterns are used to recognize activities. Also, some unique approaches exist such as in [6], where a data mining technique called emerging patterns is used to discriminate concurrent and interleaved activities.

A complementary set of approaches to ADL recognition can be categorized as *knowledge-driven*. Such approaches are based on sets of rules that define constraints on the activities to be recognized.

¹ Center for Applied Autonomous Sensor Systems, Örebro University, Sweden, email: {*name*}.{*surname*}@oru.se

These rules can be specified by a knowledge engineer explicitly, as in [13] and [3], where the activity recognition problem is solved by finding patterns of events in the sensory input that conform to a set of temporal constraints. It has been shown that temporal constraints provide a useful and flexible language for specifying requirements for ADL recognition (e.g., Allen’s temporal relations) [1]. Other knowledge-based models can be trained as in [2], where temporal constraints are employed to recognize activities outside the home with the help of the Bluetooth neighborhood and the current Cell ID of a mobile phone. In this case, learning is based on the computation of common time-use sub-graphs.

Data and knowledge driven approaches have complementary advantages and drawbacks. The former requires extensive data collection and subsequent training prior to being deployed in a specific setting. The latter category of approaches suffers from a similar drawback, namely the extensive work necessary to model discriminatory rules and consequent portability of the knowledge. It appears that no approach so far described in the literature has contributed to addressing the trade-off between knowledge modeling requirements and the need to re-learn from scratch once the setting is (re)defined.

There are, however, a few examples of hybrid approaches in the literature. Among these, [15] has shown that it is possible to obtain an accuracy of 42% over 26 activities using activity models mined without supervision from the web. The input to their system was a set of words describing the activities to be recognized. These words were then used in combination with the discriminating phrase “how to” to locate pages in the instructional genre from which they extract relevant objects with the help of WordNet. In a later work [14], it has also been shown that it is possible to use a statistical smoothing technique called hierarchical shrinkage to counteract the incompleteness of mined models. The main difference with our work lies in the fact that we bootstrap the activity recognition process with the objects available in the environment and then recognize all possible activities which they can take part in, providing a probabilistic rather than crisp output. Also relevant is LifeNet [11], a computational model of human life that attempts to anticipate and predict what humans do in the world. This approach is in some ways similar to ours. However, their system is different from an architectural point of view, and most importantly their emphasis is on forecasting what will happen in space and time in a typical person’s life, being able to show this in human language, and to a lesser extent recognize human activities as they occur.

3 STREAM BASED ARCHITECTURE

Our activity recognition approach is inspired by Digital Signal Processing (DSP), in that we cascade a series of interconnected *operators* acting on multi-channel data streams. The streams represent sensor data, and each operator acts as a *source* of data, a *sink* of data, or both. A data stream is not a construct in our system per-se but a terminological convenience to denote a connection between a source and a sink. Thus, an operator which implements the source interface can be connected to an operator implementing the sink interface, and if the latter also implements the source interface it can transform the data received from its connected source and provide it to another component.

The data emitted by the sources in our system are time-stamped data packages in the form $D = \{t, \langle a_1, a_2, \dots, a_n \rangle\}$, where t is the time stamp of the package and $\langle a_1, a_2, \dots, a_n \rangle$ is an ordered set of real-valued *activation factors*. Each source also has an ordered set of keys $K = k_1, k_2, \dots, k_n$ associated with it, where $k_1, k_2, \dots, k_n$ are

symbols that provide *semantic meaning* to the activation factors of the data. Even though it is not an inherent property in the formalism described here, the intended use of the activation factors is probabilistic in the sense that $a_i = P(k_i)$ so that a_i defines to which extent k_i holds true at time t .

When the system is initialized, and before any data is sent, each source operator dynamically generates its key set, thereby defining its “topic”. For instance, a source operator that is not a sink but provides data from an external source such as a log file in CSV format might scan the file and generate its key set based on the column headers in the file. In contrast, if the source operator is a sink as well (which is the most common situation) this is typically achieved by applying some semantic transformation on the parent’s key set. The set of keys will then remain the same during the lifespan of the system as the data is relayed between the sources and the sinks. An operator that always provides the same set of keys as its parent will be referred to as a *transparent* operator in this text.

As a consequence, an operator that implements both the source and the sink interface is able to act as a causal² filter F on the data stream during run time and on the keys at start up. Furthermore, operators can be composed so that $G \circ F \circ \dots = G(F(\dots))$, both with respect to keys and activation values.

