Automatic Recognition of Human Activities Combining Model-based AI and Machine Learning

Constantin Patsch, Marsil Zakour and Rahul Chaudhari
Human Activity Understanding Group, Chair of Media Technology (LMT), Technical University Munich (TUM), Arcisstr. 21, Munich, Germany

Keywords: Activity and Plan Recognition, Knowledge Representation and Reasoning, Machine Learning.

Abstract: Developing intelligent assistants for activities of daily living (ADL) is an important topic in eldercare due to the aging society in industrialized countries. Recognizing activities and understanding the human's intended goal are the major challenges associated with such a system. We propose a hybrid model for composite activity recognition in a household environment by combining Machine Learning and knowledge-based models. The Machine Learning part, based on structural Recurrent Neural Networks (S-RNN), performs low-level activity recognition based on video data. The knowledge-based part, based on our extended Activation Spreading Network architecture, models and recognizes the contextual meaning of an activity within a plan structure. This model is able to recognize activities, underlying goals and sub-goals, and is able to predict subsequent activities. Evaluating our action S-RNN on data from the 3D activity simulator *HOIsim* yields a macro average F1 score of 0.97 and an accuracy of 0.99. The hybrid model is evaluated with activation value graphs.

1 INTRODUCTION

With the increasing median age in industrialized countries, the relative portion of elderly people within the population is steadily increasing. For neurodegenerative diseases like Alzheimer's, assistive systems might help affected elderly people to accomplish tasks by providing guidance according to assessed intentions. Activity understanding systems for smart home environments based on sensory and visual data have proven to be capable of recognizing and predicting activities (Du et al., 2019) and also useful in health monitoring applications (Yordanova et al., 2019).

In order to recognize and understand human activities, we have to understand the contextual importance or relevance of an activity within the human's activity sequence towards a certain goal. Contextual importance indicates the significance of an activity within the plan structure as well as the logical interdependencies between activities. In this context, reasoning about the logic soundness of a recognized activity, given previous activity recognitions and the information of a plan structure, is essential. We assume that the humans are not explicitly conveying their intent to the system, and that the goal is not known beforehand.

The contribution of this paper is a hybrid activity recognition model consisting of our action Structural-RNN (S-RNN) architecture inspired by Jain et al. (2016) and a new Activation Spreading Network formulation based on the work of Saffar et al. (2015). We present an interoperation mechanism for the S-RNN and the ASN architectures. Our hybrid recognition model for composite activities, depicted in Figure 1, pursues the following objectives:

- contextual activity recognition based on logical interdependencies in the plan structure,
- predictions of feasible future activities,
- consideration of partial observability and missing activity recognitions,
- recognition of intended goals and subgoals, and
- consideration of interleaved activities contributing to multiple goals.

2 RELATED WORK

2.1 Model-based Artificial Intelligence

On the one hand, approaches based on an explicit plan library compare activity sequence recognitions to existing plans (Goldman et al., 2013; Levine and Williams, 2014; Saffar et al., 2015). On the other

Patsch, C., Zakour, M. and Chaudhari, R.
Automatic Recognition of Human Activities Combining Model-based AI and Machine Learning.
DOI: 10.5220/0010747000003116
In *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART 2022) - Volume 3*, pages 15-22
ISBN: 978-989-758-547-0; ISSN: 2184-433X

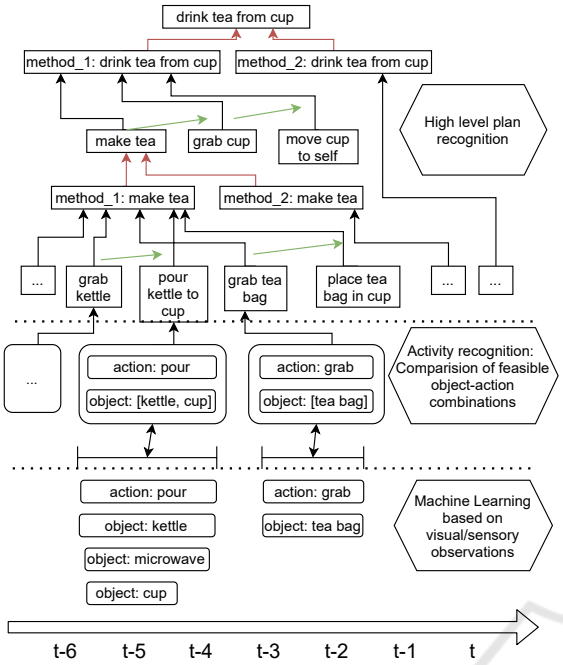

Figure 1: Hybrid model, showing exemplary recognitions and methods, within a tea drinking plan.

hand, generative approaches try to exploit ontological and probabilistic knowledge to synthesize feasible activity sequences given observations (Ramırez and Geffner, 2011; Yordanova et al., 2019; Chen and Nugent, 2019).

