Hybrid Activity and Plan Recognition for Video Streams

Roger Granada and **Ramon Fraga Pereira** and **Juarez Monteiro**
Rodrigo Barros and **Duncan Ruiz** and **Felipe Meneguzzi**
School of Informatics - Pontifical Catholic University of Rio Grande do Sul
Porto Alegre - RS, Brazil
{roger.granada, ramon.pereira, juarez.santos}@acad.pucrs.br
{rodrigo.barros, duncan.ruiz, felipe.meneguzzi}@pucrs.br

Abstract

Computer-based human activity recognition of daily living has recently attracted much interest due to its applicability to ambient assisted living. Such applications require the automatic recognition of high-level activities composed of multiple actions performed by human beings in an environment. In this work, we address the problem of activity recognition in an indoor environment, focusing on a kitchen scenario. Unlike existing approaches that identify single actions from video sequences, we also identify the goal towards which the subject of the video is pursuing. Our hybrid approach combines a deep learning architecture to analyze raw video data and identify individual actions which are then processed by a goal recognition algorithm that uses a plan library describing possible overarching activities to identify the ultimate goal of the subject in the video. Experiments show that our approach achieves the state-of-the-art for identifying cooking activities in a kitchen scenario.

1 Introduction

Plan recognition can be understood as the task of recognizing agent goals and plans based on observed interactions in an environment. These observed interactions can be either events provided by sensors or actions/activities performed by an agent. An activity can be understood as an independent set of actions that generates an interpretation to the movement that is being performed (Poppe 2010). Activity and plan recognition algorithms capable of handling real-world data are a fundamental component in many applications, such as public security (Popoola and Wang 2012), traffic monitoring (Pynadath and Wellman 1995), *etc.*

Although much research effort focuses on activity and plan recognition as separate challenges, comparatively less effort focused on attempting to identify higher-level plans from activities in video sequences (Sukthankar et al. 2014), *i.e.*, try to understand the overarching goal of subjects within a video and make the correct inference from the observed activities (Section 2). In order to address this challenge, in Section 3 we develop a hybrid approach that comprises both activity and plan recognition that identifies, from a set of candidate plans, which plan a human subject is pursuing based exclusively on still-camera video sequences. To recognize such plan, we employ an activity recognition algorithm based on convolutional neural networks (CNN), which generates a sequence of activities that are checked for temporal consistency against a plan library using a symbolic plan recognition approach modified to work with a CNN.

We evaluate our approach empirically using an existing kitchen-centered dataset in Section 4 showing that our architecture achieves good results in activities that contain a large number of frames. Finally, in Section 5 we compare our activity recognition approach to related work and point towards future work in Section 6. As supplemental material we provide a video demonstration of our architecture[1].

2 Background on Activity and Plan Recognition

Activity recognition from video sequences has been the focus of much research in the last decades (Turaga et al. 2008; Poppe 2010). Traditional approaches for activity recognition rely on hand-crafted features and domain-specific image processing algorithms and often result in limited accuracy (Bansal et al. 2013; Weinland, Ronfard, and Boyer 2006; Gorelick et al. 2007). Deep convolutional neural networks (CNNs) solve this problem by performing automatic representation learning, *i.e.*, they are capable of transforming the raw data into a set of features that properly discriminate the concepts needed for detection or classification. CNNs and similar deep approaches represent the world as a nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones (LeCun et al. 1998). Thus, given the input image, the first layer can easily identify oriented edges and similar simple concepts, whereas the subsequent hidden layers can accurately search for corners, extended contours, up to entire parts of specific objects. This final description of the image in terms of the object parts it contains can be used to a variety of tasks, such as recognizing the objects present in the image or the activities that are happening in a sequence of frames (Bengio and Courville 2016). CNNs attracted the attention of the computer vision community due to their power to classify images (Krizhevsky, Sutskever, and Hinton 2012) and videos

[1] https://youtu.be/BoiLjU1vg3E

(Karpathy et al. 2014), outperforming hand-crafted features by a significant margin.