3.1 ConceptNet

ConceptNet [8] is a freely available commonsense knowledge base trained by ordinary people who have contributed content via the web. From our perspective the most relevant constructs in ConceptNet are the *concepts*. An example of concept is an item *stove* or an activity such as *cooking*. ConceptNet also contains *relations*, for example *UsedFor*, and *assertions* that connect two concepts together with the help of one relation, for example $\langle \text{stove}, \text{UsedFor}, \text{cooking} \rangle$. In addition, each assertion has a score that determines its reliability and a polarity that defines if the assertion is made in a positive or negative way. For instance, a reliability score of 0 indicates that an assertion is considered to be irrelevant or nonsensical by the user community, while an assertion using the relation *UsedFor* with a polarity of -1 defines that a concept is not *UsedFor* some other concept.

In the current system we use the concepts in ConceptNet as keys for the stream operators. This allows us to mine an ever increasing amount of user contributed data, and thus to create operators that use information about relations between concepts to provide new, different sets of keys with other interpretations than the operators’ parent’s keys. For instance, given a parent operator whose keys represent objects in our domain, we can construct an operator that casts objects into activities with the help of the *UsedFor* relation mentioned earlier.

Compared to other KBs in widespread use such as WordNet [5] or CyC [9], ConceptNet was chosen due to the fact that it contained more information about objects and activities than CyC, and more relevant relations than the structurally similar WordNet, which contains only limited information about usage and is more exclusively a semantic network. Also, both CyC and ConceptNet are both largely handcrafted by knowledge engineers, a fact that we perceive as a drawback since the growth of information can be assumed to be slower, and the information itself less inspired by the real-world.

² In DSP terminology the word causal means that the output only depends on past and present inputs and not future ones.

3.2 System structure

A basic setup of the system that is able to provide an estimate about a user’s activity can be constructed with the help of a few operators and is illustrated in Figure 1. The topmost operator in the connec-

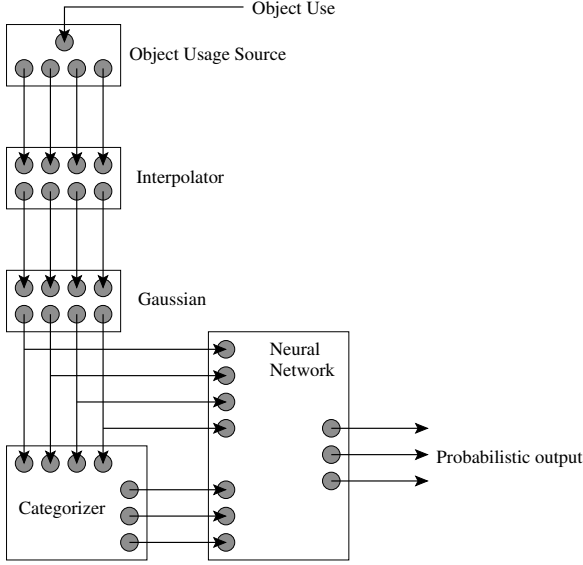


Figure 1. A basic activity recognition system realized in our architecture.

tion hierarchy (labeled “Object Usage Source”) is a source of object use events which provides a key set consisting of ConceptNet concepts that correspond to objects in a physical environment. The key set is generated at start up and contains all objects that can be used in the environment. Thus, the set of object concepts that can appear in the domain has to be known in advance or derived in some way at start up. The object usage source itself can take its input from different sources, for example from a file that contains logged usage data or on-line from a physical environment as objects are being used. When an object is being used or stops being used, this source emits a data package with a monotonically increasing time stamp that corresponds to the current time in milliseconds where $a_i = 1$ if the object that corresponds to k_i is being used, otherwise $a_i = 0$. As mentioned, the activation factors are probabilistic and the choice of values here reflects the fact that we treat the incoming data as the ground truth. This operator is designed to be able to be plugged into various data sources. It provides an abstraction layer between a physical, artificially generated or logged source of object use information and the activity recognition system itself.

The second operator (labeled *Interpolator*) is a sink and source of concepts which discretizes time into regularly sized time-steps with a user defined time span of ΔT . When this operator’s parent source emits data with a time stamp t , this data gets aggregated if $t \leq t_* + \Delta T$ where t_* is the last emit time, otherwise the data is emitted with the time stamp t_* and t_* is set to $t_* + \Delta T$ subsequently. Since the activation values are continuous, aggregation consists of a piecewise max operation (a fuzzy OR operation), which allows to represent concurrent object use within the time frame $[t_*, t_* + \Delta T]$.