Goldman et al. (2013) propose a model for plan recognition based on an explicit plan library, that decomposes goals into activities by exploiting their hierarchical structure. They incorporate partial ordering by defining subtasks in terms of logical and temporal preconditions. Their model probabilistically assesses the most likely goal from the plan library based on the contribution of an activity observation to a plan.

Saffar et al. (2015) introduce an Activation Spreading Network (ASN) that captures the hierarchical dependencies between several abstractions of activities, while grouping them according to their affiliation to their respective subgoals and goals. Hereby, preconditions between subsequent activities introduce logical order. Action-object recognitions from the RGB-D video data enable activation value propagation throughout the plan library in order to recognize the most likely goal, and ensure a logically sound activity recognition process. Our work partly relies on the above concepts from Saffar et al. (2015), while we introduce activity predictions, improved long-term recognition robustness and compatibility with Machine Learning frameworks.

2.2 Machine Learning based Models

Machine Learning based models recognize activities based on sensory and visual data as shown in (Koppula et al., 2013; Jain et al., 2016; Du et al., 2019; Shan et al., 2020; Bokhari and Kitani, 2016). Du et al. (2019) use LSTMs and RFID data for activity recognition. Bokhari and Kitani (2016) employ Q-learning and a Markov Decision Process to capture an activity sequence progression. Shan et al. (2020) propose a framework for hand-object contact recognition and hand state estimation in order to understand human object interaction and human object manipulation based on video data.

In the context of human activity and object affordance learning from RGB-D data, Koppula et al. (2013) propose the CAD120 dataset based on spatio-temporal features while introducing semantic object affordances. Based on this dataset, Jain et al. (2016) introduce Structural Recurrent Neural Networks (S-RNNs) that use Deep Learning based on spatio-temporal graphs for action and affordance recognition. These graphs capture the interactions between the human and the surrounding objects within a temporal segment of an action. Actions are classified based on human-specific and shared human-object features with the help of the corresponding RNNs, whereas the object classification is based on object-specific and shared human-object features. Our work borrows the feature preprocessing process as well as the general framework of the S-RNN, while we directly use object-specific features for the action recognition process without considering affordances.

3 HYBRID ACTIVITY MODEL

3.1 Activation Spreading Network

In this section we explain our ASN architecture, an extension of the one presented by Saffar et al. (2015). We developed this ASN to meet the requirement of performing high-level contextual activity recognition in the hybrid activity recognition model. Compared to the work by Saffar et al. (2015), our ASN provides:

- compatibility with discrete Machine Learning model recognitions,
- weighting based on activity distinctiveness,
- contextual recognition of longer complex activity sequences, independent of activity duration,
- activity predictions, and
- recovery from misclassified and missing activities

3.1.1 Extended Activation Spreading Network Architecture

The ASN is a directed acyclic graph consisting of nodes $n \in \mathcal{N}$, sum edges $e \in \mathcal{E}_s$, max edges $ma \in \mathcal{E}_{ma}$ and ordering/precondition edges $o \in \mathcal{E}_o$. $\mathcal{N}$ represents the set of all nodes and $\mathcal{E}$ the set of edges in the ASN, where $\mathcal{E}_{ma} \subset \mathcal{E}$ and $\mathcal{E}_o \subset \mathcal{E}$. $\mathcal{N}$ consists of operator nodes $n_o \in \mathcal{N}_o$, method nodes $n_m \in \mathcal{N}_m$ and compound nodes $n_c \in \mathcal{N}_c$, where $\mathcal{N}_o \subset \mathcal{N}$, $\mathcal{N}_m \subset \mathcal{N}$ and $\mathcal{N}_c \subset \mathcal{N}$ is valid.