Plan recognition is the task of recognizing how agents achieve their goals based on a set of observed interactions in an environment. These interactions can be a number of possible observed events performed by an agent in an environment, as well as actions/activities (*e.g.*, a simple movement, cook, drive), and changing properties in an environment (*e.g.*, at home, at work, resting). Recognizing agent goals and plans is important to monitor and anticipate the agent behavior, such as in stories and life understanding (Charniak and Goldman 2013), and educational environments (Uzan et al. 2015). Most plan recognition approaches require knowledge of the agent's possible plans to represent its typical behavior. In other words, this knowledge represents a set of recipes on how agents have to achieve their goals. These recipes are often encoded in plan libraries, and are used as input for most plan recognition approaches (Geib and Goldman 2005; Avrahami-Zilberbrand and Kaminka 2005; Mirsky et al. 2016). A plan library is single-root directed acyclic connected graph in which the root-node represents the top-level goal, nodes represent plan-steps, and edges that can be represented by decomposition (*e.g.*, disjunctive steps) and sequential edges. Activity recognition differs from plan recognition as the former aims to discover and extract interesting patterns in sensory data that can be interpreted as meaningful activities, while the latter concentrates on identifying high-level complex goals and intents by exploiting relationships between primitive action steps that are elements of the plan. Thus, the distinction between them is the difference between recognizing a single activity and recognizing the relationships between a set of such activities that result in a complete plan (Sukthankar et al. 2014).

3 A Hybrid Architecture for Activity and Plan Recognition

Our hybrid architecture is divided in two main parts: i) CNN-based activity recognition, and ii) CNN-backed symbolic plan recognition. Figure 1 illustrates the processing pipeline of our architecture, where CNN is a Convolutional Neural Network we use to recognize activities, and SBR is a modification of the Symbolic Behavior Recognition (Avrahami-Zilberbrand and Kaminka 2005) to recognize plans. As a machine learning technique, our approach is divided into three phases: training, validation and test phases. In training phase, the CNN receives video frames as input and learns features that represent each activity. The validation phase we use the validation data to identify the best model to use in the test phase. In particular, we observe the influence of several hyper-parameters in the network, selecting the ones that improve the performance of the model. We select and apply on test data the model that best performs on the validation data. In test phase our architecture receives video frames and a plan library containing the domain model. For each frame, the CNN identifies the activity being performed in the image. Based on the identified activity the plan recognizer returns a set of possible plans that are temporally consistent with what is recognized from the input frames. In what follows, we further detail each of the components of our approach.

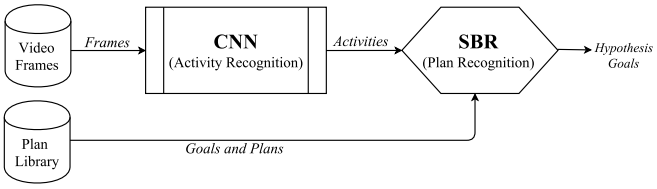

Figure 1: Pipeline of the hybrid architecture for activity and plan recognition.

3.1 CNN-based Activity Recognition

In recent years, CNNs greatly improved performance for activity recognition tasks, outperforming results achieved by the use of hand-crafted features (Karpathy et al. 2014). Researchers have developed a number of CNN architectures (Simonyan and Zisserman 2014a; 2014b; Karpathy et al. 2014), but the architecture we make use in this paper is based on GoogLeNet (Szegedy et al. 2015), mainly due to its reduced number of parameters. GoogLeNet is a 22-layer deep network based on the *Inception* module, which contains convolutional filters with different sizes, covering different clusters of information. As CNN uses supervised learning to find the best features that represent the input data, we have three steps to follow: training, validation and test phases. For all phases, we resize the input images to 256x256.

In the training phase, the CNN learns features from images using part of the dataset. The network features are learned using the mini-batch stochastic gradient descent using 0.9 of momentum. For each iteration, a mini-batch of 128 samples is constructed by randomly sampling. A random crop is applied on the input image, generating a sub-image of 224x224. A random horizontal flipping of the image is applied and each pixel is subtracted by the mean pixel of all training images. All the convolutions, including the ones inside the *Inception* modules, use the rectified linear activation (ReLU). For the weight initialization, we use the *Xavier* (Glorot and Bengio 2010) algorithm that automatically determines the scale of initialization based on the number of input neurons. To reduce over-fitting, we apply dropout on the fully-connected layers with a probability of 0.7. Learning rate is set to 10^{-3} and we drop it by a factor of 50 every epoch. Training stops after 43.5k iterations (30 epochs).