The output of the second operator is semi-continuous in time but binary in value since each activation factor can only take a value of either 0 or 1. To model how the significance of object use in the past decreases over time we add a third operator that applies a Gaussian

decay function to the data stream. This operator implements both the sink and the source interface, and adds a decay time to the activation peaks in the data stream. This is done to bridge the gap between single-use objects (events) and continuous-use objects so that we can use both types of sensor readings in the same framework. The operator’s output activations are calculated from the input activations as

$$a_{now} = a_{max} * e^{-\frac{(t_{max} - t_{now})^2}{c^2}}$$
 where c is a predefined width factor of the Gaussian, t_{now} is the current time and t_{max} and a_{max} are the last time when the input activation was greater than the output activation produced by the Gaussian and the input activation at that time. However, since the input to this operator in the current setup is always 0 or 1, the effect is equal to applying a Gaussian decay to the output when the input is 0 from the last point the input was 1 and peak the output directly to 1 as the input changes to 1 from 0. An example of the output of this operator can be seen in Figure 2, which shows how the activation factors of different concepts corresponding to objects that are being used in the domain vary over time. The width factor c of the Gaussian is set to some reasonable value depending on the objects in the domain, for example $60000 \text{ (ms}^{-1}\text{)}$. Naturally this operator is transparent to its parents keys as well.

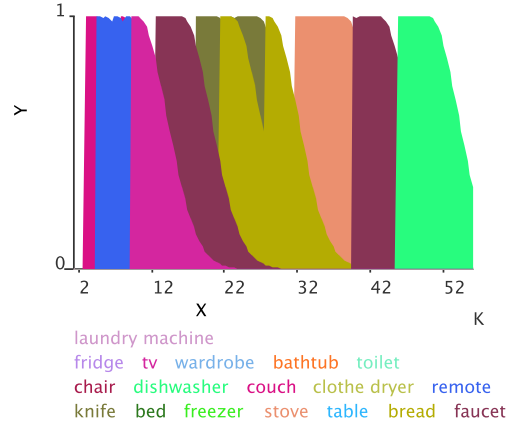


Figure 2. Output of the Gaussian operator, which indicates estimated probabilities of object use over time.

In order to cast the estimated probabilities of objects being active into probabilities of activities occurring, we define an operator called a *Categorizer*. The Categorizer uses ConceptNet to change the domain of the data stream. It is both a sink and a source of concepts, but is not transparent to the parent’s key set. When initialized, the Categorizer takes a ConceptNet relation R as an argument and uses this to cast the keys and the streaming data of the parent into a new dimension. At start up the operator checks the keys of the parent operator and for each key k_i it mines ConceptNet to find assertions of the form $A_{ij} = \langle k_i, R, x_{ij} \rangle$. The key x_{ij} is added to the operator’s key set if the assertion meets a certain goodness criterion which is determined by the reliability score of the assertion (indicated in ConceptNet), and optionally the number of keys of the parent that are involved in the same assertion. The assertions that meet the criteria are then used to build a $n * m$ matrix where n is the number of keys of the parent and m is the number of output keys. In this matrix, w_{ij} determines how much the parent key k_i influences the output key k_j . These weights are initialized to the reliability score of the assertion multiplied by the total number of assertions made on the target concept with the relation R and then normalized so that

$\forall i, \sum_{j=0}^m a_{ij} = 1$. The matrix is then used to (linearly) transform the input data to the output dimension. Also, before the data is sent it is normalized so that $\sum_{i=0}^m a_i = 1$.

By using this setup and by parametrizing the Categorizer with a `UsedFor` relation, we create a basic activity recognition system that is able to provide educated guesses about which activities are being performed. Figure 3 shows the results of transforming the data illustrated in Figure 2 into a domain of activity concepts. Due to the fact that ConceptNet is built up by the user community, the data is inherently noisy and contains many irrelevant assertions. In order to give an impression about the amount of irrelevant assertions provided by ConceptNet, we can state that when mining all possible activities for the 18 common household objects shown in Figure 2, we ended up with 367 possible activity concepts. Among these, 94.5% were only referenced by one object, while the remaining 5.5% shared two or more objects in common. We can expect that the activity concepts that are relevant fall within the latter category in most domains. We can thus reduce the noise to a large extent by pruning the set of activities using a simple reliability criteria as mentioned above and/or demand that activity concepts should be referenced by at least two object concepts in our domain, which is effective in practice but makes an unwarranted assumption on the domain. However, the operator will still choose some irrelevant or duplicate concepts as can be seen in Figure 3. For instance, `sit` and `sit down` have the same meaning and can in addition be considered to be more basic than for example `cook`; or the presence of the concept `butter`, which is not an activity.