The operator nodes are the leaf nodes within the network representing activities that can be recognized by the low-level activity recognition framework. The method nodes take the sum over the weighted activation values of the child nodes that are connected to it with sum edges. Hereby, each sum edge connected to a method has its own weight. The assignment of the activities to the respective sum edges is captured in the *SumEdges* dictionary according to definition 3.2. All methods connected to the same compound node with max edges represent different ways of achieving that compound node. The max edges $m \in \mathcal{E}_m$ only enable the method that has the highest activation value between competing methods to spread its activation value. Compound nodes can either again contribute to their parent method, or if they are on the highest level within the ASN, they are denoted as goal nodes.

Definition 3.1 (Activation Value Dictionary)**.** *ActValN* contains the activation value of every node $n \in \mathcal{N}$. The activation value $ac(n) \in [0,1]$ of each node is calculated based on the low-level activity recognition and activation value propagation process. The structure is defined by:

$$ActValN = \{n_1 : ac(n_1), n_2 : ac(n_2), ..., n_k : ac(n_k)\},$$

where k denotes the number of available operator, method and compound nodes in the plan library and $ac(n)$ denotes the activation value of the node $n \in \mathcal{N}$.

Definition 3.2 (Sum Edge Dictionary)**.** *SumEdges* contains the activities contributing to their respective methods. Accordingly, *SumEdges* is formulated as:

$$SumEdges = \{n_{m_1} : [l_1], n_{m_2} : [l_2], ..., n_{m_j} : [l_j]\},$$

where j denotes the number of all available methods within the plan library and $[l_j]$ denotes an array containing the specific activity nodes $n \in \mathcal{N}$ that are associated with the respective method $n_{m_j} \in \mathcal{N}_m$. Sum edges $e \in \mathcal{E}_s$ are displayed with black arrows in graphical plan structures.

Definition 3.3 (Max Edge Dictionary)**.** $n_m \in \mathcal{N}_m$ are method nodes that lead to a certain compound node $n_c \in \mathcal{N}_c$ which is determined by *MaxEdges* as:

$$MaxEdges = \{n_{c_1} : [methodlist_1], n_{c_2} : [methodlist_2], ..., n_{c_b} : [methodlist_b]\}.$$

Hereby, b denotes the number of compound nodes within the plan library and $methodlist_b$ denotes the method nodes that are associated with the respective compound node n_{c_b}. Max edges $ma \in \mathcal{E}_{ma}$ are displayed with red arrows in graphical plan structures.

Definition 3.4 (Ordering/Precondition Edge Dictionary)**.** *PrecondEdges* has the methods n_{m_j} as keys, while containing a list of precondition lists for each activity within the method:

$$PrecondEdges = \{n_{m_1} : [c(n_{f_1}), c(n_{f_2}), ..., c(n_{f_{m1}})], n_{m_2} : [c(n_{f_1}), c(n_{f_2}), ..., c(n_{f_{m2}})], ..., n_{m_j} : [c(n_{f_1}), c(n_{f_2}), ..., c(n_{f_{mj}})]\}.$$

Hereby, each $c(n_{f_{mj}})$ denotes a list of precondition activity nodes that are associated with the activity node $n_{f_{mj}}$. The number of entries f_{mj} associated with a method j depends on the number of activities that are assigned to the respective method in the *SumEdges* dictionary. Precondition edges $o \in \mathcal{E}_o$ are displayed with green arrows in graphical plan structures.

Definition 3.5 (Activation Sum Edge Dictionary)**.** *ActSumEdges* is a dictionary that contains a list of binary values, with a separate binary value for every activity within a method indicating whether the preconditions of the activity in *PrecondEdges* are fulfilled. Accordingly it is defined as:

$$ActSumEdges = \{n_{m_1} : actsum_1, n_{m_2} : actsum_2, ..., n_{m_j} : actsum_j\},$$

where j denotes the number of methods within the plan library and $actsum_j$ denotes the list of activation values of the sum edges connecting the activity nodes $n \in \mathcal{N}$ with their respective methods $n_{m_j} \in \mathcal{N}_m$.

In the following we explain our adaptations and extensions to the original ASN framework. Saffar et al. (2015) introduced a uniform decay of activation values, which we replace with a time independent binary activation value definition for operator nodes, in order to take long and complex activity sequences into account. Thus, we prevent the decay of an activity's relevance throughout time which means that activity durations do not influence the contribution to the recognition process. This formulation also enables the connection of the Machine Learning model outputs and the ASN operators, as the ASN is able to accept time discrete activity recognitions.