In validation phase we select the model that achieves the highest accuracy. It is important to note that while we test several parameters on validation data to adjust the model, the test phase occurs only once using the model with the highest accuracy in the validation phase.

In test phase, the CNN classifies each frame from input assigning to the image a probability score for each class (*softmax* output). The CNN may classify an image into two or more classes with exactly the same (or very close) probabilities (*e.g.*two classes containing 50% of probability). In this case, when the difference of the probabilities is lower

than a threshold (θ), we apply a heuristic to choose the class for the current frame. This heuristic consists of assigning the class of the last frame to the current frame if one of the two classes is equal to the class of the last frame. Algorithm 3.1 formalize this heuristic. The function receives as input the class of the last frame (c_{last}, the current frame (F) and a value of threshold (θ). Feeding the CNN with the current frame (F) generates a vector (C) containing the probability of the image to belong to each class. In case the difference between the two highest probabilities is less than the threshold, the function verifies whether one of the class with the highest probabilities is equal to the class of the last frame (c_{last}). If so, the class of the current frame (c) receives the same class as the last frame (c_{last}). Otherwise, the current frame receives the class that contains the highest probability (the argument with the maximum score in C).

Algorithm 1 Selecting the class for the current frame.

function SELECTCLASSCURRENTFRAME(c_{last}, F, θ)
  $C \leftarrow \text{CNN}(F)$ ▷ C is a vector of probabilities
  if $\exists i, j\ C(i) - C(j) < \theta \wedge C(i) = \arg\max_x C(x)$ **then**
    if $i = c_{last} \vee j = c_{last}$ **then**
      $c \leftarrow c_{last}$
    else
      $c \leftarrow i$
    end if
  else
    $c \leftarrow \arg\max_x C(x)$
  end if
  return c
end function

3.2 CNN-backed Symbolic Plan Recognition

To perform the task of plan recognition, we use a symbolic plan recognition approach called Symbolic Behavior Recognition (SBR). Proposed by Avrahami-Zilberbrand and Kaminka in (2005), SBR is a plan recognition approach that takes as input a plan library and a sequence of observations, in this case, a sequence of observed feature values. Feature values are used as a set of conditions to execute a plan-step in a plan library. To match observed features with plan-steps in a plan library, Avrahami-Zilberbrand and Kaminka propose an efficient matching step that maps observed features with matching plan-step nodes in a plan library. To do so, they use a feature decision tree (FDT) that maps observable features to plan-steps in a plan library. As output, SBR returns set of hypotheses plans such that each hypothesis represents a plan that achieves a top-level goal in a plan library.

Instead of using the FDT to match observations with consistent plan-steps in the plan library, we modify the SBR and replace the FDT with the CNN-backed Activity Recognition. For instance, given a video frame, the CNN-backed Activity Recognition returns which activity such video frame corresponds, and subsequently, we take this activity as input to the SBR, as shown in Figure 1. Note that to recognize goals and plans using the SBR, we must model a plan library containing a set of possible sequence of activities (*i.e.*, plan) that achieves goals. In this paper, a plan library corresponds to a model that contains a set of plans to achieve cooking menus.

4 Experiments and Results

In this section we describe the data used in the experiments (Section 4.1), how we model the plan library (Section 4.2) and the results achieved for activity (Section 4.3) and plan recognition (Section 4.4).

4.1 Dataset

For the experiments, we use the activities from ICPR 2012 Kitchen Scene Context based Gesture Recognition dataset (KSCGR[2]) (Shimada et al. 2013). The context aims to recognize cooking motions in video sequences. The dataset contains video sequences of five menus for cooking eggs in Japan: *Ham and Eggs*, *Omelet*, *Scrambled Egg*, *Boiled Egg*, and *Kinshi-Tamago*. Each menu is performed by 7 subjects: 5 actors in training datasets and 2 actors in evaluation datasets, *i.e.*, 5 cooking scenes are available for each training menu. The ground truth data contains the id of the frame and the activity being performed in the frame. Eight cooking gestures composes the dataset: *breaking*, *mixing*, *baking*, *turning*, *cutting*, *boiling*, *seasoning*, *peeling*, and *none*, where *none* means that there is not an activity being performed in the current frame. Figure 2 illustrates the timeline of the *Boiled Egg* recipe with the corresponding activities to make the recipe. We chose the KSCGR dataset instead of popular datasets for activity recognition such as UCF-101 (Soomro, Zamir, and Shah 2012) or HMDB51 (Kuehne et al. 2011) because KSCGR dataset contains the activity being performed in each frame (*e.g.*, *breaking*, *baking* and *turning*) as well as the goal achieved in the whole video sequence (*e.g.*, preparing *Ham and Eggs*, *Omelet*, *Scrambled Egg*, *etc.*). Thus, we can carry out activity recognition using activities performed in each frame and plan recognition using the steps to achieve the recipe in each video.