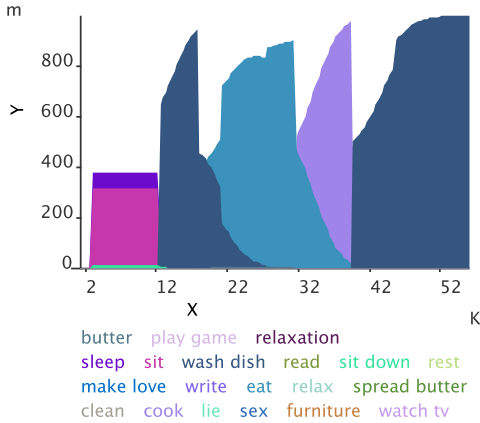


Figure 3. Estimated probabilities of activities being performed over time.

In those cases where the activities that should be monitored are fairly different in terms of object use, this setup can provide adequate results by letting the end user or an automated system pick the relevant concepts. In the remaining cases there is a need for further refinement. In order to overcome the coarseness of this out-of-the-box, untrained system, we resort to another type of operator that can be trained by the user on-line.

3.3 On-line learning

In order to adapt the noisy result produced by the chain of operators described earlier, we have provided interfaces for an operator that can be trained by the user on-line. Currently we have implemented one such operator that contains a fairly standard three layer neural network (NN) with a linear input layer, a sigmoid hidden layer and

a Softmax output layer. The neural network is a source of concepts corresponding to activities and a sink of object use information. The size of the neural network’s hidden layer is determined by its key set which is a parameter defined at start up. Currently, we use the key set computed by the Categorizer with the `UsedFor` relation. The neural network operator can also provide an arbitrary number of additional “pseudo” sinks which can be connected to other sources to provide auxiliary input to the network. This is very useful since it enables us to attach a number of Categorizers that mine additional information from ConceptNet, using for instance the relation `AtLocation`, which enables the neural network to generalize its training. That is, if the user explicitly trains the neural network to recognize that a `stove` is used for the activity `cook`, it also relates the training to the mined location property `kitchen` of the object. When an untrained object later appears as used in the data stream, the neural network is able to classify its usage on these features as well. For instance, an untrained `knife` which shares the mined location property `kitchen` would make use of the training given for the `stove` object and be classified to be `UsedFor cook` to some degree depending on the features it shares with the other trained objects in the domain.

The motivation for using a neural network to provide the adaptability is that it is expressive in terms of the relations it can encode and that it is able to provide a continuous output as any other operator in our system. Naturally this means that we can not easily disambiguate different activities with the help of historical information, we will however implement such functionality in our system in the near future.

4 EXPERIMENTAL RESULTS

We have performed an initial evaluation of our system using the data set described in [12]. The set contains usage data on 60 different RFID-equipped objects such as “cupboard”, “faucet”, “saucepan” or “vanilla_syrup”, which are used for 11 different annotated fine-grained household activities which can take place concurrently (for example “Making oatmeal” and “Eating breakfast”). The data set describes a morning household routine which consists of going to the bathroom, preparing breakfast and then eating it. This sequence of actions is repeated by a human user ten times with slight variations.

The evaluation was carried out without employing the neural network component in our system. Our goal here is to assess whether it is possible to employ the prior knowledge in ConceptNet (specifically, the relevance scores for the assertions in ConceptNet) to select activities to monitor in an unsupervised manner. It should be noticed that a system exhibiting this capability is by construction unaffected by the introduction of additional activities, nor does its accuracy depend on the size of the domain.