Moreover, we introduce a new weighting scheme for the edges connecting activity nodes within the ASN. Our weighting process relies on the frequency

of an activity within competing plans, and the higher importance of compound nodes that incorporate several different activities. We initialize the weights of the sum edges as $\frac{1}{|\text{subactivities}(n_m)|}$. The weighting of the sum edge of an activity is relatively increased to the other sum edges within the method, if the activity is rather unique for the respective plan. The weighting of sum edges connecting compound nodes to method nodes is amplified relatively to simple operator nodes. Moreover, we normalize the weights of sum edges, which enforces a comparability between activities independent of the hierarchical level.

As another extension we introduce state effects that represent the validity of a certain state upon completing a subgoal within the ASN. This is important for recognizing activity sequences that are likely to be repeated several times. A state effect is valid until another state effect is introduced.

Furthermore, we introduce a backpropagation procedure that enables predictions about future activities. At first, we determine the most likely goal with the highest activation value and then consider the child nodes of the method that constructs this compound node. We iteratively traverse the hierarchy towards the lowest levels in the plan constituting the currently assessed goal. In case of compound nodes, we iterate through the child nodes of their method. On each hierarchical level, the validity of the preconditions is checked as they serve as an indicator for possible next activities. When predicting future activities, the ones that have activated sum edges due to fulfilled preconditions and that are directly subsequent to previously activated activities are considered. Lastly, the ASN recovers from misclassifications and missed activities by setting the activation value of an activity to 1 if it has been missed or misclassified, while serving as a precondition for two subsequent successfully recognized activities.

3.1.2 Activation Spreading Process

The activation value propagation process is initiated by a new activity recognition. When the activation value of a newly recognized activity is updated to 1, we start by iterating from the lowest level methods to the highest level ones in order to ensure a correct activation value propagation within the hierarchy.

All preconditions of an activity have to be valid in order for an activity to spread its activation value. If all preconditions and state effect preconditions are fulfilled the respective sum edge of the considered activity is activated. The value of $ActSumEdges$ is updated from 0 to 1 for the relevant activity. After all activities of a method have been considered, the activation value of the method gets updated by summing over the weighted activation values from its activities. Upon a method achieving the activation value 1, the activation values of the activities involved with that method get reset to 0 while the compound node maintains its activation value at 1. The compound node itself is going to get reset if the method node it contributes to achieves the activation value 1.

3.2 Structural Recurrent Neural Network

In this section we explain the action recognition based on the action-affordance S-RNN proposed by Jain et al. (2016) and our action S-RNN.

3.2.1 Feature Preprocessing

In order for the S-RNN to be able to perform action and affordance recognition we first introduce the feature preprocessing steps that are inspired by Koppula et al. (2013). The features are computed based on skeleton and object tracking performed on stationary video data. The object node features depend on spatial object information within the segment, whereas the human node features rely on the spatial information of the upper body joints. The edge features are defined for object-object edges and human-object edges within one segment of the spatio-temporal graph. The temporal object and human features are defined based on the relations between adjacent temporal segments. Similar to Koppula et al. (2013), the continuous feature values are discretized by using cumulative binning into 10 bins yielding a discrete distribution over feature values. The resulting dimension of the feature vector thus yields $(\text{number of features}) \times 10$. As a result we obtain a histogram distribution over the feature values that is especially useful when adding object features.

The spatio-temporal graph depicted in Figure 2 represents a concise representation of the relation between the human and the objects within and between temporal segments. In order for the spatio-temporal graph to model meaningful transitions, the video is

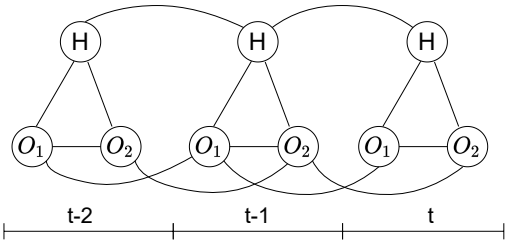

Figure 2: Exemplary spatio-temporal-graph with one human and two object nodes within three temporal segments.

segmented in a way that every segment ideally only contains one action. The segmentation of videos according to actions is not investigated here, and we rely on the provided ground truth segmentations. Labels of human nodes are the action labels whereas an object can be annotated with an affordance label. The semantic affordance of an object depends on the activity it is involved in. For example in the activity 'pour from bottle to glass' the action label is 'pour' and the affordance labels of the bottle and the glass are 'pourable' and 'pour-to'. Koppula et al. (2013) introduced the notion of affordances in order to define how an object is being interacted with in the scene.