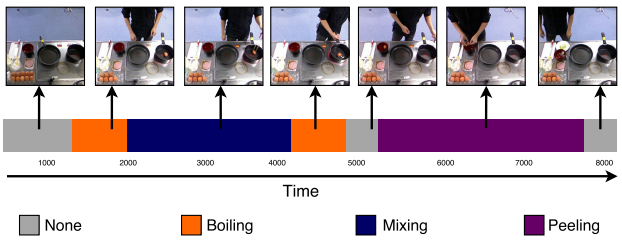

Figure 2: Timeline containing the frame/activity sequence for the *Boiled-Egg* menu.

We divided the dataset into training, validation, and test sets. The training set contains 4 subjects, each of them performing 5 recipes, *i.e.*, 20 videos and 139,196 frames in total. We use the validation set to obtain the model configuration that best fits to the training data, *i.e.*, the configuration

[2]http://www.murase.m.is.nagoya-u.ac.jp/KSCGR/

with the highest accuracy. This set contains 1 subject performing 5 recipes with 32,897 frames in total. We use the test set to assess the accuracy of the selected model in unseen data. This set contains 2 subjects, each performing 5 recipes, *i.e.*, 10 videos with 55,781 frames in total.

4.2 Plan Library Modeling

For recognizing goals and plans, we model a plan library containing knowledge of the agent's possible goals and plans based on the KSCGR dataset (Shimada et al. 2013). We model each recipe as a top-level goal in the plan library. Videos in the Shimada *et al.* (2013), correspond to sequences of cooking gestures that result in a menu (recipe). Based on videos from the training set, we model all possible plans for achieving each possible menu (*i.e.*, top-level goal). We consider that a sequence of cooking gestures is analogous to a sequence of plan-steps, *i.e.*, a plan in the plan library. Figure 3 shows part of the plan library that we model, representing the set of plans and plan-steps to achieve the top-level goals *Boiled Egg* and *Kinshi-Egg*.

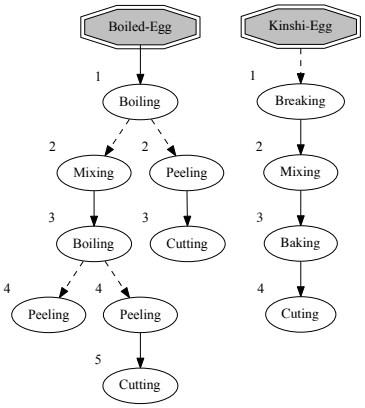

Figure 3: Set of plans to achieve the menus *Boiled Egg* and *Kinshi-Egg*.

4.3 Activity Recognition Results

To evaluate the activity recognition module, we select the CNN model that achieves the best accuracy using the validation set. The best model of the GoogLeNet architecture achieved 76% of accuracy in the validation dataset after 43,500 iterations (30 epochs). Our experiment consists in evaluating the performance of the trained network on the test set. We measure the performance of the system in terms of Accuracy (ACC), Precision (P), Recall (R) and F-measure (F) for each activity. Accuracy is the number of correctly classified images over the total number of images in the test set, Precision of a class is the true positive values divided by the total number of predicted values to that class, Recall is the true positive values divided by the total number of true values, and F-measure is a harmonic mean of Precision and Recall given by $F = \frac{2 \times P \times R}{P+R}$.

Table 1 shows the precision, recall, F-measure and accuracy scores for each activity. As we can see that the highest

Table 1: Precision, Recall, F-measure and Accuracy for each activity using the KSCGR test data.