Since our aim is not to perform fine-grained activity recognition, nor to recognize concurrent activities, we have chosen to evaluate how well our unsupervised approach is able to give a high activation of important concepts and how well the activities can be identified. We have done this by considering all instances of preparing any kind of meal as a `cook` activity, and all instances of eating breakfast as an `eat` activity. This downscaling makes the activities disjoint in this data set so that there is no time instant which has both the `cook` and `eat` activities as ground truth at the same time. We then proceed and classify these two activities while leaving out the “Use The Bathroom” activity since it is quite short and too vague to be translated into a meaningful concept. It should be mentioned explicitly that this changes the activity recognition problem described in [12]: on one hand, it simplifies the problem as we only need to distinguish be-

tween two activities; on the other hand, the problem becomes more difficult since we consider any time instant in which the user is cooking as a cooking activity exclusively, thus we are negatively penalized by the influence of some concurrent activities. Although not fully suited for our needs, we have decided to use this data set to perform a simple evaluation of our system since it contains many instances of object use while at the same time being relatively long.

We have manually mapped the 60 object names in this data set into 36 ConceptNet concepts which we use as keys in the input by using a best-match-first principle. This means that we choose a concept in ConceptNet that has the same name as the given object label in the data set when possible, which was the case for most labels. The downscaling comes from the fact that the data set contains different instances of the same types of objects which we do not take into account currently and that it also tags different parts of the same object. From this data, our system automatically selects 44 activity concepts to use as outputs. The concepts which correspond to objects in the domain are visible in Figure 4, here scaled by the number of assertions made on them in ConceptNet.

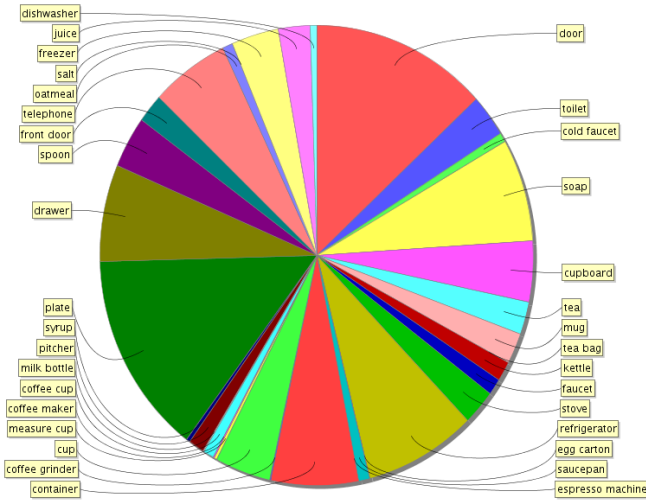


Figure 4. ConceptNet concepts used to represent objects. The relative sizes of the portions in the pie chart are determined by the amount of knowledge ConceptNet contains on each object.

When classifying the data and trying to classify the cooking and eating activity without any prior information, our system selects 44 activity concepts to use as outputs and the cooking concept is the output with the maximum activation 19.8% of the time during the cooking activity, while the eating concept is the likeliest activity 33.8% of the time during the eating activity, both well above the baseline of 2.2%. This proves that it is possible to use the relevance score for the assertions in ConceptNet to pick activities to monitor. Also, due to the fact that the set of activities we classify is generated automatically from the set of objects, this result would be unaffected by the introduction of additional objects used for other activities. Thus we are always able to provide this accuracy on these two activities regardless of the domain size.

It should also be mentioned that with prior knowledge of the fact that we should only distinguish between the cooking and the eating activity, the answer is only correct in 47.2% percent of the cases for the cooking activity and 98.5% of the time for the eating activity (compared to the baseline of 50%). This bias is due to the fact that the

cooking activity in this data set involves some objects that are more intuitively used for eating and also because the person is setting the table (thus touching forks, knives and dishes) while cooking. This shows the need for user adaptability to distinguish between different activities, and/or taking into account the location of the person. These results are calculated as a mean over all 10 instances in the data set. We can however expect that these types of classification errors will not grow when classifying additional activities which have a more disjoint object use such as taking a shower. In such cases, the baseline would be reduced and the static classification would be able to provide an interesting prior into an adaptable system with little effort. When taking into account the current quality and coverage of ConceptNet, the most realistic application of it today is as a source of prior knowledge that interacts positively with more sophisticated and adaptable operators such as a NN as proposed above

4.1 Generator of object use events

Due to the lack of publicly available long term object use data sets that suit our needs, we have resorted to implementing a virtual source of object use events in order to test our system. The motivation for doing this rather than creating our own custom data set is that we want to be able to quickly and repeatedly try with different variations in object use behavior in different scenarios containing different objects in order to pinpoint any problems. The use of datasets obtained through the generator during development allows us to training against a fixed data set would make our approach vulnerable to over-customization in the sense that we choose operators and their configurations to get good results in a particular data set.