3.2.2 Action-affordance S-RNN and Action S-RNN

Firstly, we consider the original S-RNN based on the joint action-affordance recognition. Hereby, an object only has one affordance at a time which can vary over time depending on its usage. Within a segment of the spatio-temporal graph, while there is always only one human node corresponding to one action label, there is generally a higher number of object nodes which varies depending on the scenario that results in a varying number of affordance labels. Thus, the overall action-affordance model has to be separated into two submodels, one dealing with affordance classification and the other dealing with action classification.

However, compared to Jain et al. (2016), we do not consider semantic affordances and only focus on action classification while relying on spatio-temporal-graph features. The architecture of this action S-RNN model is shown in figure 3. Hereby, the *Human_Input* layer receives the concatenation of the human node and the human temporal edge features as an input. As we consider only one human node within a temporal segment the number of features remains the same throughout temporal segments. The *Object_Input* layer receives the concatenation of the object node, the object-object edge and the object temporal edge features as an input. However, as the number of objects might vary throughout temporal segments, the length of concatenated features might vary accordingly. Thus, we have to sum over the object related features of each object in order to achieve a fixed length representation. Empirically, the aforementioned cumulative binning process for discretizing the features is important to limit the information loss during the feature summation. The *Human_Object* input layer receives the human-object edge features as an input. As the human node is connected to a potentially varying number of object nodes, we sum over the discretized human-object edge features.

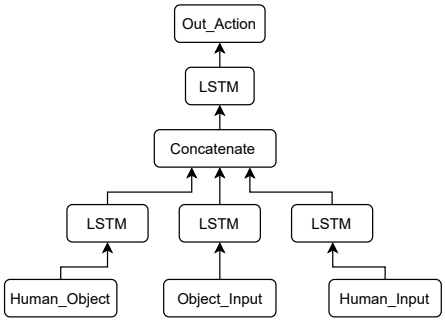

Figure 3: Our action S-RNN implementation.

The advantage of our approach is that we do not need affordance labels as we only focus on action classification which simplifies the dataset creation process. Differently to the original S-RNN the inputs are not divided into terms that contribute to either the affordance or the action classification but rather we directly use all features to perform action classification. Thus, the action classification is not only trained on the human and human-object edge features but also on object and object-object edge features of the spatio-temporal-graph representation.

3.3 Combining Machine Learning Results with ASN

In the following we combine action and object recognitions from the S-RNN into feasible action-object combinations which are passed as an input to the ASN model. We rely on simulated data where labels of objects and kitchen furniture are available. Thus, the object recognition part is not explicitly considered. The main focus of the matching process depicted in figure 4 is to combine probabilistic assessments regarding actions with object recognitions, and verify whether these action-object combinations result in feasible activities.

Given object recognitions, we only consider objects within imaginary spheres around the human hand joints (t_{sphere}). Feasible activities are obtained by comparing the action-object combinations a_i with available operator activity nodes within the plan library. For each combination we define a joint detection score consisting of three parts. The first part is the score $s_{dist-inv}(O_p)$ that is calculated for each object O_p within an action-object combination based on the equation

$$s_{dist\text{-}inv}(O_p) = \frac{t_{sphere} - dist_{human\text{-}joint\text{-}O_p}}{t_{sphere}}. \quad (1)$$

The second one is the probability $prob_{a_i}$ of the action-object combination. The third part enables the high-level reasoning process to influence the low-level activity recognition process by considering previous

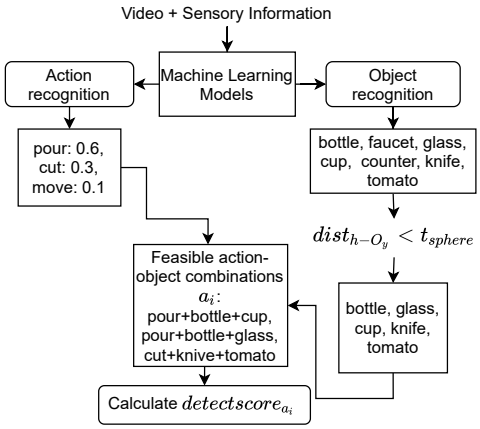

Figure 4: Matching process between action and object recognition and the ASN architecture.