Activity	Precision	Recall	F-measure	Accuracy
None	0.65	**0.97**	0.78	0.64
Breaking	0.44	0.41	0.42	0.27
Mixing	0.67	0.34	0.45	0.29
Baking	0.74	0.88	**0.80**	**0.67**
Turning	0.77	0.38	0.51	0.34
Cutting	0.87	0.63	0.73	0.58
Boiling	0.61	0.34	0.43	0.28
Seasoning	**0.89**	0.37	0.52	0.35
Peeling	0.72	0.10	0.18	0.09

value of precision (89%) is achieved by *Seasoning*. A high value of precision means that the CNN can adjust very well the features to identify the class, while low values means that the CNN can not identify relevant features to identify the class among other classes. For example, *Breaking* achieved the lowest precision score (44%), which means that the features of the class *Breaking* are not so evident as the features of *Seasoning*. A possible reason that these features are not so evident is due to the number of examples given to the CNN with the true label *Breaking*.

Using the output of the CNN we perform an analysis to see how predicted classes are classified in relation to the true classes. In order to perform this analysis we generate a normalized confusion matrix, depicted in Figure 4, where rows represent the true classes and columns the predicted classes. The value of each cell is represented by shades of blue going chromatically from a dark blue for higher values to a light blue for lower values. All values are normalized, *i.e.*, predicted values are divided by the total number of true values for each cell. The main diagonal means the correct predicted labels and values out of the main diagonal are incorrect predicted labels.

Analyzing the results in Figure 4 for *Breaking* we can ob-

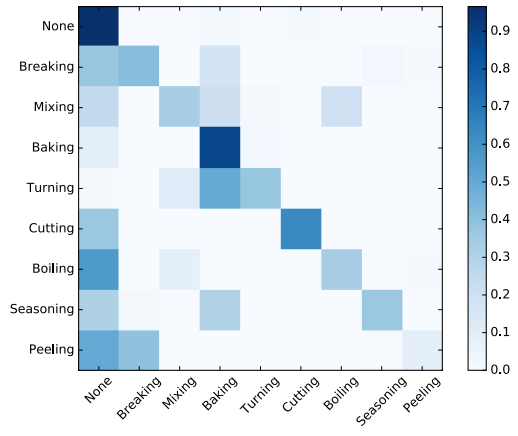

Figure 4: Normalized confusion matrix for the activity recognition task.

serve that the system classify wrongly the activity as *Peeling*. The misclassification cna be explained because both activities occur in the same region of the frame using the same objects (*e.g.*, in *Breaking* activity, the egg is broken on the bowl and the white egg and yolk fall down to the bowl on the left corner of the scene, while in *Peeling* activity, the egg is peeled on the bowl and the eggshell falls down to the bowl on the same left corner of the scene. Other classes such as *None* and *Baking* are wrongly classified with many other classes. Although both activities have high values in the main diagonal, the precision is smoothed with the misclassification to other classes. For example, *Baking* is classified many times as *Turning*. As occurred with *Breaking* and *Peeling*, this misclassification can be understood since both activities occur in the same region of the scene with the same objects and little details are changed between activities.

The activity *None* achieved the highest recall (97%), followed by *Baking* (88%). The high recalls may be due to the unbalanced nature of the dataset, since the *None* activity has the largest number of frames (29% of total) followed by *Baking* (25% of total). This unbalancing in the dataset makes the CNN learns much more features about these classes than the other classes. Unlike the *Baking* activity that has almost a standard behavior in the scene (*e.g.*, the egg baking inside the pan), the *None* activity is classified as anything but any other activity. The *None* activity includes frames where the subject is preparing kitchen utensils, moving pans, *etc.*and inter-activity frames such as removing the egg from boiling to peeling. The low variability in regions of the scene and the large number of frames is the possible reason for *Baking* to achieve the highest F-measure (80%). Although the unbalanced nature of the dataset, the values of accuracy follow the F-measure scores, having the lowest value obtained by *Peeling* and the highest value obtained *Baking*. The total accuracy of the model in test set achieves 69%.

Following the work by Bansal *et al.* (2013) we apply the idea that some activities only occur in consequence of previous activities. For example, the cooker cannot mixing an egg (*Mixing*) without having broken the egg (*Breaking*) before. Thus, we need to learn the sequence of activities from training phase and in test phase only accept sequences of activities that occur in training phase. In order to do that, we create a directed graph (digraph) containing the sequence of activities learned from each recipe in training phase. Each node of the graph corresponds to an activity and the edge between two nodes corresponds to the direct sequence of activities in training set. For example, in Figure 5 we can see a direct edge between *Boiling* and *Peeling* (red edge), meaning that in training phase the *Peeling* activity succeed the *Boiling* activity.