For example, by utilizing operators that make use of and enforce temporal ordering information mined from ConceptNet, e.g., `<eat,HasPrerequisite,cook>`, we have been able to get almost perfect results on the data set presented in [12]. However such configurations are only possible if we were able to make rough assumptions in advance about the length of the `cook` activity and define a corresponding parameter accordingly. The generator of object use information makes it simple for us to pinpoint these kinds of problems and evaluate different solutions for them.

The generator is a source of concepts that corresponds to objects as described previously. When the generator is instantiated, it presents the user with a graphical user interface such as the one seen in Figure 5, which uses an image of a floor plan and an accompanying XHTML image map to define a home along with regions containing objects. The user can then mimic different activities by hovering the mouse over any object to produce object-use events. If the user clicks the objects, they become continuously activated until the user deactivates them. The user can also specify the ground truth activity that is being performed at any given time in order to train any adaptable operator such as the neural network present in the set of operators. This would correspond to a user giving feedback in a real environment. The user interface also allows the user to inspect different aspects of the activity recognition system, such as the estimated probability of different activities occurring as in Figure 6, the number of assertions made on different concepts as in Figure 4, or the correlation between instances of object use (e.g., the `stove` tends to be used together with the `saucepan`).

According to our observations, the system is able to provide correct results in about 50% of the cases without any modifications. For instance, it classifies correctly that the `stove` is used for cooking, the `faucet` is used for the activity wash dish, and that the `table` is used for eating (amongst other things). In the remaining

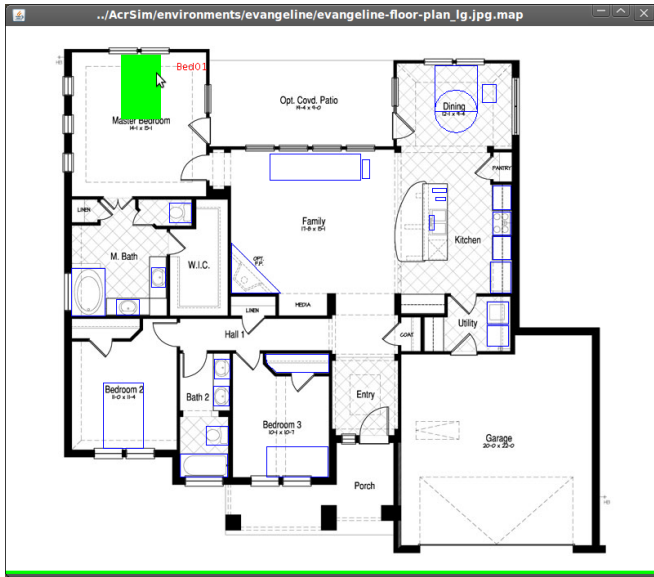


Figure 5. The main user interface of the generator of object use events.

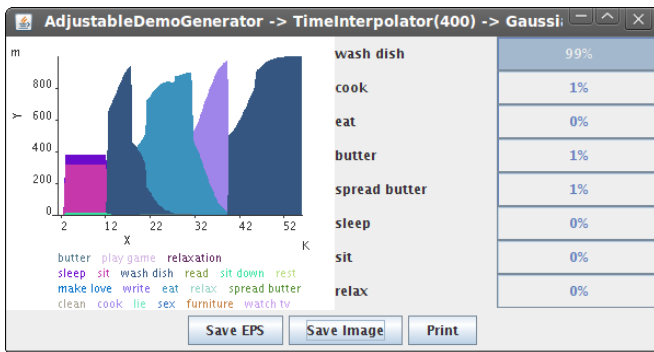


Figure 6. GUI that enables the user to inspect the probabilities of different hypotheses.

cases, the relevant concept is usually ranked at the second place among the top rated approximations. The most common problem is that some ConceptNet concepts lack associated usages, or the provided usages are few and unintuitive. This is mostly the case for (intuitively) uncommon objects such as a clothes dryer, but can also occur for more common objects such as a laundry machine. Furthermore, the user community seems to be inclined to add assertions on concepts such as *sex* rather than on *sleep*, giving the former activity concept a higher reliability than the latter when using the *bed*.