recognitions of the ASN. If the action-object combination is contextually valid and contributes to the goal g, which has been assessed as the most likely one, the detection score is increased by a predefined factor q. Contextual validity is verified by comparing the predictions regarding the next activity made by the ASN in the previous temporal segment with the currently considered action-object combination. If an action-object combination does not contribute to the goal g or no goal has been determined the detection-score remains unchanged. Each detection score $detectscore_{a_i}$ is defined by

$$detectscore_{a_i} = \frac{\sum_{p}^{n_{seg}} (prob_{a_i})(s_{dist\text{-}inv(O_p)})}{n_{seg}}(q), \quad (2)$$

where index i denotes the 'i-th' feasible action object combination for the current segment, index y denotes the 'y-th' recognized object and n_{seg} equals the number of objects in the scene. The final activity recognition is the one with the highest $detectscore_{a_i}$.

Table 1: Action recognition Macro average F1 score (F1) and accuracy (Acc) of the action-affordance S-RNN and our action S-RNN on the test sets of the 4-fold cross-validation.

Metric	1. Set	2. Set	3. Set	4. Set	Average
F1 (Act-Aff)	0.88	0.62	0.81	0.77	0.77
Acc (Act-Aff)	0.92	0.76	0.82	0.76	0.82
F1 (Act)	0.88	0.66	0.75	0.66	0.74
Acc (Act)	0.89	0.76	0.78	0.73	0.79

4 EVALUATION

4.1 S-RNN Results on CAD120 Dataset

In the following we compare the performance of the action-affordance S-RNN with the action S-RNN. We use the features provided by the CAD120 dataset based on the multi-segmentation approach by Koppula et al. (2013).

For evaluation purposes we employ 4-fold cross-validation. We use the RMSprop optimizer provided by Keras with a learning rate of 0.001, and categorical crossentropy as a loss function. When training the action-affordance S-RNN and our action S-RNN for 100 epochs on a batch size of 4, the results on the different test sets are displayed in Table 1. By summing over all object features within a temporal segment, compared to training on them separately, the performance deteriorates on average by roughly 3 percent. An important reason for that is that during the training process we have not been able to accurately distinguish the respective affiliation of objects to a certain temporal segment. So we summed over the average number of objects in a temporal segment within the whole training dataset. Thus, object features of one temporal segment might be summed with object features of another segment. When object affiliation to temporal segments is known, the performance gap should decrease significantly. This hypothesis is going to be investigated in section 4.2 based on simulated data where the object affiliation is recorded.

4.2 Hybrid Model Results on Simulator Data

In the following the hybrid model performance is evaluated based on our action S-RNN classification performance and the activation spreading graphs of our adapted ASN. The data originates from the Human-Object Interaction Simulator 'HOIsim' of Zakour et al. (2021) which randomly samples activities contributing to a plan under varying kitchen environments. The simulator data consists of the plans *Breakfast*, *Serve_Lunch* and *Prepare_Lunch*. Furthermore, the plan library is extended by plans contributing to the goals of drinking *Tea*, *Coffee* and *Juice*. We compare the performance of the action-affordance S-RNN and the action S-RNN on simulator data, with the aforementioned training metrics, and combine the resulting recognitions with the object recognitions. During action and affordance recognition we consider 12 actions and 18 affordances.

Our action S-RNN that relies on the summed object feature vectors yields a macro average F1 score