Unlike Bansal *et al.* (2013) that achieved an increase of 7% in accuracy when applied this filtering process, the improvement is not very substantial in our experiments. Our total accuracy increased about 1% when applied such filtering process, achieving 70% of accuracy. The lower improvement in our experiments when compared with the work by Bansal *et al.* might be due to a better identification of activities that occur in sequence by the CNN.

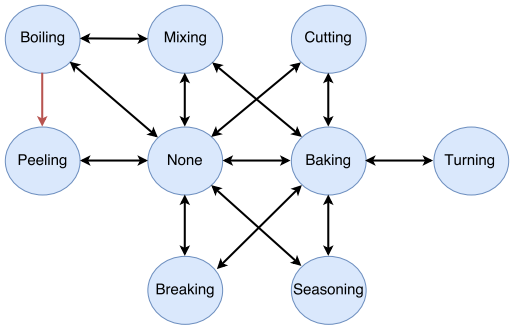

Figure 5: Digraph containing the sequence of activities in training dataset.

4.4 Plan Recognition Results

We evaluated the whole pipeline using the number of correct recipes yield by the plan recognizer. In order to do so, we feed the recognizer with activities predicted by the CNN and calculate the rate of correct predictions. This rate is calculated counting the number of correct recipes divided by the total number of recipes yield by the SBR. For example, if the plan recognizer outputs two possible recipes in the end of the pipeline and one of them is the correct recipe, we assign 0.5 to its correctness. In case where the plan recognizer yields four recipes and none of them is the target recipe, the score is set to 0. The final score corresponds to the sum of all values divided by the total number of recipes. Table 2 presents the two parts of the test set with their true recipes followed by the recipes predicted by the plan recognizer and their scores.

Analysing Table 2, we observe that some plans obtained zero score. When comparing the activity sequence from training set and test set we observe that some activity sequences occur only in test set and not in training set. For example, the plan for *Kinshi-Egg* in test set 11 contains the activity sequence: *Breaking* $\rightarrow$ *Mixing* $\rightarrow$ *Baking* $\rightarrow$ *Turning* $\rightarrow$ *Baking* $\rightarrow$ *Cutting*, while in all training set the *Turning* activity is never associated with the plan *Kinshi-Egg*. Specif-

Table 2: True and predicted labels for recipes according for each testing set.

#	True Recipe	Predicted Recipes	Score
10	Boild-Egg	Scramble-Egg, Omelette, Ham-Egg	0.00
	Ham-Egg	Scramble-Egg, Omelette	0.00
	Kinshi-Egg	Kinshi-Egg	1.00
	Omelette	Scramble-Egg, Omelette	0.50
	Scramble-Egg	Ham-Egg	0.00
11	Boild-Egg	Kinshi-Egg, Omelette, Ham-Egg	0.00
	Ham-Egg	Scramble-Egg	0.00
	Kinshi-Egg	Scramble-Egg, Omelette, Ham-Egg	0.00
	Omelette	Kinshi-Egg, Scramble-Egg, Omelette, Ham-Egg	0.25
	Scramble-Egg	Kinshi-Egg	0.00
		Total:	0.18

ically, there are 3 plans in test set that do not appear in training set, and thus, leading the recognizer to errors. Analysing the sequences of activities we observe that some sequences occur in more than one recipe. For example, both *Scramble Egg* and *Omelette* recipes contain the same sequence of activities: *breaking* → *cutting* → *seasoning* → *mixing* → *baking* → *mixing* → *baking*. When calculating the correctness of such sequence we do not consider that both recipes are correct, but instead consider that the output of the plan recognizer should yield the correct recipe, *i.e.*, the true recipe. We understand that there are some drawbacks of using plan libraries to recognized goals and plans. First, designing plan libraries requires much design effort, and therefore, we argue that for encoding a plan library that contains results for all goals and plans that can be encountered in a non-trivial domain is impossible or intractable to be designed. Second, symbolic plan libraries are not flexible enough to deal with noisy data in input. For example, consider the excerpt presented in Figure 3 where the correct sequence of activities in the plan library is described as *Boiling* → *Peeling* → *Cutting*. If the output of the CNN yields at least one frame out of this sequence, the plan library would not identify the correct recipe.