5 CONCLUSIONS AND FUTURE WORK

Being able to recognize ADLs is a fundamental capability for any intelligent home environment that provides contextualized services to its residents. However, prior solutions to this problem either require extensive model training or disproportionate modeling effort in order to be deployed in real environments. In this paper, we have presented the first step of a work in progress that aims to provide on-line ADL

recognition in a way that is both practical and non-intrusive. The novelty of our approach lies in how we overcome the extensive training requirements of previous activity recognition systems by integrating easily accessible knowledge about the possible usages of household objects into our system. This integration is realized through the implementation of different modular operators which simplifies the task of configuring the system for deployment in new environments and at the same time allows us to attach semantic meaning to our sensory input which is made available for reasoning in all parts of our system.

We acknowledge the need for experimental comparisons of the proposed system to previous ones, such evaluations will be performed as more elaborate classifiers are integrated as operators in the framework and will also serve as evaluations of the respective strengths and weaknesses of different approaches. This will include taking into account the temporal relations between instances of object use and activities. At the same time, we will develop methods for interacting with the user and explore how the user provided feedback can be seamlessly combined with the estimate provided by the knowledge base. We will also evaluate the benefits of using multiple aspects of the data to achieve abstraction over different categorizations of objects in the domain. Finally, we also plan to integrate the ADL recognition system in a intelligent home environment maintained in our labs.

REFERENCES

- [1] J.F. Allen, 'Towards a general theory of action and time', *Artificial Intelligence*, **23**(2), 123–154, (1984).
- [2] D. Choujaa and N. Dulay, 'Tracme: Temporal activity recognition using mobile phone data', in *EUC '08: Proceedings of the 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, pp. 119–126, Washington, DC, USA, (2008). IEEE Computer Society.
- [3] C. Dousson, P. Gaborit, and M. Ghallab, 'Situation recognition: Representation and algorithms', in *IJCAI*, pp. 166–174, (1993).
- [4] T. Duong, D. Phung, H. Bui, and S. Venkatesh, 'Efficient duration and hierarchical modeling for human activity recognition', *Artificial Intelligence*, **173**(7-8), 830–856, (2009).
- [5] *WordNet An Electronic Lexical Database*, ed., C. Fellbaum, The MIT Press, Cambridge, MA ; London, May 1998.
- [6] T. Gu, Z. Wu, X. Tao, P.H. Keng, and J. Lu, 'epsicar: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition', in *Proceedings of the 2009 IEEE International Conference on Pervasive Computing and Communications (PERCOM '09)*, pp. 1–9, Washington, DC, USA, (2009). IEEE Computer Society.
- [7] N. Györfi, Á. Fábán, and G. Hományi, 'An activity recognition system for mobile phones', *Mob. Netw. Appl.*, **14**(1), 82–91, (2009).
- [8] C. Havasi, R. Speer, and J. Alonso, 'Conceptnet 3: a flexible, multilingual semantic network for common sense knowledge', in *Recent Advances in Natural Language Processing*, Borovets, Bulgaria, (2007).
- [9] D.B. Lenat, 'Cyc: a large-scale investment in knowledge infrastructure', *Commun. ACM*, **38**(11), 33–38, (1995).
- [10] L. Liao, D. Fox, and H. Kautz, 'Location-based activity recognition using relational markov networks', in *Proceedings of the Nineteenth International Conference on Artificial Intelligence, IJCAI'05*, (2005).
- [11] B. Morgan and P. Singh, 'Elaborating sensor data using temporal and spatial commonsense reasoning', in *BSN*, pp. 187–190, (2006).
- [12] D.J. Patterson, D. Fox, H. Kautz, and M. Philipose, 'Fine-grained activity recognition by aggregating abstract object usage', *Wearable Computers, IEEE International Symposium*, **0**, 44–51, (2005).
- [13] F. Pecora and M. Cirillo, 'A Constraint-Based Approach for Plan Management in Intelligent Environments', in *Proc. of the Scheduling and Planning Applications Workshop at ICAPS09*, (2009).
- [14] E.M. Tapia, T. Choudhury, and M. Philipose, 'Building reliable activity models using hierarchical shrinkage and mined ontology', in *Proceedings of Pervasive 2006*. Springer-Verlag, (2006).
- [15] D. Wyatt, M. Philipose, and T. Choudhury, 'Unsupervised activity recognition using automatically mined common sense', in *AAAI'05: Proceedings of the 20th national conference on Artificial intelligence*, pp. 21–27. AAAI Press, (2005).