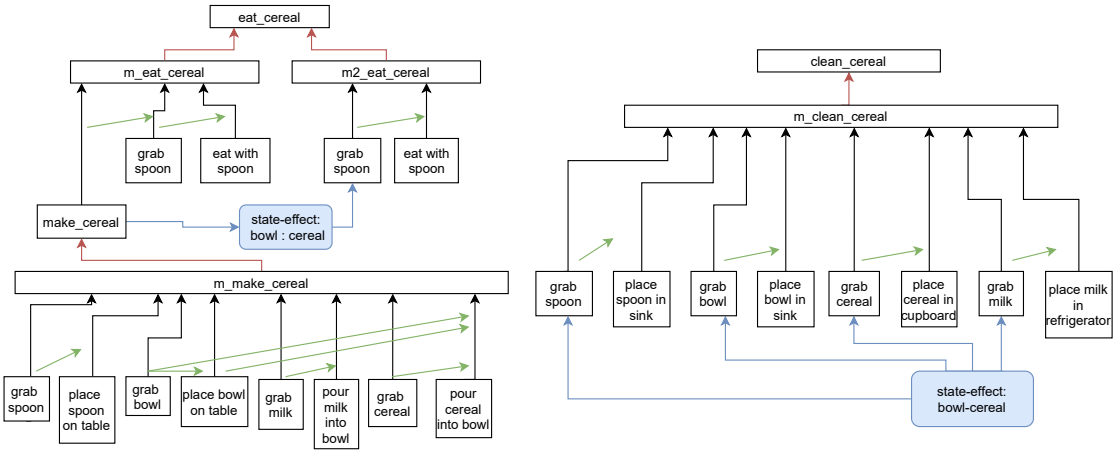

Figure 5: ASN architecture of the plan for the goals *eat_cereal* and *clean_cereal*.

of 0.97 and an accuracy of 0.99 on the simulator data. The action-affordance S-RNN returns a macro average F1 score of 0.968 and an accuracy of 0.986. In contrast to the CAD120 dataset evaluation, the performance of both models is quite similar. The similar performance metrics of the models on the simulator data might further indicate that associating objects to the correct temporal segment and action label is especially important for the action S-RNN. Given the action recognitions of our action S-RNN we consider only the three actions with the highest probabilities and the first and second closest objects based on whether the action requires one or two objects. The activity sequence that we consider, follows the underlying intention of making breakfast and its plan structure is depicted in figure 5. In figure 6 the activity recognitions are displayed on the x-axis and the activation values are displayed on the y-axis. A change in the activation value of a method shows a successful activity recognition contributing to the specific method of the plan library.

The first subgoal that the human follows is *make_cereal* which is indicated by the highest activation values of the yellow line. The green line corresponds to the method of eating cereal, which is based on the *make_cereal* subgoal. On the one hand the relatively frequent activities like *grab_milk* or *grab_spoon* contribute less to the recognition process which can be concluded from the relatively low slopes. On the other hand infrequent and more plan specific activities like *pour_cereal_into_bowl* and *pour_milk_into_bowl* lead to higher slopes of the yellow line. Activities that do not contribute to any method in the plan library do not influence the recognition process. As soon as the method *m_make_cereal* reaches the activation value 1, all activation values of the activity nodes contributing to this method are reset to 0 to enable new recognitions. Moreover, the method *m2_eat_cereal* reaches the activation value 1 as the state effect precondition of the bowl containing cereal is fulfilled. By considering state effects, the hybrid model is able to recognize the human resuming an activity after it has been interrupted. After that the human tidies up the objects used in the breakfast activity which contributes to the goal *clean_cereal*. The activities associated with the cleaning activity are enabled by the state effect of the bowl containing cereal.

While the clean cereal and make cereal method share similar activities one can see from the lower activation value lines of the method *m_make_cereal* compared to *m_clean_cereal* that there is no confusion regarding the goal assessment as long as the activity context is correct. Moreover, with the backpropagation procedure one can verify which activity recognitions serve as preconditions for subsequent activities. Thus, we can make predictions regarding future activities throughout the recognition sequence. For example in case of the *grab_cereal* recognition, one future prediction of the hybrid model is the activity *pour_cereal_into_bowl* as it is conditioned on this activity. As there is no milk in the bowl up to that point in time in the example the model also suggests the activity *grab_milk* as it has no precondition itself but contributes to the goal of eating cereal.

5 CONCLUSION

Machine Learning based frameworks like LSTMs are incapable of verifying logical preconditions, capturing multigoal activity execution and making long-term activity predictions. Hence, we propose a hybrid model consisting of a Machine Learning and a knowledge based part. Compared to the action-affordance