5 Related work

Much research effort has been made for recognize activities in video frames (Gorelick et al. 2007; Shimada et al. 2013; Bansal et al. 2013; Simonyan and Zisserman 2014a; Karpathy et al. 2014). Simonyan and Zisserman (2014a) propose the two-stream convolutional network architecture, incorporating spatial and temporal networks. Spatial network is composed by still images while the temporal network is composed by hand-crafted features that encode the temporal domain, such as optical flow. Karpathy *et al.* (2014) extends the idea of Simonyan and Zisserman providing an architecture that processes the input at two different spatial resolutions. In that paper, several methods of two-stream network fusion are tested. A survey by Poppe (2010) provides a complete overview on methods that use hand-crafted features to recognize activities in video frames. A work closely related to ours is the one by Bansal *et al.* (2013) that perform daily life cooking activity recognition based on hand-crafted features for hand movements and object use. Their method first detects hand regions through color segmentation and skin identification. Since objects may give hints of the activity (*e.g.*, the use of the knife may indicate the cutting activity), objects are identified as "*Not in use*" and "*In use*". A dynamic Support Vector Machine (SVM) and Hidden Markov Model (HMM) hybrid model combines the structural and temporal information to jointly infer the activity. In experiments, Bansal *et al.* achieving 64% of accuracy using the KSCGR dataset (Shimada et al. 2013). In order to improve the performance of the system, they propose a post-processing of the output, removing noise frames, *i.e.*, frames that are classified wrongly among a cluster of correctly classified frames. A smoothing operation using majority voting removes the noise frames, re-classifying them according to their neighborhood. As some activities are temporally dependent of others, *e.g.*, *peeling* only occurs after *boiling*, they create a context grammar to select the the most likely guess for misclassified frames. Using the post processing, Bansal *et al.* increased about 7% of the accuracy for the activity recognition, achieving a final accuracy of 71%. Holtzen *et al.* (2016) propose a method to infer a human's intent from partially observed RGBD videos. They represent intents as a probabilistic And-Or graph structure which describes a relationship between activities and plans. The main idea of their work is to construct an And-Or graph that explains the observed sequence of actions performed in a continuous 2D plane.

6 Conclusion and Future Work

In this work we proposed a hybrid architecture for activity and plan recognition. The pipeline of the architecture includes a deep learning algorithm called Convolutional Neural Network (CNN) to extract features from images and classify unseen frames. Experiments showed that the CNN can learn features automatically for activity recognition using the kitchen scene KSCGR dataset, achieving results comparable to work that employs hand-crafted features. We modify a symbolic plan recognition approach called Symbolic Behavior Recognition (SBR) to work with the CNN and identify the goal that describes the sequence of activities. As some sequences of activities occur only in testing phase and not in training phase, we observed the limitations of using plan library as plan recognizer.

As future work, we intend to employ other deep learning architectures such as Long-Short Term Memory networks (LSTM) (Hochreiter and Schmidhuber 1997) or two-stream CNNs (Simonyan and Zisserman 2014a) since they can encode temporal features better than spatial CNNs as the one used in this work. As the plan library have limitations such as modeling the entire domain before testing, we intend to use a more flexible approach for plan recognition, *i.e.*, a planning-based plan recognition approach, such as (Ramírez and Geffner 2009; 2010; Pereira and Meneguzzi 2016; Pereira, Oren, and Meneguzzi 2017). Planning-based plan recognition approaches do not limit recognition capabilities to unknown plans (while library-based approaches do, *e.g.*, SBR), since such approaches do not have to represent all plans to achieve all goals. To do so, such approaches use a planning domain definition (a set of predicates and actions), an initial state, a set of candidate goals, and observations (actions or properties). The object identification may also be explored and translated to predicates, since they can give clues of movements being performed by the subject. For example, the presence of a knife in the hand of the subject may indicate that a cutting activity is being performed, representing a set of properties in an observed state. Using object recognition, a plan recognizer that use such properties may be employed, improving the identification of the recipes.

Acknowledgement

This paper was achieved in cooperation with HP Brasil Indústria e Comércio de Equipamentos Eletrônicos LTDA. using incentives of Brazilian Informatics Law (Law nº 8.2.48 of 1991).

References

Avrahami-Zilberbrand, D., and Kaminka, G. A. 2005. Fast and Complete Symbolic Plan Recognition. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*, 653–658.

Bansal, S.; Khandelwal, S.; Gupta, S.; and Goyal, D. 2013. Kitchen activity recognition based on scene context. In *Proceedings of the 2013 IEEE International Conference on Image Processing (ICIP 2013)*, 3461–3465. IEEE.

Bengio, I. G. Y., and Courville, A. 2016. Deep learning. Book in preparation for MIT Press.

Charniak, E., and Goldman, R. P. 2013. Plan recognition in stories and in life. *Computing Research Repository (CoRR)* abs/1304.1497.

Geib, C. W., and Goldman, R. P. 2005. Partial Observability and Probabilistic Plan/Goal Recognition. In *Proceedings of the International Workshop on Modeling Others from Observations (MOO-2005)*.

Glorot, X., and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS10). Society for Artificial Intelligence and Statistics*, volume 9, 249–256.

Gorelick, L.; Blank, M.; Shechtman, E.; Irani, M.; and Basri, R. 2007. Actions as space-time shapes. *IEEE transactions on pattern analysis and machine intelligence* 29(12):2247–2253.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Holtzen, S.; Zhao, Y.; Gao, T.; Tenenbaum, J. B.; and Zhu, S.-C. 2016. Inferring human intent from video by sampling hierarchical plans. In *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016)*, 1489–1496. IEEE.

Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; and Fei-Fei, L. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the 2014 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR' 14)*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; and Serre, T. 2011. Hmdb: a large video database for human motion recognition. In *2011 International Conference on Computer Vision*, 2556–2563. IEEE.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Mirsky, R.; Stern, R.; Gal, Y. K.; and Kalech, M. 2016. Sequential plan recognition. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI 2016)*, 401–407.

Pereira, R. F., and Meneguzzi, F. 2016. Landmark-Based Plan Recognition. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI 2016)*.

Pereira, R. F.; Oren, N.; and Meneguzzi, F. 2017. Landmark-Based Heuristics for Goal Recognition. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI 2017)*, To Appear.

Popoola, O. P., and Wang, K. 2012. Video-based abnormal human behavior recognitiona review. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42(6):865–878.

Poppe, R. 2010. A survey on vision-based human action recognition. *Image and Vision Computing* 28(6):976–990.

Pynadath, D. V., and Wellman, M. P. 1995. Accounting for context in plan recognition, with application to traffic monitoring. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, 472–481. Morgan Kaufmann Publishers Inc.

Ramírez, M., and Geffner, H. 2009. Plan Recognition as Planning. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI 2009)*.

Ramírez, M., and Geffner, H. 2010. Probabilistic Plan Recognition Using Off-the-Shelf Classical Planners. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI 2010)*.

Shimada, A.; Kondo, K.; Deguchi, D.; Morin, G.; and Stern, H. 2013. Kitchen scene context based gesture recognition: A contest in icpr2012. In *Advances in depth image analysis and applications*. Springer. 168–185.

Simonyan, K., and Zisserman, A. 2014a. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, 568–576.

Simonyan, K., and Zisserman, A. 2014b. Very deep convolutional networks for large-scale image recognition. *Computing Research Repository (CoRR)* abs/1409.1556.

Soomro, K.; Zamir, A. R.; and Shah, M. 2012. Ucf101: A dataset of 101 human actions classes from videos in the wild. *Computing Research Repository (CoRR)* abs/1212.0402.

Sukthankar, G.; Geib, C.; Bui, H. H.; Pynadath, D.; and Goldman, R. P. 2014. *Plan, activity, and intent recognition: theory and practice*. Newnes.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015)*, 1–9.

Turaga, P.; Chellappa, R.; Subrahmanian, V. S.; and Udrea, O. 2008. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology* 18(11):1473–1488.

Uzan, O.; Dekel, R.; Seri, O.; and Gal, Y. K. 2015. Plan recognition for exploratory learning environments using interleaved temporal search. *AI Magazine* 36(2):10–21.

Weinland, D.; Ronfard, R.; and Boyer, E. 2006. Free viewpoint action recognition using motion history volumes. *Computer vision and image understanding* 104(2):249–257.