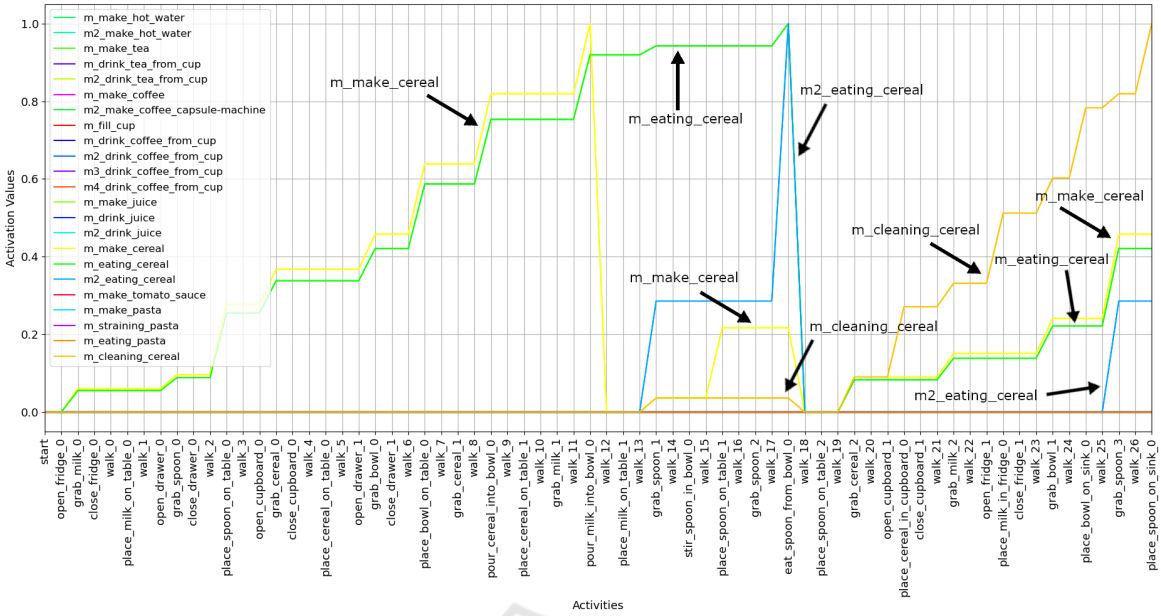

Figure 6: Activation spreading graph of a simulator test sample with the underlying goal *eat_cereal*.

S-RNN, our action S-RNN is able to obtain similar results on simulated activity data without additional effort for affordance labeling. Our proposed ASN deals with activity recognitions, misclassifications, missing activities and is capable of predicting future activitivities. As the Knowledge Base is defined by a human expert, one could address the limited validity of the plan representation by automatically extracting structured plans from sensor data. Additionally, the approach can be extended to egocentric video data. Furthermore, the object classification part has to be investigated for activity recognition on real-world data.

ACKNOWLEDGEMENT

This work has been funded by the Initiative Geriatronics by StMWi Bayern (Project X, grant no. 5140951).

REFERENCES

Bokhari, S. Z. and Kitani, K. M. (2016). Long-term activity forecasting using first-person vision. In *Asian Conference on Computer Vision*, pages 346–360. Springer.

Chen, L. and Nugent, C. D. (2019). Composite activity recognition. In *Human Activity Recognition and Behaviour Analysis*, pages 151–181. Springer.

Du, Y., Lim, Y., and Tan, Y. (2019). A novel human activity recognition and prediction in smart home based on interaction. *Sensors*, 19(20):4474.

Goldman, R. P., Geib, C. W., and Miller, C. A. (2013). A new model of plan recognition. *arXiv preprint arXiv:1301.6700*.

Jain, A., Zamir, A. R., Savarese, S., and Saxena, A. (2016). Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 5308–5317.

Koppula, H. S., Gupta, R., and Saxena, A. (2013). Learning human activities and object affordances from rgb-d videos. *The International Journal of Robotics Research*, 32(8):951–970.

Levine, S. and Williams, B. (2014). Concurrent plan recognition and execution for human-robot teams. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 24.

Ramırez, M. and Geffner, H. (2011). Goal recognition over POMDPs: Inferring the intention of a POMDP agent. In *IJCAI*, pages 2009–2014. Citeseer.

Saffar, M. T., Nicolescu, M., Nicolescu, M., and Rekabdar, B. (2015). Intent understanding using an activation spreading architecture. *Robotics*, 4(3):284–315.

Shan, D., Geng, J., Shu, M., and Fouhey, D. F. (2020). Understanding human hands in contact at internet scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9869–9878.

Yordanova, K., Lüdtke, S., Whitehouse, S., Krüger, F., Paiement, A., Mirmehdi, M., Craddock, I., and Kirste, T. (2019). Analysing cooking behaviour in home settings: Towards health monitoring. *Sensors*, 19(3):646.

Zakour, M., Mellouli, A., and Chaudhari, R. (2021). HOIsim: Synthesizing realistic 3d human-object interaction data for human activity recognition. In *2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN), pp. 1124-1131*.